AWS Well-Architected Framework

Pilier Fiabilité

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Pilier Fiabilité: AWS Well-Architected Framework

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Pilier Fiabilité – AWS Well-Architected Framework.

Date de publication : 27 juin 2024 (Révisions du document)

Ce livre blanc porte sur le pilier Fiabilité du AWS Well-Architected Framework. Il fournit des conseils pour aider les clients à appliquer les bonnes pratiques de conception, de livraison et de maintenance des environnements Amazon Web Services (AWS).

Introduction

La AWS Well-Architected Framework vous aide à comprendre les avantages et les inconvénients des décisions que vous prenez lors de la création de charges de travail sur AWS. En utilisant ce cadre, vous apprendrez les bonnes pratiques architecturales pour concevoir et exploiter des charges de travail fiables, sécurisés, efficaces, économiques et durables dans le cloud. Il permet d'évaluer régulièrement vos architectures par rapport aux bonnes pratiques et d'identifier les axes d'amélioration. Nous pensons qu'une charge de travail bien structurée augmente considérablement les chances de réussite des entreprises.

Le cadre AWS Well-Architected Framework repose sur six piliers :

- Excellence opérationnelle
- Sécurité
- Fiabilité
- Efficacité des performances
- Optimisation des coûts
- Durabilité

Ce livre blanc se concentre sur le pilier Fiabilité et sur la manière de l'appliquer à vos solutions. La fiabilité peut être difficile à atteindre dans les environnements sur site traditionnels, en raison de points de défaillance uniques, d'un manque d'automatisation et d'élasticité. L'adoption des pratiques détaillées dans ce document permettra de construire des architectures résilientes aux fondations solides, avec une gestion cohérente des modifications et des processus éprouvés de reprise en cas de défaillance.

Le présent document est conçu pour ceux et celles qui sont dépositaires de rôles technologiques, comme les directeurs de la technologie, les architectes, les développeurs et les membres de l'équipe

d'exploitation. Après avoir lu ce document, vous comprendrez les bonnes pratiques et les stratégies AWS à utiliser lors de la conception d'architectures cloud fiables. Ce livre blanc comprend des détails d'implémentation de haut niveau et des modèles d'architecture, ainsi que des références à des ressources supplémentaires.

Fiabilité

Le pilier Fiabilité englobe la capacité d'une charge de travail à exécuter sa fonction de manière correcte et cohérente et ce, en temps utile. Cela inclut la possibilité d'exploiter et de tester la charge de travail tout au long de son cycle de vie. Ce livre blanc fournit des bonnes pratiques détaillées pour la mise en œuvre de charges de travail fiables sur AWS.

Rubriques

Modèle de responsabilité partagée pour la résilience

La résilience est une responsabilité partagée entre AWS et vous. Il est important que vous compreniez comment la reprise après sinistre (DR) et la disponibilité, qui font partie de la résilience, fonctionnent dans le cadre de ce modèle partagé.

AWS responsibility - Resiliency of the cloud (Responsabilité AWS : modèle de responsabilité partagée pour la résilience)

AWS est responsable de la résilience de l'infrastructure qui fait fonctionner tous les services proposés dans le AWS Cloud. Cette infrastructure comprend le matériel, les logiciels, les réseaux et les installations qui permettent d'exécuter les services AWS Cloud. AWS déploie des efforts commercialement raisonnables pour rendre ces services AWS Cloud disponibles, en veillant à ce que la disponibilité des services respecte ou dépasse les accords de niveau de service (SLA) AWS.

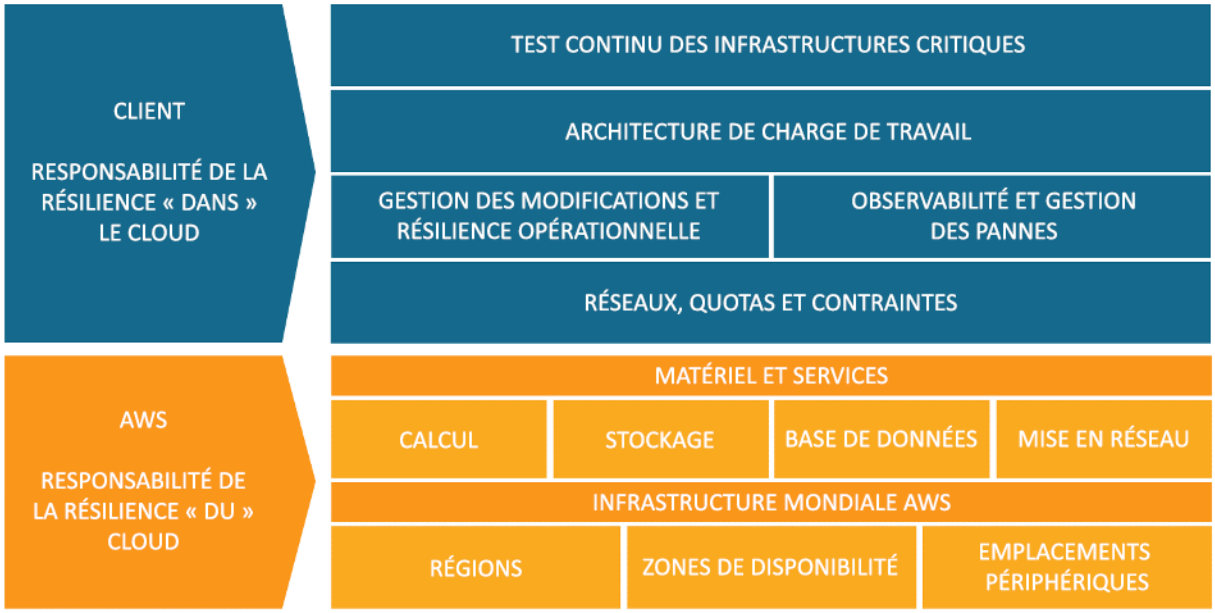
L'infrastructure de cloud mondial AWS est conçue pour permettre aux clients de créer des architectures de charges de travail hautement résilientes. Chaque Région AWS est entièrement isolée et se compose de plusieurs zones de disponibilité, qui sont des partitions physiquement isolées de l'infrastructure. Les zones de disponibilité isolent les défaillances susceptibles d'affecter la résilience des charges de travail, en les empêchant d'avoir un impact sur les autres zones de la région. Toutes les zones de disponibilité d'une Région AWS sont interconnectées avec un réseau à large bande passante et à faible latence, qui utilise une fibre métropolitaine dédiée entièrement redondante fournissant un réseau à haut débit et à faible latence entre les AZ. Tout le trafic entre

les zones est chiffré. Les performances du réseau sont suffisantes pour réaliser une réplication synchrone entre les zones. Si une application est partitionnée sur plusieurs AZ, vous êtes mieux isolé et protégé contre les problèmes tels que les pannes de courant, la foudre, les tornades, les tremblements de terre, etc.

Responsabilité du client : résilience dans le cloud

Votre responsabilité est déterminée par les services AWS Cloud que vous choisissez. Cela détermine la quantité de travail de configuration que vous devez effectuer dans le cadre de vos responsabilités en matière de résilience. Par exemple, un service tel que Amazon Elastic Compute Cloud (Amazon EC2) exige du client qu'il effectue toutes les tâches de configuration et de gestion de la résilience nécessaires. Les clients qui déploient des instances Amazon EC2 sont responsables du déploiement des instances Amazon EC2 sur plusieurs sites (tels que les zones de disponibilité AWS), de la mise en œuvre de l'auto-réparation à l'aide de services tels que Auto Scaling, et du respect des bonnes pratiques en matière d'architecture de charge de travail résiliente pour les applications installées sur les instances. Pour les services gérés, tels que Amazon S3 et Amazon DynamoDB, AWS exploite la couche infrastructure, le système d'exploitation et les plateformes, et les clients accèdent aux points de terminaison pour stocker et récupérer les données. Vous êtes responsable de la gestion de la résilience de vos données, y compris des stratégies de sauvegarde, de gestion des versions et de réplication.

Le déploiement de votre charge de travail dans plusieurs zones de disponibilité d'une Région AWS fait partie d'une stratégie de haute disponibilité conçue pour protéger les charges de travail en isolant les problèmes dans une zone de disponibilité donnée. Cette stratégie utilise la redondance des autres zones de disponibilité pour continuer à répondre aux requêtes. Une architecture Multi-AZ s'inscrit également dans une stratégie DR conçue pour mieux isoler et protéger les charges de travail contre des problèmes tels que les pannes de courant, la foudre, les tornades, les tremblements de terre, etc. Les stratégies de DR peuvent également faire appel à de multiples Régions AWS. Par exemple, dans une configuration active/passive, le service de la charge de travail passe de sa région active à sa région DR si la région active ne peut plus répondre aux requêtes.



Responsabilité de la résilience dans le cloud et de la résilience du cloud pour les clients et AWS.

Vous pouvez utiliser les services AWS pour atteindre vos objectifs de résilience. En tant que client, vous êtes responsable de la gestion des aspects suivants de votre système pour atteindre la résilience dans le cloud. Pour obtenir plus de détails sur chaque service en particulier, consultez la documentation AWS.

Réseaux, quotas et contraintes

- Les bonnes pratiques pour ce domaine du modèle de responsabilité partagée sont décrites en détail dans la section Fondations.
- Planifiez votre architecture en prévoyant une marge de manœuvre suffisante et comprenez les contraintes et les quotas de service que vous incorporez, en fonction des augmentations prévues de la demande de charge, le cas échéant.
- Concevez votre topologie de réseau pour qu'elle soit hautement disponible, redondante et évolutive.

Gestion des modifications et résilience opérationnelle

- La gestion des modifications comprend la manière d'introduire et de gérer le changement dans votre environnement. La mise en œuvre du changement nécessite la création et le maintien à jour de runbooks et de stratégies de déploiement pour votre application et votre infrastructure.

- Une stratégie résiliente de surveillance des ressources de charge de travail prend en compte tous les composants, y compris les métriques techniques et commerciales, les notifications, l'automatisation et l'analyse.
- Les charges de travail dans le cloud doivent s'adapter aux changements d'échelle de la demande en réaction aux dégradations ou aux fluctuations de l'utilisation.

Observabilité et gestion des pannes

- L'observation des pannes par le biais de la surveillance est nécessaire pour automatiser la réparation afin que vos charges de travail puissent résister aux pannes de composants.
- Pour gérer les pannes, il faut sauvegarder les données, appliquer les bonnes pratiques pour permettre à votre charge de travail de résister aux pannes des composants et planifier la reprise après sinistre.

Architecture de charge de travail

- L'architecture de votre charge de travail comprend la manière dont vous concevez les services autour des domaines d'activité, dont vous appliquez l'architecture orientée services (SOA) et la conception des systèmes distribués pour éviter les pannes, et dont vous intégrez des capacités telles que la limitation, les tentatives, la gestion des files d'attente, les délais et les leviers d'urgence.
- Appuyez-vous sur des solutions AWS éprouvées, les ressources Amazon Builders Library et les modèles sans serveur pour vous aligner sur les bonnes pratiques et lancer les mises en œuvre.
- Utilisez l'amélioration continue pour décomposer votre système en services distribués afin d'évoluer et d'innover plus rapidement. Utilisez les conseils sur les microservices AWS et les options de services gérés pour simplifier et accélérer votre capacité à incorporer des modifications et à innover.

Test continu des infrastructures critiques

- Tester la fiabilité signifie tester aux niveaux fonctionnel, des performances et du chaos. Cela nécessite en outre d'adopter des pratiques d'analyse d'incidents et de jeux de rôle pour acquérir une expertise dans la résolution des problèmes qui ne sont pas bien compris.

- Pour les applications tout cloud et hybrides, le fait de savoir comment votre application se comporte lorsque des problèmes surviennent ou que des composants tombent en panne permet une reprise rapide et fiable après une panne.
- Créez et documentez des expériences reproductibles pour comprendre comment votre système se comporte lorsque les objets ne fonctionnent pas comme prévu. Ces tests prouveront l'efficacité de votre résilience globale et fourniront une boucle de rétroaction pour vos procédures opérationnelles avant de vous confronter à des scénarios de panne réels.

Principes de conception

Dans le cloud, il existe un certain nombre de principes qui peuvent vous aider à renforcer la fiabilité. Gardez les éléments suivants à l'esprit lorsque nous aborderons les meilleures pratiques :

- Récupération automatique après une panne : en contrôlant les indicateurs clés de performance d'une charge de travail, vous pouvez déclencher l'automatisation en cas de transgression d'un seuil. Ces KPI doivent couvrir la valeur commerciale et non des aspects techniques du fonctionnement du service. Cela permet la création de notifications automatiques, le suivi des pannes et l'exécution de processus de récupération automatique qui contournent ou corrigent les pannes. Une automatisation plus sophistiquée rend possible l'anticipation et la correction des pannes avant qu'elles ne se produisent.
- Test des procédures de reprise : dans un environnement sur site, des tests sont souvent conduits pour prouver que la charge de travail fonctionne dans un scénario particulier. Ces tests ne sont généralement pas utilisés pour valider les stratégies de récupération. Dans le cloud, vous pouvez tester de quelle façon votre charge de travail cesse de fonctionner et valider vos procédures de récupération. Vous pouvez utiliser l'automatisation pour simuler différentes pannes ou recréer les scénarios qui ont déjà conduit à des pannes. Cette approche permet de réduire les risques en exposant les chemins de défaillance que vous pouvez tester et corriger avant qu'un scénario de panne réelle ne survienne.
- Mise à l'échelle horizontale pour augmenter la disponibilité de la charge de travail : remplacez une ressource volumineuse par plusieurs petites ressources pour réduire l'impact d'une panne unique sur la charge de travail globale. Répartissez les demandes entre plusieurs ressources plus petites pour garantir qu'elles ne partagent pas un point de panne commun.
- Une capacité réellement adaptée à vos besoins : une cause courante de panne des charges de travail sur site est la saturation des ressources, lorsque les demandes ciblant une charge de travail en dépassent la capacité (c'est souvent l'objectif des attaques par déni de service). Dans le cloud, vous pouvez contrôler la demande et l'utilisation de la charge de travail. Vous pouvez

aussi automatiser l'ajout ou la suppression de ressources afin de maintenir le niveau optimal de satisfaction de la demande sans surallocation ou sous-allocation. Des limites demeurent, mais certains quotas peuvent être contrôlés et d'autres gérés (consultez Gestion des quotas et contraintes de service).

- Gérer les changements avec l'automatisation : les modifications apportées à l'infrastructure doivent être appliquées via l'automatisation. Les modifications qui doivent être gérées incluent celles apportées à l'automatisation et qui peuvent ensuite être suivies et vérifiées.

Définitions

Ce livre blanc aborde la fiabilité dans le cloud et décrit les bonnes pratiques pour ces quatre domaines :

- Fondations
- Architecture de charge de travail
- Gestion des modifications
- Gestion des défaillances

Pour atteindre la fiabilité, vous devez commencer par les fondations : un environnement où les quotas de service et la topologie réseau s'adaptent à la charge de travail. L'architecture de la charge de travail du système distribué doit être conçue pour prévenir et atténuer les défaillances. La charge de travail doit gérer les modifications au niveau de la demande ou des exigences. Elle doit être conçue pour détecter les défaillances et se réparer automatiquement.

Rubriques

- Résilience et composants de la fiabilité
- Disponibilité
- Objectifs de de reprise après sinistre

Résilience et composants de la fiabilité

La fiabilité d'une charge de travail dans le cloud dépend de plusieurs facteurs, dont le principal est la résilience :

- La résilience est la capacité d'une charge de travail à récupérer après des perturbations de l'infrastructure ou du service, d'acquérir de manière dynamique les ressources de calcul pour satisfaire la demande et d'atténuer les perturbations telles que les erreurs de configuration ou les problèmes réseau temporaires.

Les autres facteurs qui affectent la fiabilité de la charge de travail sont les suivants :

- L'excellence opérationnelle qui comprend l'automatisation des modifications, l'utilisation de manuels pour corriger les pannes et les examens de disponibilité opérationnelle pour vérifier que les applications sont prêtes pour les opérations de production.
- La sécurité qui englobe la prévention des dommages causés aux données ou à l'infrastructure par des acteurs malveillants pour prévenir tout impact sur la disponibilité. Par exemple, chiffrez les sauvegardes pour vous sécuriser les données.
- L'efficacité des performances qui intègre la conception pour maximiser les taux de requêtes et réduire au maximum les latences de votre charge de travail.
- L'optimisation des coûts, ce qui implique des compromis, tels que le choix entre une hausse des dépenses sur les instances EC2 pour atteindre la stabilité statique ou le recours à la mise à l'échelle automatique pour augmenter la capacité en fonction des besoins.

La résilience est l'objectif principal de ce livre blanc.

Les quatre autres aspects sont également importants et sont couverts par leurs piliers respectifs du Cadre AWS Well-Architected. Un grand nombre des bonnes pratiques décrites ici traitent également des aspects liés à la fiabilité, mais l'accent est mis sur la résilience.

Disponibilité

La disponibilité (également appelée disponibilité des services) est à la fois une métrique couramment utilisée pour mesurer quantitativement la résilience et un objectif de résilience.

- La disponibilité correspond au pourcentage de temps pendant lequel une charge de travail est disponible à l'utilisation.

Disponible à l'utilisation signifie que la charge de travail exécute sa fonction chaque fois que nécessaire.

Ce pourcentage se calcule sur une période de temps, par exemple un mois, un an ou les trois dernières années. Dans son interprétation la plus stricte, la disponibilité est réduite chaque fois que l'application ne fonctionne pas normalement, y compris pendant les interruptions planifiées et non planifiées. Nous définissons la disponibilité comme suit :

$$Disponibilité = \frac{Durée\ de\ disponibilité\ à\ l'utilisation}{Durée\ totale}$$

- La disponibilité est un pourcentage de temps de fonctionnement (par exemple, 99,9 %) sur une période (généralement un mois ou un an)
- Un raccourci courant consiste à parler du « nombre de neuf ». Par exemple, « cinq 9 » correspond à une disponibilité de 99,999 %.
- Certains clients choisissent d'exclure les temps d'arrêt de service planifiés (par exemple, maintenance planifiée) du temps total dans la formule. Cependant, cela n'est pas conseillé, car vos utilisateurs voudront probablement utiliser votre service pendant ces périodes.

Voici un tableau des objectifs courants de disponibilité des applications et de la durée maximale des interruptions qui peuvent se produire sur une année, tout en continuant d'atteindre l'objectif. Le tableau contient des exemples des types d'applications généralement rencontrées à chaque niveau de disponibilité. Tout au long de ce document, nous ferons référence à ces valeurs.

Disponibilité	Indisponibilité maximale (par an)	Catégories d'applications
99 %	3 jours 15 heures	Tâches de traitement par lots, d'extraction de données, de transfert et de chargement
99,9 %	8 heures 45 minutes	Outils internes, tels que la gestion des connaissances, le suivi des projets
99,95 %	4 heures 22 minutes	Commerce en ligne, point de vente

Disponibilité	Indisponibilité maximale (par an)	Catégories d'applications
99,99 %	52 minutes	Diffusion vidéo, charges de travail de diffusion
99,999 %	5 minutes	Transactions de distributeurs, charges de travail de télécommunications

Mesurez la disponibilité en fonction des demandes. Pour votre service, il peut être plus facile de compter les demandes ayant réussi ou échoué au lieu du "temps disponible pour utilisation". Dans ce cas, le calcul suivant peut être utilisé :

$$Disponibilité = \frac{Réponses\ fructueuses}{Demandes\ valides}$$

Cette mesure porte souvent sur des périodes d'une minute ou de cinq minutes. Un pourcentage de disponibilité mensuelle (mesure de la disponibilité sur la base du temps) peut être calculé à partir de la moyenne de ces périodes. Si aucune demande n'est reçue au cours d'une période donnée, elle est comptée comme étant disponible à 100 % pour cette période.

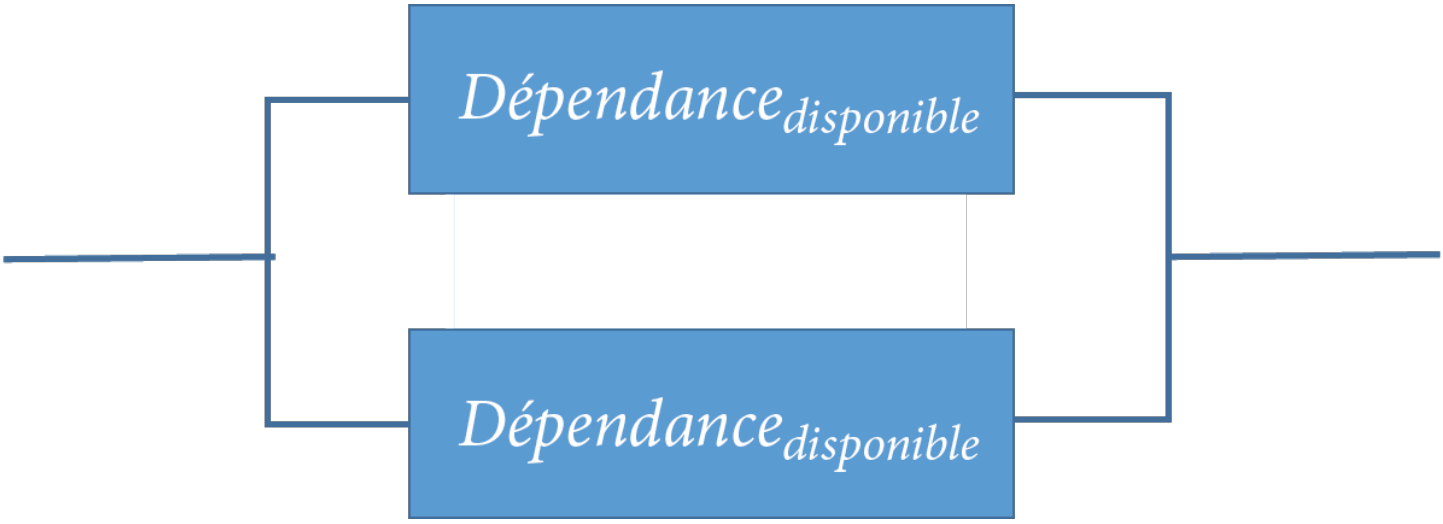
Calcul de disponibilité avec des dépendances strictes. De nombreux systèmes ont des dépendances strictes avec d'autres systèmes. Dans ce cas, une interruption dans un système dépendant se traduit directement par une interruption du système appelant. Cette notion s'oppose à celle de dépendance souple, où une défaillance du système dépendant est compensée dans l'application. Quand de telles dépendances strictes se produisent, la disponibilité du système appelant est le produit de la disponibilité des systèmes dépendants. Par exemple, si un système conçu pour offrir une disponibilité de 99,99 % a une dépendance stricte avec deux autres systèmes indépendants, qui sont chacun conçus pour offrir une disponibilité de 99,99 %, la charge de travail peut théoriquement atteindre 99,97 % de disponibilité :

$Dispo_{appelant} \times Dispo_{dep1} \times Dispo_{dep2} = Dispo_{scalable}$

99,99 % × 99,99 % × 99,99 % = 99,97 %

Il est donc important de comprendre vos dépendances et leurs objectifs de conception de disponibilité dans votre propre calcul de disponibilité.

Calcul de disponibilité avec des composants redondants. Lorsqu'un système implique l'utilisation de composants redondants et indépendants (par exemple, des ressources redondantes dans des zones de disponibilité redondantes), la disponibilité théorique est calculée à hauteur de 100 %, moins le produit des taux de défaillance des composants. Par exemple, si un système utilise deux composants indépendants, chacun avec une disponibilité de 99,9 %, la disponibilité effective de cette dépendance est > 99,9999 % :



$Dispo_{économique} = Dispo_{MAX} - ((100<\%-Dépendance\ de\ la\ disponibilité_{dépendance})\times(100<\%-Dépendance\ de\ la\ disponibilité_{dépendance}))$

99,9999 % = 100 % − (0,1 %×0,1 %)

Calcul du raccourci : si les disponibilités de tous les composants de votre calcul contiennent uniquement le chiffre neuf, vous pouvez additionner le nombre de neuf pour obtenir votre réponse. Dans l'exemple ci-dessus, deux composants redondants et indépendants avec trois neuf disponibles donnent six neuf.

Calcul de la disponibilité d'une dépendance. Certaines dépendances fournissent des informations sur leur disponibilité, y compris les objectifs de conception de disponibilité pour de nombreux services AWS. Si cette information n'est pas disponible (par exemple, pour un composant où le fabricant ne publie pas les données de disponibilité), un moyen d'estimation consiste à déterminer le temps moyen entre deux pannes (MTBF) et le temps moyen de récupération (MTTR). Une estimation de disponibilité peut être établie par la formule suivante :

$$Dispo_{EST} = \frac{MTBF}{MTBF + MTTR}$$

Par exemple, si le MTBF est de 150 jours et que le MTTR est de 1 heure, l'estimation de disponibilité est de 99,97 %.

Pour obtenir plus de détails, consultez la page Disponibilité et plus encore : comprendre et améliorer la résilience des systèmes distribués sur AWS, qui peut vous aider à calculer votre disponibilité.

Coûts liés à la disponibilité. La conception d'applications pour de plus hauts niveaux de disponibilité est généralement synonyme de coûts accrus. Par conséquent, il est nécessaire d'identifier les vrais besoins de disponibilité avant de démarrer la conception de votre application. Un haut niveau de disponibilité impose des exigences plus strictes pour les tests et la validation dans des scénarios de défaillance exhaustifs. L'automatisation est nécessaire pour la récupération à partir de toutes sortes de défaillances, et tous les aspects des opérations système doivent être conçus et testés selon les mêmes normes. Par exemple, l'ajout ou la suppression de capacité, le déploiement ou la restauration des mises à jour logicielles ou des modifications de configuration ou la migration des données système doivent être réalisées en fonction de l'objectif de disponibilité souhaité. Outre le coût de développement du logiciel, les très hauts niveaux de disponibilité sont un frein à l'innovation, car ils nécessitent de déployer beaucoup plus lentement les systèmes. Il est donc recommandé d'être minutieux dans l'application des normes et dans la sélection d'un objectif de disponibilité adapté pour tout le cycle de vie de l'exploitation du système.

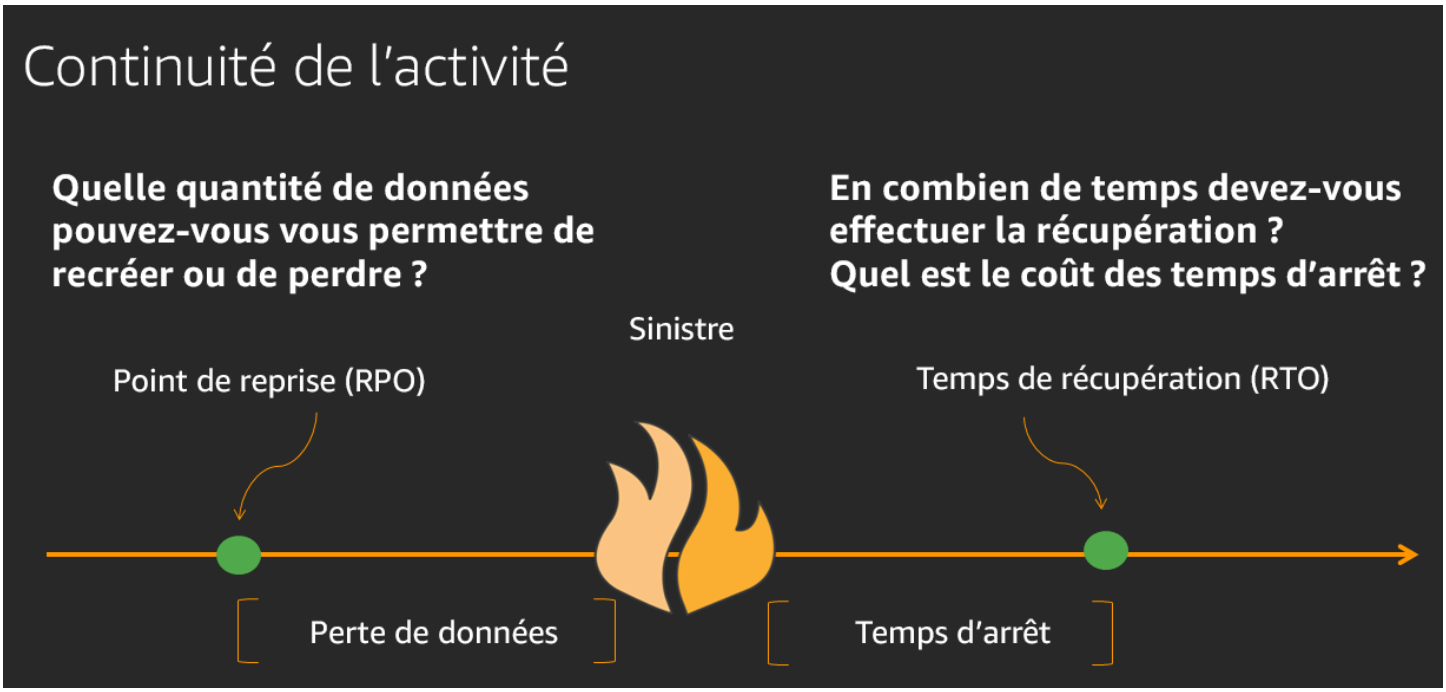
La sélection de dépendances est un autre facteur d'augmentation des coûts pour les systèmes opérant avec des objectifs de conception de disponibilité élevés. Avec ces objectifs plus élevés, l'ensemble de logiciels ou services qui peuvent être choisis en tant que dépendances diminue en fonction des services qui ont bénéficié des investissements importants décrits précédemment. Quand l'objectif de conception de disponibilité augmente, les services multifonctions (par exemple, les bases de données relationnelles) sont moins nombreux par rapport aux services plus spécialisés. Ces derniers sont en effet plus faciles à évaluer, tester et automatiser, et ils sont moins susceptibles de présenter des interactions imprévues avec des fonctionnalités incluses, mais non utilisées.

Objectifs de de reprise après sinistre

Outre les objectifs de disponibilité, votre stratégie de résilience doit également inclure des objectifs de reprise après sinistre (DR) basés sur des stratégies de récupération de votre charge de travail en cas de sinistre. La reprise après sinistre se concentre sur des objectifs de reprise ponctuels en réponse à des catastrophes naturelles, des pannes techniques à grande échelle ou des menaces humaines telles qu'une attaque ou une erreur. Elle diffère de la disponibilité qui, elle, mesure la résilience sur une période de temps en réponse aux pannes de composants, aux pics de charge ou aux bogues logiciels.

Objectif de temps de récupération (RTO) Défini par l'organisation. La RTO correspond au délai maximum acceptable entre l'interruption du service et la restauration du service. Elle détermine ce qui est considéré comme étant un créneau de temps acceptable d'indisponibilité du service.

Objectif de point de reprise (RPO) défini par l'organisation. Le RPO correspond au temps maximal acceptable depuis le dernier point de reprise des données. Il détermine ce qui est considéré comme étant une perte de données acceptable entre le dernier point de reprise et l'interruption du service.



Relation entre le RPO (objectif de point de reprise), la RTO (durée maximale d'interruption admissible) et l'événement de sinistre.

La RTO est similaire au MTTR (temps moyen de reprise) en ce que ces deux valeurs mesurent le délai entre le début d'une panne et la reprise de la charge de travail. Cependant, le MTTR est une valeur moyenne obtenue à partir de plusieurs événements ayant un impact sur la disponibilité au

cours d'une période donnée, alors que la RTO est une cible, ou une valeur maximale autorisée, pour un seul événement ayant un impact sur la disponibilité.

Compréhension des besoins en disponibilité

Il est fréquent de considérer initialement la disponibilité d'une application comme un objectif unique à atteindre pour l'application dans son ensemble. Toutefois, en y regardant de plus près, on constate souvent que certains aspects d'une application ou d'un service présentent différentes exigences en termes de disponibilité. Par exemple, certains systèmes peuvent prioriser la possibilité de recevoir et de stocker de nouvelles données, au détriment de la récupération de données existantes. D'autres systèmes vont privilégier les opérations en temps réel sur les opérations qui modifient la configuration ou l'environnement d'un système. Les services peuvent avoir des exigences de disponibilité très élevées à certains moments de la journée, mais tolérer des périodes de perturbation beaucoup plus longues le reste du temps. Voici quelques-unes des manières dont vous pouvez diviser une même application en différents éléments constitutifs, afin d'évaluer les exigences de disponibilité pour chaque partie. Cette façon de procéder a pour avantage de permettre de concentrer vos efforts (et dépenses) sur la disponibilité en fonction de besoins spécifiques, plutôt que de concevoir l'ensemble du système sur la base de l'exigence la plus stricte.

Recommandations
Évaluez de manière critique les aspects uniques de vos applications et, le cas échéant, différenciez les objectifs de conception de la disponibilité et de la reprise après sinistre pour refléter les besoins de votre entreprise.

Chez AWS, nous divisons souvent les services en deux, avec un « plan de données » et un « plan de contrôle ». Le plan de données vise à fournir un service en temps réel, tandis que les plans de contrôle servent à configurer l'environnement. Par exemple, les instances Amazon EC2, les bases de données Amazon RDS et les opérations de lecture/écriture sur les tables Amazon DynamoDB sont autant d'opérations qui relèvent du plan de données. En revanche, le lancement de nouvelles instances EC2 ou bases de données RDS, ou l'ajout ou la modification des métadonnées de table dans DynamoDB, sont considérés comme des opérations de plan de contrôle. Tandis que de hauts niveaux de disponibilité sont importants pour l'ensemble de ces fonctionnalités, les plans de données ont généralement des objectifs de conception de disponibilité plus élevés que les plans de contrôle. Par conséquent, les charges de travail avec des exigences de haute disponibilité doivent éviter la dépendance d'exécution vis-à-vis des opérations du plan de contrôle.

La plupart des clients AWS adoptent une approche similaire pour évaluer leurs applications avec un esprit critique et identifier les sous-composants avec différents besoins de disponibilité. Les objectifs de conception de disponibilité sont ensuite adaptés aux différents aspects et les efforts de travail appropriés sont mis en œuvre pour concevoir le système. AWS a accumulé une expérience importante dans l'ingénierie d'applications avec un éventail d'objectifs de conception de disponibilité, y compris des services avec une disponibilité de 99,999 % ou plus. Les architectes de solution AWS peuvent vous aider à concevoir de manière appropriée en fonction de vos objectifs de disponibilité. Impliquer AWS de manière précoce dans votre processus de conception nous permet de vous aider à atteindre vos objectifs de disponibilité. La planification pour la disponibilité ne se fait pas uniquement avant le lancement de votre charge de travail. Elle s’effectue également en continu de manière à affiner votre conception à mesure que vous accumulez de l'expérience opérationnelle, tirez des enseignements des événements réels et êtes confronté à différents types de défaillances. Vous pouvez ensuite appliquer l'effort de travail approprié pour améliorer votre mise en œuvre.

Les besoins en disponibilité requis pour une charge de travail doivent être alignés sur les besoins et la criticité de l'entreprise. En commençant par définir un cadre de criticité commerciale avec une RTO, un RPO et une disponibilité spécifiques, vous pouvez évaluer chaque charge de travail. Une telle approche exige que les personnes impliquées dans la mise en œuvre de la charge de travail connaissent le cadre et l'impact de leur charge de travail sur les besoins de l'entreprise.

Fondations

Les exigences de base sont celles dont le champ d'application s'étend au-delà d'une seule charge de travail ou d'un seul projet. Avant de concevoir l'architecture d'un système, les exigences de base qui influent sur la fiabilité doivent être mises en place. Par exemple, vous devez avoir une bande passante réseau suffisante pour votre centre de données.

Dans un environnement local, ces exigences peuvent entraîner de longs délais d'attente en raison des dépendances et, par conséquent, doivent être intégrées lors de la planification initiale. Toutefois, avec AWS, la plupart de ces exigences en matière de fondation sont déjà intégrées ou peuvent être satisfaites en fonction des besoins. Le cloud est conçu pour être presque illimité. Il est donc de la responsabilité d'AWS de satisfaire l'exigence d'une capacité suffisante de mise en réseau et de calcul, ce qui vous laisse la liberté de modifier la taille des ressources et les allocations à la demande.

Les sections suivantes présentent les bonnes pratiques qui se concentrent sur ces considérations relatives à la fiabilité.

Rubriques

Gestion des quotas et contraintes de service

Pour les architectures de charge de travail basées sur le cloud, il existe des quotas de service (aussi appelés Service Limits). Le rôle de ces quotas est d'empêcher la mise en service accidentelle de plus de ressources que nécessaire et de limiter les taux de demandes sur les opérations d'API afin de protéger les services contre les abus. Il existe également des contraintes de ressource. Par exemple, la vitesse à laquelle vous pouvez transmettre des bits sur un câble de fibre optique, ou la quantité de stockage sur un disque physique.

Si vous utilisez des applications AWS Marketplace, vous devez en comprendre les limitations. De la même manière, si vous utilisez des services web tiers ou des solutions logicielles en tant que service, vous devez être conscient de leurs limites.

Bonnes pratiques

- REL01-BP02 Gérer les quotas de service entre les comptes et les régions
- REL01-BP03 Tenir compte des quotas et des contraintes de service fixes dans l'architecture
- REL01-BP04 Surveiller et gérer les quotas
- REL01-BP05 Automatiser la gestion des quotas
- REL01-BP06 Garantir un écart suffisant entre les quotas actuels et l'utilisation maximale pour permettre le basculement

REL01-BP01 Connaissance des quotas de service et des contraintes

Connaissez vos quotas par défaut et gérez vos demandes d'augmentation de quota pour votre architecture de charge de travail. Connaissez également les contraintes de ressources, comme le disque ou le réseau, qui sont susceptibles d'avoir un impact.

Résultat souhaité : les clients peuvent prévenir la dégradation ou l'interruption des services dans leurs Comptes AWS en mettant en œuvre des directives appropriées pour la surveillance des métriques clés, les vérifications de l'infrastructure et l'automatisation des étapes de remédiation pour vérifier que les quotas des services et les contraintes ne sont pas atteints, ce qui pourrait entraîner une dégradation ou une interruption des services.

Anti-modèles courants :

- Déployer une charge de travail sans comprendre les quotas matériels ou logiciels et leurs limites pour les services utilisés.
- Déployer une charge de travail de remplacement sans analyser ni reconfigurer les quotas nécessaires ou contacter d'abord l'assistance.
- Supposer que les services cloud sont sans limite et que les services peuvent être utilisés sans prendre en compte les taux, les limites, les nombres et les quantités.
- Supposer que les quotas augmenteront automatiquement.
- Ne pas connaître le processus et la chronologie des demandes de quotas.
- Supposer que le quota du service cloud pas défaut est le même pour chaque service par rapport à d'autres régions.
- Supposer que les contraintes de service peuvent être enfreintes et que les systèmes se mettront automatiquement à l'échelle ou augmenteront la limite au-delà des contraintes de la ressource.
- Ne pas tester l'application sur des pics de trafic pour tester la résistance de l'utilisation de ces ressources.

- Provisionner les ressources sans analyser la taille de ressource nécessaire.
- Surprovisionner la capacité en choisissant des types de ressources largement supérieures aux besoins réels ou aux pics attendus.
- Ne pas évaluer les exigences de capacité pour les nouveaux niveaux de trafic avant un nouvel événement client ou le déploiement d'une nouvelle technologie.

Avantages liés au respect de cette bonne pratique : la surveillance et la gestion automatisée des quotas de service et des contraintes de ressources peuvent proactivement réduire les échecs. Les changements dans les modèles de trafic d'un service client peuvent entraînement une interruption ou une dégradation si les bonnes pratiques ne sont pas suivies. En surveillante et en gérant ces valeurs sur toutes les régions et tous les comptes, les applications peuvent bénéficier d'une meilleure résilience lors d'événements indésirables ou imprévus.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

Service Quotas est un service AWS qui vous aide à gérer vos quotas pour plus de 250 services AWS depuis un seul et même emplacement. En plus de rechercher les valeurs de quota, vous pouvez également demander et suivre les augmentations de quota à partir de la console Service Quotas ou via le kit SDK AWS. AWS Trusted Advisor propose un contrôle des quotas de service qui affiche votre utilisation et les quotas de différents aspects de certains services. Les quotas de service par défaut par service figurent également dans la documentation AWS de chaque service (par exemple, consultez Quotas Amazon VPC).

Certaines limites de services, comme les limites de taux sur les API limitées sont définies dans Amazon API Gateway en configurant un plan d'utilisation. Certaines limites définies en tant que configuration sur leurs services respectifs incluent les IOPS provisionnés, le stockage Amazon RDS alloué et les allocations de volume Amazon EBS. Amazon Elastic Compute Cloud dispose de son propre tableau de bord des limites de service qui peut vous aider à gérer votre instance, Amazon Elastic Block Store et les limites d'adresses IP Elastic. Si vous possédez un cas d'utilisation pour lequel les quotas de service affectent les performances de votre application sans être ajustables à vos besoins, contactez AWS Support pour déterminer si des mesures d'atténuation peuvent être implémentées.

Les quotas de service peuvent être spécifiques à une région ou mondiaux par nature. Un service AWS qui atteint son quota ne se comportera pas comme lors d'une utilisation normale et peut entraîner une interruption ou une dégradation du service. Par exemple, un quota de service limité le

nombre de DL Amazon EC2 qui peuvent être utilisés dans une région et cette limite peut être atteinte lors d'un événement de mise à l'échelle du trafic avec des groupes Auto Scaling (ASG).

L'utilisation des quotas de service pour chaque compte doit être évaluée régulièrement pour déterminer quelles seraient les limites de service appropriées pour ce compte. Ces quotas de service existent en tant que barrières de protection opérationnelles pour empêcher le provisionnement accidentel de plus de ressources que nécessaire. Ils servent également à limiter les taux de requêtes sur les opérations d'API pour protéger les services des abus.

Les contraintes de service sont différentes des quotas de service. Les contraintes de service représentent les limites d'une ressource spécifique, telles que définies par ce type de ressource. Il peut s'agir de la capacité de stockage (par exemple, gp2 a une limite de 1 Go à 16 To) ou du débit du disque (10 000 IOPS). Il est essentiel qu'une contrainte d'un type de ressource soit optimisée et constamment évaluée par rapport à une utilisation qui pourrait atteindre ses limites. Si une contrainte est atteinte de manière inattendue, les applications ou les services du compte peuvent être dégradés ou interrompus.

S'il existe un cas d'utilisation pour lequel les quotas de service affectent les performances d'une application sans être ajustables à vos besoins, contactez AWS Support pour déterminer si des améliorations sont possibles. Pour plus d'informations sur l'ajustement des quotas fixes, consultez REL01-BP03 Tenir compte des quotas et des contraintes de service fixes dans l'architecture.

Il existe un grand nombre de services et d'outils AWS pour vous aider à surveiller et gérer Service Quotas. Les services et les outils doivent être exploités pour vérifier automatiquement ou manuellement les niveaux de quotas.

- AWS Trusted Advisor propose un contrôle des quotas de service qui affiche votre utilisation et les quotas de différents aspects de certains services. Il peut aider à identifier des services proches du quota.
- AWS Management Console fournit des méthodes permettant d'afficher les valeurs des quotas de service, de les gérer, de demander de nouveaux quotas, de surveiller le statut des demandes de quotas et d'afficher l'historique des quotas.
- AWS CLI et CDK offrent des méthodes par programmation pour gérer et surveiller automatiquement les niveaux et l'utilisation des quotas de service.

Étapes d'implémentation

Pour Service Quotas :

- Vérifier AWS Service Quotas.
- Pour connaître vos quotas de service existant, déterminez les services (comme IAM Access Analyzer) utilisés. Il existe environ 250 services AWS contrôlés par des quotas de service. Ensuite, déterminez le nom du quota de service spécifique qui pourrait être utilisé au sein de chaque compte et région. Il existe environ 3 000 noms de quotas de service par région.
- Augmentez cette analyse des quotas avec AWS Config pour trouver toutes les ressources AWS utilisées dans vos Comptes AWS.
- Utilisez les données AWS CloudFormation pour déterminer vos ressources AWS utilisées. Examinez les ressources créées dans AWS Management Console ou via la commande `list-stack-resources` de l'AWS CLI. Vous pouvez également voir les ressources configurées pour être déployées directement dans le modèle.
- Déterminez tous les services indispensables à votre charge de travail en prenant en compte le code de déploiement.
- Identifiez les quotas de service pertinents. Utilisez les informations accessibles par programmation par le biais de Trusted Advisor et Service Quotas.
- Établissez une méthode de surveillance automatisée (consultez REL01-BP02 Gérer les quotas de service entre les comptes et les régions et REL01-BP04 Surveiller et gérer les quotas) pour vous alerter et vous informer si des quotas de service sont sur le point d'atteindre ou ont atteint leur limite.
- Établissez une méthode automatisée et par programmation pour vérifier si un quota de service a été modifié dans une région mais pas dans d'autres régions au sein du même compte (consultez REL01-BP02 Gérer les quotas de service entre les comptes et les régions et REL01-BP04 Surveiller et gérer les quotas).
- Automatisez l'analyse des journaux et des métriques de l'application pour déterminer s'il existe des erreurs de quotas ou de contraintes de service. Si de telles erreurs existent, envoyez des alertes au système de surveillance.
- Établissez des procédures d'ingénierie pour calculer le changement de quota nécessaire (consultez REL01-BP05 Automatiser la gestion des quotas) une fois qu'il a été identifié que des quotas plus importants sont nécessaires pour des services spécifiques.
- Créez une flux de travail de provisionnement et d'approbation pour demander des modifications des quotas de service. Cela doit inclure un flux de travail d'exception en cas de refus d'une demande ou d'une approbation partielle.

- Créez une méthode d'ingénierie pour vérifier les quotas de service avant le provisionnement et utilisez de nouveaux services AWS avant le déploiement dans des environnements de production ou chargés (par exemple, un compte de test de charge).

Pour les contraintes de service :

- Établissez des méthodes de surveillance et des métriques pour alerter quand les ressources sont proches de leurs contraintes. Tirez profit de CloudWatch tel que nécessaire pour la surveillance des métriques ou des journaux.
- Établissez des seuils d'alertes pour chaque ressource ayant une contrainte importante pour l'application ou le système.
- Créez des procédures de gestion des flux de travail et de l'infrastructure pour changer le type de ressource si la contrainte est proche. Ce flux de travail doit inclure le test de charge comme une bonne pratique pour vérifier que ce nouveau type est le bon type de ressource avec les nouvelles contraintes.
- Procédez à la migration des ressources identifiées vers le nouveau type de ressource recommandé avec les procédures et les processus existants.

Ressources

Bonnes pratiques associées :

- REL01-BP02 Gérer les quotas de service entre les comptes et les régions
- REL01-BP03 Tenir compte des quotas et des contraintes de service fixes dans l'architecture
- REL01-BP04 Surveiller et gérer les quotas
- REL01-BP05 Automatiser la gestion des quotas
- REL01-BP06 Garantir un écart suffisant entre les quotas actuels et l'utilisation maximale pour permettre le basculement
- REL03-BP01 Choisir comment segmenter votre charge de travail
- REL10-BP01 Déployer la charge de travail sur plusieurs emplacements
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances
- REL11-BP03 Automatiser la réparation sur toutes les couches
- REL12-BP05 Tester la résilience à l'aide de l'ingénierie du chaos

Documents connexes :

- Pilier Fiabilité du cadre AWS Well-Architected : Disponibilité
- AWS Service Quotas (anciennement appelés limites de service)
- Vérifications des bonnes pratiques AWS Trusted Advisor (voir la section Limites de service)
- AWS limit monitor on AWS answers (Surveillance de limites AWS sur les réponses AWS)
- Amazon EC2 Service Limits (Limites de service EC2)
- Qu'est-ce qu'Service Quotas ?
- How to Request quota increase (Comment demander une augmentation du quota)
- Service endpoints and quotas (Points de terminaison et quotas de service)
- Guide de l'utilisateur Service Quotas
- Quota Monitor for AWS (Surveillance de quotas pour AWS)
- AWS Fault Isolation Boundaries (Limites d'isolement des pannes AWS)
- Availability with redundancy (Disponibilité avec redondance)
- AWS pour les données
- Qu'est-ce que l'intégration continue ?
- Qu'est-ce que la livraison continue ?
- Partenaire APN : partenaires facilitant la gestion de la configuration
- Managing the account lifecycle in account-per-tenant SaaS environments on AWS (Gestion du cycle de vie des comptes dans les environnements SaaS de type compte par locataire sur AWS)
- Gestion et surveillance de la limitation des API dans vos charges de travail
- View AWS Trusted Advisor recommendations at scale with AWS Organizations (Examiner les recommandations AWS Trusted Advisor à grande échelle avec AWS Organizations)
- Automating Service Limit Increases and Enterprise Support with AWS Control Tower (Automatisation de l'augmentation des limites de service et Enterprise Support avec AWS Control Tower)

Vidéos connexes :

- AWS Live re:Inforce 2019 - Service Quotas
- View and Manage Quotas for AWS Services Using Service Quotas (Examiner et gérer les quotas pour les services AWS avec les quotas de service)
- AWS IAM Quotas Demo (Démonstration sur les quotas IAM d'AWS)

Outils associés :

- Amazon CodeGuru Reviewer
- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- Amazon DevOps Guru
- AWS Config
- AWS Trusted Advisor
- AWS CDK
- AWS Systems Manager
- AWS Marketplace

REL01-BP02 Gérer les quotas de service entre les comptes et les régions

Si vous utilisez plusieurs comptes ou régions, demandez les quotas appropriés dans tous les environnements où vos charges de travail de production s'exécutent.

Résultat souhaité : Les services et les applications ne doivent pas être affectés par un épuisement de quota de service pour les configurations qui englobent les comptes ou les régions ou dont les conceptions de résilience s'appuient sur le basculement de zone, de région ou de compte.

Anti-modèles courants :

- Laisser l'utilisation des ressources dans une région d'isolement se développer sans aucun mécanisme pour maintenir de la capacité dans les autres zones.
- Définir manuellement tous les quotas de manière indépendante dans les régions d'isolement.
- Ne pas prendre en considération l'effet des architectures de résilience (par ex., actives ou passives) dans les futurs besoins de quotas alors qu'une dégradation est observée dans la région non principale.
- Ne pas évaluer les quotas régulièrement et ne pas apporter les changements qui s'imposent dans chaque région et chaque compte où la charge de travail s'exécute.
- Ne pas tirer parti des modèles de demande de quota pour demander des augmentations dans plusieurs régions et comptes.

- Ne pas mettre à jour les quotas de service pensant à tort que l'augmentation de quotas a des répercussions sur les coûts comme les demandes de réservation de capacité de calcul.

Avantages liés au respect de cette bonne pratique : Possibilité de vérifier que vous êtes en mesure de gérer votre charge actuelle dans les régions ou comptes secondaires au cas où les services régionaux viendraient à être indisponibles. Cela peut contribuer à limiter le nombre d'erreurs ou les niveaux de dégradations observés lors d'une perte de région.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

Les quotas de service sont suivis par compte. Sauf indication contraire, chaque quota est propre à une Région AWS. En plus des environnements de production, tâchez également de gérer les quotas dans tous les autres environnements applicables de façon à ne pas entraver les tests et le développement. Pour maintenir un haut niveau de résilience, il convient d'évaluer constamment les quotas de service (que ce soit de façon automatisée ou manuelle).

Compte tenu du plus grand nombre de charges de travail englobant les régions du fait de l'implémentation de conceptions utilisant les approches Actif/Actif, Actif/Passif – Chaud, Actif/Passif – Froid et Actif/Passif – Veilleuse, il est essentiel de comprendre tous les niveaux de quota de région et de compte. Les modèles de trafic passés ne permettent pas toujours de déterminer correctement si le quota de service est bien défini.

Tout aussi important, la longueur limite des noms de quota de service n'est pas toujours identique d'une région à l'autre. Ainsi, cette valeur peut être égale à cinq dans une région et à dix dans une autre. La gestion de ces quotas doit englober tous les services, comptes et régions identiques pour offrir une résilience cohérente dans des conditions de charge.

Rapprochez toutes les différences de quota de service entre les différentes régions (région active ou région passive) et créez des processus permettant de rapprocher constamment ces différences. Les plans de test de basculements de régions passives sont rarement mis à l'échelle pour atteindre une capacité active de pointe, ce qui signifie que les exercices de simulation (« game day ») et les exercices de table (« table top ») ne permettent pas nécessairement d'identifier les différences dans les quotas de service entre les régions et donc de maintenir les limites adéquates.

La dérive de quota de service, condition où les limites de quota de service pour un quota nommé spécifique changent dans une région mais pas dans toutes, est très importante pour le suivi et

l'évaluation. Il doit être envisagé de changer le quota dans les régions qui présentent du trafic ou qui pourraient potentiellement en véhiculer.

- Sélectionnez les comptes et les régions appropriés en fonction de vos exigences de service, de latence, de réglementation et de reprise après sinistre (DR).
- Identifiez les quotas de services dans l'ensemble des comptes, régions et zones de disponibilité appropriés. Les limites s'appliquent au compte et à la région. Ces valeurs doivent être comparées pour repérer les différences.

Étapes d'implémentation

- Examinez les valeurs Service Quotas susceptibles d'avoir transgressé le niveau d'utilisation à risque. AWS Trusted Advisor propose des alertes pour les violations de seuil de 80 % et 90 %.
- Examinez les valeurs de quotas de service dans les régions passives (dans une conception de type Actif/Passif). Vérifiez que la charge s'exécutera correctement dans les régions secondaires en cas de défaillance dans la région principale.
- Automatisez l'évaluation pour identifier une éventuelle dérive de quota de service entre des régions d'un même compte et agissez en conséquence pour changer les limites.
- Si la structure des unités d'organisation (UO) est prise en charge, les modèles de quota de service doivent être mis à jour en fonction des changements apportés aux quotas qui doivent s'appliquer à plusieurs régions et comptes.
  - Créez un modèle et associez les régions au changement de quota.
  - Examinez tous les modèles de quota de service existants pour y apporter les changements nécessaires (région, limites et comptes).

Ressources

Bonnes pratiques associées :

- REL01-BP01 Connaissance des quotas de service et des contraintes
- REL01-BP03 Tenir compte des quotas et des contraintes de service fixes dans l'architecture
- REL01-BP04 Surveiller et gérer les quotas
- REL01-BP05 Automatiser la gestion des quotas
- REL01-BP06 Garantir un écart suffisant entre les quotas actuels et l'utilisation maximale pour permettre le basculement

- REL03-BP01 Choisir comment segmenter votre charge de travail
- REL10-BP01 Déployer la charge de travail sur plusieurs emplacements
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances
- REL11-BP03 Automatiser la réparation sur toutes les couches
- REL12-BP05 Tester la résilience à l'aide de l'ingénierie du chaos

Documents connexes :

- Pilier Fiabilité du cadre AWS Well-Architected : Disponibilité
- AWS Service Quotas (anciennement appelés limites de service)
- Vérifications des bonnes pratiques AWS Trusted Advisor (voir la section Limites de service)
- AWS limit monitor on AWS answers (Surveillance de limites AWS sur les réponses AWS)
- Amazon EC2 Service Limits (Limites de service EC2)
- Qu'est-ce qu'Service Quotas ?
- How to Request quota increase (Comment demander une augmentation du quota)
- Service endpoints and quotas (Points de terminaison et quotas de service)
- Guide de l'utilisateur Service Quotas
- Quota Monitor for AWS (Surveillance de quotas pour AWS)
- AWS Fault Isolation Boundaries (Limites d'isolement des pannes AWS)
- Availability with redundancy (Disponibilité avec redondance)
- AWS pour les données
- Qu'est-ce que l'intégration continue ?
- Qu'est-ce que la livraison continue ?
- Partenaire APN : partenaires facilitant la gestion de la configuration
- Managing the account lifecycle in account-per-tenant SaaS environments on AWS (Gestion du cycle de vie des comptes dans les environnements SaaS de type compte par locataire sur AWS)
- Gestion et surveillance de la limitation des API dans vos charges de travail
- View AWS Trusted Advisor recommendations at scale with AWS Organizations (Examiner les recommandations AWS Trusted Advisor à grande échelle avec AWS Organizations)
- Automating Service Limit Increases and Enterprise Support with AWS Control Tower (Automatisation de l'augmentation des limites de service et Enterprise Support avec AWS Control Tower)

Vidéos connexes :

- AWS Live re:Inforce 2019 - Service Quotas
- View and Manage Quotas for AWS Services Using Service Quotas (Examiner et gérer les quotas pour les services AWS avec les quotas de service)
- AWS IAM Quotas Demo (Démonstration sur les quotas IAM d'AWS)

Services associés :

- Amazon CodeGuru Reviewer
- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- Amazon DevOps Guru
- AWS Config
- AWS Trusted Advisor
- AWS CDK
- AWS Systems Manager
- AWS Marketplace

REL01-BP03 Tenir compte des quotas et des contraintes de service fixes dans l'architecture

Ayez conscience des quotas de service non modifiables, des contraintes de service et des limites de ressources physiques. Concevez des architectures pour les applications et les services afin d'éviter que ces limites n'aient un impact sur la fiabilité.

Par exemple, la bande passante du réseau, la taille de la charge utile des appels de fonctions sans serveur, le taux d'accélération d'une passerelle API et les connexions simultanées d'utilisateurs à une base de données.

Résultat souhaité : l'application ou le service fonctionne comme prévu dans des conditions de trafic normal et élevé. Ils ont été conçus pour fonctionner dans les limites des contraintes fixes ou des quotas de service de cette ressource.

Anti-modèles courants :

- Choix d'une conception qui utilise une ressource d'un service, sans savoir qu'il existe des contraintes de conception qui entraîneront l'échec de cette conception au fil des mises à l'échelle.
- Effectuer une évaluation comparative qui n'est pas réaliste et qui atteindra les quotas fixés par le service pendant les tests. Par exemple, l'exécution de tests à une limite de débordement mais pendant une durée prolongée.
- Le choix d'une conception qui ne peut pas évoluer ou être modifiée si des quotas de service fixes doivent être dépassés. Par exemple, une taille de charge utile SQS de 256 KB.
- L'observabilité n'a pas été conçue et mise en œuvre pour surveiller et alerter sur les seuils des quotas de service qui pourraient être compromis lors d'événements à fort trafic

Avantages liés au respect de cette bonne pratique : vérifier que l'application fonctionnera sous tous les niveaux de charge des services projetés sans perturbation ni dégradation.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

Contrairement aux quotas de services souples ou aux ressources qui peuvent être remplacées par des unités de plus grande capacité, les quotas fixes des services AWS ne peuvent pas être modifiés. Cela signifie que tous ces types de services AWS doivent être évalués en fonction des limites potentielles de capacité matérielle lorsqu'ils sont utilisés dans la conception d'une application.

Les limites strictes sont affichées dans la console Service Quotas. Si les colonnes affichent la valeur `ADJUSTABLE = No`, alors le service comporte une limite stricte. Des limites strictes sont également indiquées dans les pages de configuration de certaines ressources. Par exemple, Lambda possède des limites strictes spécifiques qui ne peuvent pas être ajustées.

À titre d'exemple, lors de la conception d'une application python destinée à être exécutée dans une fonction Lambda, l'application doit être évaluée pour déterminer si Lambda risque de s'exécuter pendant plus de 15 minutes. Si le code peut fonctionner au-delà de cette limite de quota de service, il faut envisager d'autres technologies ou conceptions. Si cette limite est atteinte après le déploiement de la production, l'application subira une dégradation et des perturbations jusqu'à ce qu'il soit possible d'y remédier. Contrairement aux quotas souples, il n'existe aucune méthode permettant de passer à ces limites, même en cas d'événements d'urgence de gravité 1.

Une fois que l'application a été déployée dans un environnement de test, il convient d'utiliser des stratégies pour déterminer si des limites strictes peuvent être atteintes. Les tests de résistance, les tests de charge et les tests de chaos doivent faire partie du plan de test d'introduction.

Étapes d'implémentation

- Examinez la liste complète des services AWS qui pourraient être utilisés dans la phase de conception de l'application.
- Examinez les limites de quota logiciel et de quota matériel pour tous ces services. Toutes les limites ne sont pas affichées dans la console Service Quotas. Certains services détaillent ces limites à d'autres endroits.
- Lors de la conception de votre application, examinez les facteurs opérationnels et technologiques de votre charge de travail, tels que les résultats opérationnels, le cas d'utilisation, les systèmes dépendants, les objectifs de disponibilité et les objets de reprise après sinistre. Laissez vos facteurs commerciaux et technologiques guider le processus d'identification du système distribué qui convient à votre charge de travail.
- Analysez la charge de service dans les régions et les comptes. De nombreuses limites strictes se basent sur la région pour les services. Cependant, certaines limites sont basées sur le compte.
- Analysez les architectures de résilience pour l'utilisation des ressources lors d'une panne de zone et d'une panne régionale. Dans la progression des conceptions multirégionales utilisant des approches actives/actives, actives/passives – à chaud, actives/passives – à froid, et actives/passives – environnement de veille, ces cas de panne entraîneront une utilisation plus importante. Cela crée un cas d'utilisation potentiel pour atteindre des limites strictes.

Ressources

Bonnes pratiques associées :

- REL01-BP01 Connaissance des quotas de service et des contraintes
- REL01-BP02 Gérer les quotas de service entre les comptes et les régions
- REL01-BP04 Surveiller et gérer les quotas
- REL01-BP05 Automatiser la gestion des quotas
- REL01-BP06 Garantir un écart suffisant entre les quotas actuels et l'utilisation maximale pour permettre le basculement
- REL03-BP01 Choisir comment segmenter votre charge de travail

- REL10-BP01 Déployer la charge de travail sur plusieurs emplacements
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances
- REL11-BP03 Automatiser la réparation sur toutes les couches
- REL12-BP05 Tester la résilience à l'aide de l'ingénierie du chaos

Documents connexes :

- Pilier Fiabilité du cadre AWS Well-Architected : Disponibilité
- AWS Service Quotas (anciennement appelés limites de service)
- Vérifications des bonnes pratiques AWS Trusted Advisor (voir la section Limites de service)
- AWS limit monitor on AWS answers (Surveillance de limites AWS sur les réponses AWS)
- Amazon EC2 Service Limits (Limites de service EC2)
- Qu'est-ce qu'Service Quotas ?
- How to Request quota increase (Comment demander une augmentation du quota)
- Service endpoints and quotas (Points de terminaison et quotas de service)
- Guide de l'utilisateur Service Quotas
- Quota Monitor for AWS (Surveillance de quotas pour AWS)
- AWS Fault Isolation Boundaries (Limites d'isolement des pannes AWS)
- Availability with redundancy (Disponibilité avec redondance)
- AWS pour les données
- Qu'est-ce que l'intégration continue ?
- Qu'est-ce que la livraison continue ?
- Partenaire APN : partenaires facilitant la gestion de la configuration
- Managing the account lifecycle in account-per-tenant SaaS environments on AWS (Gestion du cycle de vie des comptes dans les environnements SaaS de type compte par locataire sur AWS)
- Gestion et surveillance de la limitation des API dans vos charges de travail
- View AWS Trusted Advisor recommendations at scale with AWS Organizations (Examiner les recommandations AWS Trusted Advisor à grande échelle avec AWS Organizations)
- Automating Service Limit Increases and Enterprise Support with AWS Control Tower (Automatisation de l'augmentation des limites de service et Enterprise Support avec AWS Control Tower)

- Actions, ressources et clés de condition pour les Service Quotas

Vidéos connexes :

- AWS Live re:Inforce 2019 - Service Quotas
- View and Manage Quotas for AWS Services Using Service Quotas (Examiner et gérer les quotas pour les services AWS avec les quotas de service)
- AWS IAM Quotas Demo (Démonstration sur les quotas IAM d'AWS)
- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small

Outils associés :

- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- Amazon DevOps Guru
- AWS Config
- AWS Trusted Advisor
- AWS CDK
- AWS Systems Manager
- AWS Marketplace

REL01-BP04 Surveiller et gérer les quotas

Évaluez votre utilisation potentielle et augmentez vos quotas de manière appropriée afin d'assurer une croissance planifiée de l'utilisation.

Résultat souhaité : des systèmes actifs et automatisés de gestion et de suivi ont été déployés. Ces solutions opérationnelles permettent de s'assurer que les seuils d'utilisation des quotas sont sur le point d'être atteints. Les changements de quotas demandés permettraient de remédier à ces problèmes de manière proactive.

Anti-modèles courants :

- Ne pas configurer la surveillance pour vérifier les seuils de quota de service
- Ne pas configurer la surveillance pour les limites strictes, même si ces valeurs ne peuvent pas être modifiées.
- En supposant que le délai nécessaire pour demander et obtenir un changement de quota souple soit immédiat ou de courte durée.
- Configuration d'alarmes d'approche des quotas de service, mais sans processus sur la façon de répondre à une alerte.
- Configuration d'alarmes uniquement pour les services pris en charge par les AWS Service Quotas, sans surveiller les autres services AWS.
- Ne pas prendre en compte la gestion des quotas pour les conceptions de résilience à régions multiples, comme les approches actives/actives, actives/passives – à chaud, actives/passives – à froid, et actives/passives – environnement de veille.
- Ne pas évaluer les différences de quotas entre les régions.
- Ne pas évaluer les besoins de chaque région pour une demande spécifique d'augmentation de quota.
- Ne pas utiliser les modèles pour la gestion des quotas multirégionaux.

Avantages liés au respect de cette bonne pratique : le suivi automatique des Service Quotas AWS et la surveillance de votre utilisation par rapport à ces quotas vous permettront de voir quand vous approchez de la limite du quota. Vous pouvez également utiliser ces données de surveillance pour limiter les dégradations dues à l'épuisement des quotas.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

Pour les services pris en charge, vous pouvez surveiller vos quotas en configurant différents services qui peuvent évaluer et ensuite envoyer des alertes ou des alarmes. Cela peut aider à surveiller l'utilisation et vous alerter sur l'approche des quotas. Ces alarmes peuvent être déclenchées à partir de AWS Config, de fonctions Lambda, de Amazon CloudWatch, ou de AWS Trusted Advisor. Vous pouvez également utiliser des filtres de métriques sur les journaux CloudWatch pour rechercher et extraire des modèles dans les journaux afin de déterminer si l'utilisation approche des seuils de quota.

Étapes d'implémentation

Pour la surveillance :

- Enregistrez la consommation des ressources actuelles (par exemple, les compartiments, ou les instances). Utilisez les opérations de l'API de service, telles que l'API Amazon EC2, `DescribeInstances` pour recueillir la consommation actuelle des ressources.
- Saisissez vos quotas actuels qui sont essentiels et applicables aux services utilisés :
  - AWS Service Quotas
  - AWS Trusted Advisor
  - documentation AWS
  - Pages spécifiques aux services AWS
  - AWS Command Line Interface (AWS CLI)
  - AWS Cloud Development Kit (AWS CDK)
- Utilisez AWS Service Quotas, un service AWS qui vous aide à gérer vos quotas pour plus de 250 services AWS à partir d'un seul emplacement.
- Utilisez les limites de service Trusted Advisor pour surveiller vos limites de service actuelles à différents seuils.
- Utilisez l'historique des quotas de service (console ou AWS CLI) pour vérifier les augmentations régionales.
- Comparez les changements de quotas de service dans chaque région et chaque compte pour créer une équivalence, si nécessaire.

Pour la gestion :

- Automatisé : configurez une règle AWS Config personnalisée pour analyser les quotas de service dans les régions et comparer les différences.
- Automatisé : configurez une fonction programmée Lambda pour analyser les quotas de service dans les régions et comparer les différences.
- Manuel : analysez les quotas de services par le biais de la AWS CLI, d'API ou de la console AWS pour analyser les quotas de services dans les régions et comparer les différences. Signalez les différences.
- Si des différences de quotas sont identifiées entre les régions, demandez un changement de quota, si nécessaire.

- Passez en revue le résultat de toutes les demandes.

Ressources

Bonnes pratiques associées :

- REL01-BP01 Connaissance des quotas de service et des contraintes
- REL01-BP02 Gérer les quotas de service entre les comptes et les régions
- REL01-BP03 Tenir compte des quotas et des contraintes de service fixes dans l'architecture
- REL01-BP05 Automatiser la gestion des quotas
- REL01-BP06 Garantir un écart suffisant entre les quotas actuels et l'utilisation maximale pour permettre le basculement
- REL03-BP01 Choisir comment segmenter votre charge de travail
- REL10-BP01 Déployer la charge de travail sur plusieurs emplacements
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances
- REL11-BP03 Automatiser la réparation sur toutes les couches
- REL12-BP05 Tester la résilience à l'aide de l'ingénierie du chaos

Documents connexes :

- Pilier Fiabilité du cadre AWS Well-Architected : Disponibilité
- AWS Service Quotas (anciennement appelés limites de service)
- Vérifications des bonnes pratiques AWS Trusted Advisor (voir la section Limites de service)
- AWS limit monitor on AWS answers (Surveillance de limites AWS sur les réponses AWS)
- Amazon EC2 Service Limits (Limites de service EC2)
- Qu'est-ce qu'Service Quotas ?
- How to Request quota increase (Comment demander une augmentation du quota)
- Service endpoints and quotas (Points de terminaison et quotas de service)
- Guide de l'utilisateur Service Quotas
- Quota Monitor for AWS (Surveillance de quotas pour AWS)
- AWS Fault Isolation Boundaries (Limites d'isolement des pannes AWS)
- Availability with redundancy (Disponibilité avec redondance)
- AWS pour les données

- Qu'est-ce que l'intégration continue ?
- Qu'est-ce que la livraison continue ?
- Partenaire APN : partenaires facilitant la gestion de la configuration
- Managing the account lifecycle in account-per-tenant SaaS environments on AWS (Gestion du cycle de vie des comptes dans les environnements SaaS de type compte par locataire sur AWS)
- Gestion et surveillance de la limitation des API dans vos charges de travail
- View AWS Trusted Advisor recommendations at scale with AWS Organizations (Examiner les recommandations AWS Trusted Advisor à grande échelle avec AWS Organizations)
- Automating Service Limit Increases and Enterprise Support with AWS Control Tower (Automatisation de l'augmentation des limites de service et Enterprise Support avec AWS Control Tower)
- Actions, ressources et clés de condition pour les Service Quotas

Vidéos connexes :

- AWS Live re:Inforce 2019 - Service Quotas
- View and Manage Quotas for AWS Services Using Service Quotas (Examiner et gérer les quotas pour les services AWS avec les quotas de service)
- AWS IAM Quotas Demo (Démonstration sur les quotas IAM d'AWS)
- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small

Outils associés :

- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- Amazon DevOps Guru
- AWS Config
- AWS Trusted Advisor
- AWS CDK
- AWS Systems Manager

- AWS Marketplace

REL01-BP05 Automatiser la gestion des quotas

Mettez en place des outils pour être informé à lorsque l'atteinte des seuils est proche. Vous pouvez automatiser les demandes d'augmentation de quota à l'aide des API AWS Service Quotas.

Si vous intégrez votre base de données de gestion de configuration (CMDB) ou votre système de tickets avec Service Quotas, vous pouvez automatiser le suivi des requêtes d'augmentation de quotas et des quotas actuels. En plus du kit SDK AWS, Service Quotas propose une automatisation avec AWS Command Line Interface (AWS CLI).

Anti-modèles courants :

- Suivi des quotas et de l'utilisation dans les feuilles de calcul.
- Exécution de rapports sur l'utilisation quotidienne, hebdomadaire ou mensuelle, puis comparaison de l'utilisation aux quotas.

Avantages liés au respect de cette bonne pratique : Le suivi automatisé des quotas de service AWS et la surveillance de votre utilisation par rapport à ce quota vous permettent de voir quand vous vous rapprochez d'un quota. Vous pouvez configurer l'automatisation pour vous aider à demander une augmentation de quota si nécessaire. Vous pouvez envisager de réduire certains quotas lorsque votre utilisation évolue dans le sens inverse pour profiter des avantages d'une réduction des risques (en cas d'informations d'identification corrompues) et des économies de coûts.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

- Configurer la surveillance automatisée : mettez en place des outils à l'aide de kits SDK pour être informé lorsque des seuils sont sur le point d'être atteints.
  - Utilisez les Service Quotas et complétez le service avec une solution automatisée de surveillance des quotas, tels qu'AWS Limit Monitor ou une offre sur AWS Marketplace.
    - Qu'est-ce que Service Quotas ?
    - Surveillance des quotas sur AWS - Solution AWS
  - Configurez des réponses déclenchées en fonction des seuils de quota, à l'aide des API Amazon SNS et AWS Service Quotas .

- Testez l'automatisation.
  - Configurez les seuils de limites.
  - Intégrez les événements de modification provenant d'AWS Config, des pipelines de déploiement, d'Amazon EventBridge ou de tiers.
  - Définissez des seuils de limites artificiellement bas pour tester les réponses.
  - Configurez des déclencheurs pour prendre les mesures appropriées en cas de notifications et contactez AWS Support, le cas échéant
  - Déclenchez manuellement les événements de modifications.
  - Organisez un jeu de rôle pour tester le processus d'augmentation des quotas.

Ressources

Documents connexes :

- Partenaire APN : partenaires facilitant la gestion de la configuration
- AWS Marketplace : produits CMDB facilitant le suivi des limites
- AWS Service Quotas (anciennement appelés Service Limits)
- Vérifications des bonnes pratiques AWS Trusted Advisor (voir la section Service Limits)
- Surveillance des quotas sur AWS - Solution AWS
- Amazon EC2 Service Limits
- Qu'est-ce que Service Quotas ?

Vidéos connexes :

- AWS Live re:Inforce 2019 - Service Quotas

REL01-BP06 Garantir un écart suffisant entre les quotas actuels et l'utilisation maximale pour permettre le basculement

En cas de panne ou d'inaccessibilité d'une ressource, celle-ci peut être comptabilisée dans un quota jusqu'à ce qu'elle soit correctement terminée. Vérifiez que vos quotas couvrent le chevauchement des ressources défaillantes ou inaccessibles et de leurs remplacements. Vous devez prendre en compte les cas d'utilisation tels que les pannes de réseau, la panne de la zone de disponibilité ou les pannes régionales lorsque vous calculez cet écart.

Résultat souhaité : les défaillances, petites ou grandes, des ressources ou de leur accessibilité peuvent être gérées par les seuils de service actuels. Les pannes de zone, les pannes de réseau, voire les pannes régionales ont été prises en compte dans la planification des ressources.

Anti-modèles courants :

- Définition de quotas de service en fonction des quotas actuels sans tenir compte des scénarios de basculement.
- Ne pas tenir compte des principes de stabilité statique lors du calcul du quota de pointe pour un service.
- Ne pas tenir compte du potentiel des ressources inaccessibles dans le calcul du quota total nécessaire pour chaque région.
- Ne pas prendre en compte les limites d'isolement des pannes de service AWS pour certains services et leurs potentiels schémas d'utilisation anormaux.

Avantages liés au respect de cette bonne pratique : lorsqu'une interruption de service a un impact sur la disponibilité des applications, le cloud vous permet de mettre en œuvre des stratégies pour atténuer ou récupérer ces événements. Ces stratégies incluent souvent la création de ressources supplémentaires pour remplacer celles qui ont échoué ou celles qui sont inaccessibles. Votre stratégie de quotas devrait tenir compte de ces conditions de basculement et ne pas ajouter de dégradations supplémentaires dues à l'épuisement des limites de service.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

Lors de l'évaluation des limites de quotas, il faut tenir compte des cas de basculement qui pourraient survenir en raison d'une certaine dégradation. Les types de basculement suivants doivent être pris en compte :

- Un VPC perturbé ou inaccessible.
- Un sous-réseau inaccessible.
- Une zone de disponibilité suffisamment dégradée pour avoir un impact sur l'accessibilité de nombreuses ressources.
- Divers itinéraires de mise en réseau ou points d'entrée et de sortie sont bloqués ou modifiés.
- Une région a été suffisamment dégradée pour avoir un impact sur l'accessibilité de nombreuses ressources.

- Il existe plusieurs ressources, mais toutes ne sont pas affectées par une panne dans une région ou une zone de disponibilité.

Des pannes comme celles énumérées ci-dessus peuvent être le déclencheur d'un événement de basculement. La décision de basculement est unique selon la situation et le client, car l'impact sur l'entreprise peut varier considérablement. Toutefois, lorsque l'on décide, sur le plan opérationnel, de basculer une application ou des services, la planification de la capacité des ressources dans l'emplacement de basculement et les quotas correspondants doivent être définis avant l'événement.

Passez en revue les quotas de service pour chaque service en tenant compte des pics supérieurs à la normale qui pourraient se produire. Ces pics peuvent être liés à des ressources qui ne peuvent être atteintes en raison de la mise en réseau ou des autorisations, mais qui sont toujours actives. Les ressources actives non résiliées seront toujours comptabilisées dans la limite du quota de service.

Étapes d'implémentation

- Vérifiez que l'écart entre votre quota de service et votre utilisation maximale est suffisant pour faire face à un basculement ou à une perte d'accessibilité.
- Identifiez les quotas de service en tenant compte de vos modèles de déploiement, de vos exigences en matière de disponibilité et de la croissance de votre consommation.
- Demandez des augmentations de quota si nécessaire Prévoyez le temps nécessaire pour l'approbation des demandes d'augmentation des quotas.
- Déterminez vos exigences de fiabilité (également connues sous le nom de « nombre de neuf »).
- Définissez vos scénarios de défaillance (par exemple, perte d'un composant, d'une zone de disponibilité ou d'une région).
- Définissez votre méthodologie de déploiement (par exemple, canary, bleu/vert, rouge/noir ou par propagation).
- Ajoutez une mémoire tampon appropriée (par exemple, 15 %) à la limite actuelle.
- Ajoutez les calculs de stabilité statique (zonale et régionale), le cas échéant.
- Anticipez la croissance de la consommation (par exemple, surveillance de vos tendances de consommation).
- Songez à l'impact de la stabilité statique pour vos charges de travail les plus critiques. Évaluez les ressources conformes à un système statiquement stable dans toutes les régions et zones de disponibilité.

- Envisagez l'utilisation de réservations de capacité à la demande pour programmer la capacité avant tout basculement. Cette stratégie peut s'avérer utile lors des calendriers d'activité les plus critiques afin de réduire les risques potentiels liés à l'obtention de la bonne quantité et du bon type de ressources lors du basculement.

Ressources

Bonnes pratiques associées :

- REL01-BP01 Connaissance des quotas de service et des contraintes
- REL01-BP02 Gérer les quotas de service entre les comptes et les régions
- REL01-BP03 Tenir compte des quotas et des contraintes de service fixes dans l'architecture
- REL01-BP04 Surveiller et gérer les quotas
- REL01-BP05 Automatiser la gestion des quotas
- REL03-BP01 Choisir comment segmenter votre charge de travail
- REL10-BP01 Déployer la charge de travail sur plusieurs emplacements
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances
- REL11-BP03 Automatiser la réparation sur toutes les couches
- REL12-BP05 Tester la résilience à l'aide de l'ingénierie du chaos

Documents connexes :

- Pilier Fiabilité du cadre AWS Well-Architected : Disponibilité
- AWS Service Quotas (anciennement appelés limites de service)
- Vérifications des bonnes pratiques AWS Trusted Advisor (voir la section Limites de service)
- AWS limit monitor on AWS answers (Surveillance de limites AWS sur les réponses AWS)
- Amazon EC2 Service Limits (Limites de service EC2)
- Qu'est-ce qu'Service Quotas ?
- How to Request quota increase (Comment demander une augmentation du quota)
- Service endpoints and quotas (Points de terminaison et quotas de service)
- Guide de l'utilisateur Service Quotas
- Quota Monitor for AWS (Surveillance de quotas pour AWS)
- AWS Fault Isolation Boundaries (Limites d'isolement des pannes AWS)

- Availability with redundancy (Disponibilité avec redondance)
- AWS pour les données
- Qu'est-ce que l'intégration continue ?
- Qu'est-ce que la livraison continue ?
- Partenaire APN : partenaires facilitant la gestion de la configuration
- Managing the account lifecycle in account-per-tenant SaaS environments on AWS (Gestion du cycle de vie des comptes dans les environnements SaaS de type compte par locataire sur AWS)
- Gestion et surveillance de la limitation des API dans vos charges de travail
- View AWS Trusted Advisor recommendations at scale with AWS Organizations (Examiner les recommandations AWS Trusted Advisor à grande échelle avec AWS Organizations)
- Automating Service Limit Increases and Enterprise Support with AWS Control Tower (Automatisation de l'augmentation des limites de service et Enterprise Support avec AWS Control Tower)
- Actions, ressources et clés de condition pour les Service Quotas

Vidéos connexes :

- AWS Live re:Inforce 2019 - Service Quotas
- View and Manage Quotas for AWS Services Using Service Quotas (Examiner et gérer les quotas pour les services AWS avec les quotas de service)
- AWS IAM Quotas Demo (Démonstration sur les quotas IAM d'AWS)
- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small

Outils associés :

- AWS CodeDeploy
- AWS CloudTrail
- Amazon CloudWatch
- Amazon EventBridge
- Amazon DevOps Guru
- AWS Config
- AWS Trusted Advisor

- AWS CDK
- AWS Systems Manager
- AWS Marketplace

Planification de votre topologie réseau

Les charges de travail existent souvent dans plusieurs environnements. Il s'agit notamment de plusieurs environnements cloud (accessibles publiquement et privés) et éventuellement de votre infrastructure de centre de données existante. Les plans doivent tenir compte de facteurs liés au réseau : connectivité intrasystème et intersystème, gestion des adresses IP publiques, gestion des adresses IP privées et résolution des noms de domaine.

Lorsque vous concevez des systèmes à l'aide de réseaux basés sur des adresses IP, vous devez planifier la topologie du réseau et l'adressage en prévision d'éventuelles pannes et pour faire face à une future croissance et à l'intégration à d'autres systèmes à leurs réseaux.

Amazon Virtual Private Cloud (Amazon VPC) permet d'allouer une section isolée et privée du cloud AWS où vous pouvez lancer des ressources AWS dans un réseau virtuel.

Bonnes pratiques

- REL02-BP01 Utiliser une connectivité réseau hautement disponible pour vos points de terminaison publics de charge de travail
- REL02-BP02 Mettre en service une connectivité redondante entre les réseaux privés dans le cloud et les environnements sur site
- REL02-BP03 S’assurer que l’allocation des sous-réseaux IP tient compte de l’expansion et de la disponibilité
- REL02-BP04 Préférer les topologies en étoile au maillage « many-to-many »
- REL02-BP05 Appliquer des plages d’adresses IP privées sans chevauchement dans tous les espaces d’adressage privés où ils sont connectés

REL02-BP01 Utiliser une connectivité réseau hautement disponible pour vos points de terminaison publics de charge de travail

La mise en place d'une connectivité réseau hautement disponible aux points de terminaison publics de vos charges de travail peut vous aider à réduire les temps d'arrêt dus à la perte de connectivité

et à améliorer la disponibilité et le SLA de votre charge de travail. Pour ce faire, utilisez le DNS hautement disponible, les réseaux de diffusion de contenu (CDN), des passerelles API, l'équilibrage de charge ou les proxys inverses.

Résultat souhaité : il est essentiel de planifier, de construire et de rendre opérationnelle une connectivité réseau hautement disponible pour vos points de terminaison publics. Si votre charge de travail devient inaccessible en raison d'une perte de connectivité, même si elle est en cours d'exécution et disponible, vos clients verront votre système comme étant en panne. En combinant une connectivité réseau hautement disponible et résiliente pour les points de terminaison publics de votre charge de travail, ainsi qu'une architecture résiliente pour votre charge de travail elle-même, vous pouvez offrir la meilleure disponibilité et le meilleur niveau de service possible à vos clients.

Les services AWS Global Accelerator, Amazon CloudFront, Amazon API Gateway, les URL de fonction AWS Lambda, les API AWS AppSync et Elastic Load Balancing (ELB) fournissent tous des points de terminaison publics hautement disponibles. Amazon Route 53 fournit un service DNS hautement disponible pour la résolution des noms de domaine afin de vérifier que les adresses de vos points de terminaison publics peuvent être résolues.

Vous pouvez également évaluer des appliances logicielles AWS Marketplace pour l'équilibrage de charge et les proxys.

Anti-modèles courants :

- Concevoir une charge de travail hautement disponible sans planifier le DNS et la connectivité réseau pour la haute disponibilité.
- Utilisation d'adresses Internet publiques sur des instances ou des conteneurs individuels et gestion de la connectivité à ces adresses avec le DNS.
- Utilisation des adresses IP au lieu des noms de domaine pour localiser les services.
- Ne pas tester des scénarios où la connectivité à vos points de terminaison publics est perdue.
- Ne pas analyser les besoins en débit du réseau et les modèles de distribution.
- Ne pas tester et planifier des scénarios dans lesquels la connectivité du réseau Internet à vos points de terminaison publics de votre charge de travail pourrait être interrompue.
- Fourniture du contenu (comme les pages web, les ressources statiques ou les fichiers multimédias) à une grande zone géographique sans utilisation d'un réseau de diffusion de contenu.
- Ne pas se préparer aux attaques par déni de service distribué (DDoS). Les attaques DDoS risquent d'interrompre le trafic légitime et de réduire la disponibilité pour vos utilisateurs.

Avantages liés au respect de cette bonne pratique : la conception d'une connectivité réseau hautement disponible et résiliente garantit que votre charge de travail est accessible et disponible pour vos utilisateurs.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

Le routage du trafic est au cœur de la mise en place d'une connectivité réseau hautement disponible pour vos points de terminaison publics. Pour vérifier que votre trafic est en mesure d'atteindre les points de terminaison, le DNS doit être capable de résoudre les noms de domaine à leurs adresses IP correspondantes. Utilisez un système de nom de domaine (DNS) hautement disponible et évolutif tel que Amazon Route 53 pour gérer les enregistrements DNS de votre domaine. Vous pouvez également utiliser les surveillances de l'état fournies par Amazon Route 53. Les surveillances de l'état permettent de s'assurer que votre application est accessible, disponible et fonctionnelle. Elles peuvent être configurées de manière à imiter le comportement de l'utilisateur, comme la demande d'une page web ou d'une URL spécifique. En cas de panne, Amazon Route 53 répond aux demandes de résolution DNS et dirige uniquement le trafic vers les points de terminaison en bonne santé. Vous pouvez également envisager d'utiliser les fonctionnalités de Geo DNS et de routage basé sur la latence offertes par Amazon Route 53.

Pour vérifier que votre charge de travail elle-même est hautement disponible, utilisez Elastic Load Balancing (ELB).Amazon Route 53 peut cibler le trafic vers ELB, qui distribue le trafic vers les instances de calcul cibles. Vous pouvez également utiliser Amazon API Gateway avec AWS Lambda pour une solution sans serveur. Les clients peuvent également exécuter des charges de travail dans plusieurs Régions AWS. Avec le modèle multisite actif/actif, la charge de travail peut servir le trafic de plusieurs régions. Avec un modèle multisite actif/passif, la charge de travail sert le trafic de la région active tandis que les données sont répliquées dans la région secondaire et deviennent actives en cas de panne de la région principale. Les surveillances de l'état Route 53 peuvent alors contrôler le basculement DNS de n'importe quel point de terminaison dans une région principale vers un point de terminaison dans une région secondaire, vérifiant que votre charge de travail est accessible et disponible pour vos utilisateurs.

Amazon CloudFront fournit une API simple pour distribuer du contenu avec une faible latence et des taux de transfert de données élevés en répondant aux demandes à l'aide d'un réseau d'emplacements périphériques dans le monde entier. Les réseaux de diffusion de contenu (CDN) servent les clients en proposant un contenu situé ou mis en cache à un endroit proche de l'utilisateur. La disponibilité de votre application s'en trouve également améliorée, car la charge de contenu est déplacée de vos serveurs vers lesemplacements périphériques de CloudFront. Les emplacements

périphériques et les caches périphériques régionaux conservent des copies en cache de votre contenu à proximité de vos utilisateurs, ce qui permet une récupération rapide et augmente l'accessibilité et la disponibilité de votre charge de travail.

Pour les charges de travail avec des utilisateurs dispersés géographiquement, AWS Global Accelerator améliore la disponibilité et les performances des applications. AWS Global Accelerator fournit des adresses IP statiques Anycast qui servent de point d'entrée fixe à votre application hébergée dans une ou plusieurs Régions AWS. Cela permet au trafic d'entrer sur le réseau mondial AWS aussi près que possible de vos utilisateurs, améliorant ainsi l'accessibilité et la disponibilité de votre charge de travail. AWS Global Accelerator surveille également l'état de santé des points de terminaison de vos applications en utilisant la surveillance de l'état TCP, HTTP et HTTPS. Toute modification de l'état ou de la configuration de vos points de terminaison déclenche la redirection du trafic utilisateur vers des points de terminaison sains qui offrent les meilleures performances et la meilleure disponibilité à vos utilisateurs. De plus, AWS Global Accelerator est conçu pour être isolé des pannes et utilise deux adresses IPv4 statiques qui sont desservies par des zones réseau indépendantes, ce qui augmente la disponibilité de vos applications.

Pour aider à protéger les clients contre les attaques DDoS, AWS propose AWS Shield Standard. Shield Standard est automatiquement activé et protège contre les attaques d'infrastructure courantes (couches 3 et 4) telles que les inondations SYN/UDP et les attaques par réflexion pour assurer la haute disponibilité de vos applications sur AWS. Pour bénéficier de protections supplémentaires contre des attaques plus sophistiquées et plus importantes (comme les inondations UDP), les attaques par épuisement d'état (comme les inondations TCP SYN), et pour aider à protéger vos applications fonctionnant sur Amazon Elastic Compute Cloud (Amazon EC2), Elastic Load Balancing (ELB), Amazon CloudFront, AWS Global Accelerator, et Route 53, envisagez d'utiliser AWS Shield Advanced. Pour se protéger contre les attaques au niveau de la couche application, comme les inondations HTTP POST ou GET, utilisez AWS WAF. AWS WAF peut utiliser les adresses IP, les en-têtes HTTP, le corps HTTP, les chaînes URI, l'injection SQL et les conditions de script intersite pour déterminer si une requête doit être bloquée ou autorisée.

Étapes d'implémentation

1. Configurez un DNS hautement disponible : Amazon Route 53 est un service web de système de nom de domaine (DNS) hautement disponible et évolutif. Route 53 connecte les demandes des utilisateurs aux applications Internet fonctionnant sur AWS ou sur site. Pour obtenir plus d'informations, consultez Configuration d'Amazon Route 53 en tant que service DNS.
2. Configurez la surveillance de l'état : lorsque vous utilisez Route 53, vérifiez que seules les cibles saines sont résolubles. Commencez par créer des surveillances de l'état Route 53 et par

configurer le basculement DNS. Il est important de tenir compte des aspects suivants lors de la mise en place des surveillances de l'état :

   a. Comment Amazon Route 53 détermine si une vérification de l'état est saine
   b. Création, mise à jour et suppression de vérifications de l'état
   c. Statut de la surveillance de l'état et réception de notifications
   d. Bonnes pratiques relatives à Amazon Route 53 DNS

3. Connectez votre service DNS à vos points de terminaison.
   a. Lorsque vous utilisez Elastic Load Balancing comme cible pour votre trafic, créez un enregistrement d'alias utilisant Amazon Route 53 qui pointe vers le point de terminaison régional de votre équilibreur de charge. Pendant la création de l'enregistrement de l'alias, réglez l'option « Évaluer l'état de la cible » sur « Oui ».
   b. Pour les charges de travail sans serveur ou les API privées, lorsque vous utilisez API Gateway, utilisez Route 53 pour router le trafic vers API Gateway.
4. Choisissez un réseau de diffusion de contenu.
   a. Pour diffuser du contenu en utilisant des emplacements périphériques plus proches de l'utilisateur, il faut commencer par comprendre comment CloudFront diffuse le contenu.
   b. Commencez par une simple distribution CloudFront. CloudFront comprend alors l'endroit d'où vous souhaitez que le contenu soit diffusé, ainsi que les détails concernant le suivi et la gestion de la diffusion du contenu. Il est important de comprendre et de prendre en compte les aspects suivants lors de la mise en place de la distribution CloudFront :
      i. Fonctionnement de la mise en cache avec les emplacements périphériques CloudFront
      ii. Augmentation de la proportion de demandes servies directement à partir des caches CloudFront (taux d'accès au cache)
      iii. Utilisation Amazon CloudFront Origin Shield
      iv. Optimisation de la haute disponibilité avec le basculement d'origine CloudFront
5. Configurez la protection de la couche d'application : AWS WAF vous aide à vous protéger contre les exploits et les bots web courants qui peuvent affecter la disponibilité, compromettre la sécurité ou consommer des ressources excessives. Pour mieux comprendre, examinez comment AWS WAF fonctionne et quand vous êtes prêt à mettre en œuvre les protections de la couche application HTTP POST ET GET, consultez Getting started with AWS WAF (Démarrer avec AWS WAF). Vous pouvez également utiliser AWS WAF avec CloudFront. Consultez la documentation pour comprendre comment AWS WAF fonctionne avec les fonctionnalités Amazon CloudFront.

6. Configurez une protection DDoS supplémentaire : par défaut, tous les clients AWS bénéficient d'une protection contre les attaques DDoS les plus fréquentes au niveau de la couche réseau et de la couche transport qui ciblent votre site web ou votre application avec AWS Shield Standard, et ce sans frais supplémentaires. Pour bénéficier d'une protection supplémentaire des applications accessibles sur Internet et fonctionnant sur Amazon EC2, Elastic Load Balancing, Amazon CloudFront, AWS Global Accelerator, et Amazon Route 53, vous pouvez envisager AWS Shield Advanced et passer en revue des exemples d'architectures résistantes aux attaques DDoS. Pour protéger votre charge de travail et vos points de terminaison publics contre les attaques DDoS, consultez Getting started with AWS Shield Advanced (Démarrer avec AWS Shield Advanced).

Ressources

Bonnes pratiques associées :

- REL10-BP01 Déployer la charge de travail sur plusieurs emplacements
- REL10-BP02 Sélectionner les emplacements appropriés pour votre déploiement multisite
- REL11-BP04 S'appuyer sur le plan de données et non sur le plan de contrôle pendant la récupération
- REL11-BP06 Envoyer des notifications lorsque des événements affectent la disponibilité

Documents connexes :

- Partenaire APN : partenaires pouvant vous aider à planifier votre mise en réseau
- AWS Marketplace pour l'infrastructure réseau
- Qu'est-ce que AWS Global Accelerator ?
- Qu'est-ce qu'Amazon CloudFront ?
- Qu'est-ce qu'Amazon Route 53 ?
- Qu'est-ce qu'Elastic Load Balancing ?
- Capacité de connectivité du réseau : établir les fondements de votre cloud
- Qu'est-ce que Amazon API Gateway ?
- Que sont AWS WAF, AWS Shield, et AWS Firewall Manager ?
- What is Amazon Route 53 Application Recovery Controller? (Qu'est-ce que le contrôleur de récupération d'application d'Amazon Route 53 ?)

- Configurer des surveillances de l'état personnalisées pour le basculement DNS

Vidéos connexes :

- AWS re:Invent 2022 - Improve performance and availability with AWS Global Accelerator
- AWS re:Invent 2020: Global traffic management with Amazon Route 53
- AWS re:Invent 2022 - Operating highly available Multi-AZ applications
- AWS re:Invent 2022 - Dive deep on AWS networking infrastructure
- AWS re:Invent 2022 - Building resilient networks

Exemples connexes :

- Disaster Recovery with Amazon Route 53 Application Recovery Controller (ARC) [Reprise après sinistre avec le contrôleur de récupération d'application (ARC) d'Amazon Route 53]
- Ateliers sur la fiabilité
- Atelier AWS Global Accelerator

REL02-BP02 Mettre en service une connectivité redondante entre les réseaux privés dans le cloud et les environnements sur site

Mettez en œuvre une redondance dans vos connexions entre les réseaux privés dans le cloud et les environnements sur site pour obtenir la résilience de la connectivité. Cela peut se faire en déployant au moins deux liens et chemins de trafic, afin de préserver la connectivité en cas de défaillance du réseau.

Anti-modèles courants :

- Vous dépendez d’une seule connexion réseau, ce qui crée un point de défaillance unique.
- Vous utilisez un seul tunnel VPN ou plusieurs tunnels qui aboutissent à la même zone de disponibilité.
- Vous dépendez d’un seul fournisseur de services Internet (FSI) pour la connectivité VPN, ce qui peut entraîner des défaillances complètes en cas de panne de ce dernier.
- Vous n’implémentez pas de protocoles de routage dynamique tels que BGP, qui sont essentiels pour rediriger le trafic lors de perturbations du réseau.

- Vous ignorez les limites de bande passante des tunnels VPN et surestimez leurs capacités de secours.

Avantages liés au respect de cette bonne pratique : grâce à la mise en œuvre d'une connectivité redondante entre votre environnement cloud et votre environnement d'entreprise/sur site, les services dépendants entre les deux environnements peuvent communiquer de manière fiable.

Niveau de risque exposé si cette bonne pratique n'est pas établie : élevé

Directives d'implémentation

Lorsque vous utilisez AWS Direct Connect pour connecter votre réseau sur site à AWS, vous pouvez obtenir une résilience réseau maximale (SLA de 99,99 %) en utilisant des connexions distinctes qui mènent à des appareils distincts dans plusieurs emplacements sur site et plusieurs emplacements AWS Direct Connect. Cette topologie offre une résilience contre les pannes d'appareils, les problèmes de connectivité et les pannes complètes d'un site. Vous pouvez également atteindre une résilience élevée (SLA de 99,9 %) en utilisant deux connexions individuelles à plusieurs emplacements (chaque emplacement sur site étant connecté à un seul emplacement Direct Connect). Cette approche assure une protection contre les interruptions de connectivité causées par des coupures de fibre ou des pannes d'appareils, et contribue à pallier des défaillances complètes d'un site. AWS Direct Connect Resiliency Toolkit peut vous aider à concevoir votre topologie AWS Direct Connect.

Vous pouvez également envisager qu'AWS Site-to-Site VPN mène à une passerelle AWS Transit Gateway faisant office de solution de secours à moindre coût de votre connexion AWS Direct Connect principale. Cette configuration permet un routage ECMP (Equal cost multi-path) via plusieurs tunnels VPN, permettant d'obtenir un débit allant jusqu'à 50 Gbit/s, bien que chaque tunnel VPN soit plafonné à 1,25 Gbit/s. Il est toutefois important de noter qu'AWS Direct Connect constitue toujours le choix le plus efficace pour réduire au maximum les perturbations du réseau et assurer une connectivité stable.

Lorsque vous utilisez des VPN sur Internet pour connecter votre environnement cloud à votre centre de données sur site, configurez deux tunnels VPN dans le cadre d'une connexion VPN unique de site à site. Chaque tunnel doit mener à une zone de disponibilité différente pour garantir la haute disponibilité, et utiliser du matériel redondant pour éviter une panne d'appareil sur site. En outre, envisagez plusieurs connexions Internet provenant de différents fournisseurs de services Internet (FSI) à votre emplacement sur site pour éviter une interruption complète de la connectivité VPN en cas de panne au niveau d'un FSI. Pour garantir une connectivité à haute disponibilité, il convient de

sélectionner des fournisseurs de services Internet offrant un routage et une infrastructure diversifiés, notamment avec des chemins physiques distincts vers les points de terminaison AWS.

Outre la redondance physique avec plusieurs connexions AWS Direct Connect et plusieurs tunnels VPN (ou une combinaison des deux), il est essentiel de mettre en œuvre un routage dynamique via le protocole BGP (Border Gateway Protocol). Le protocole BGP dynamique permet de rediriger automatiquement le trafic d'un chemin vers un autre en fonction des conditions en temps réel du réseau et des politiques configurées. Ce comportement dynamique est particulièrement utile pour maintenir la disponibilité du réseau et la continuité des services en cas de panne de liaison ou de réseau. Il sélectionne rapidement des chemins alternatifs, améliorant ainsi la résilience et la fiabilité du réseau.

Étapes d'implémentation

- Bénéficiez d'une connectivité hautement disponible entre AWS et votre environnement sur site.
  - Utilisez plusieurs connexions AWS Direct Connect ou tunnels VPN entre des réseaux privés déployés séparément.
  - Utilisez plusieurs emplacements AWS Direct Connect pour une haute disponibilité.
  - Si vous utilisez plusieurs Régions AWS, mettez en œuvre la redondance dans au moins deux d'entre elles.
- Utilisez AWS Transit Gateway dans la mesure du possible pour mettre fin à votre connexion VPN.
- Évaluez les appliances AWS Marketplace pour mettre fin aux VPN ou étendre votre SD-WAN à AWS. Si vous utilisez des appliances AWS Marketplace, déployez des instances redondantes pour une plus haute disponibilité dans différentes zones de disponibilité.
- Assurez-vous que vous disposez d'une connexion redondante à votre environnement sur site.
  - Il se peut que vous ayez besoin de connexions redondantes à plusieurs Régions AWS pour répondre à vos besoins en matière de disponibilité.
  - Utilisez AWS Direct Connect Resiliency Toolkit pour commencer.

Ressources

Documents connexes :

- AWS Direct Connect Resiliency Recommendations
- Utilisation de connexions Site-to-Site VPN redondantes pour assurer le basculement
- Politiques de routage et communautés BGP

- Configurations actives/actives et actives/passives dans AWS Direct Connect
- Partenaire APN : partenaires pouvant vous aider à planifier votre mise en réseau
- AWS Marketplace pour l'infrastructure réseau
- Livre blanc sur les options de connectivité Amazon Virtual Private Cloud
- Création d'une infrastructure réseau AWS multi-VPC évolutive et sécurisée
- Utilisation de connexions Site-to-Site VPN redondantes pour assurer le basculement
- Utilisation d'AWS Direct Connect Resiliency Toolkit pour commencer
- Points de terminaison de VPC et services de point de terminaison de VPC (AWS PrivateLink)
- Qu'est-ce qu'Amazon VPC ?
- Qu'est-ce qu'une passerelle de transit ?
- Qu'est-ce qu'AWS Site-to-Site VPN ?
- Utilisation des passerelles Direct Connect

Vidéos connexes :

- AWS re:Invent 2018: Advanced VPC Design and New Capabilities for Amazon VPC
- AWS re:Invent 2019: AWS Transit Gateway reference architectures for many VPCs

REL02-BP03 S'assurer que l'allocation des sous-réseaux IP tient compte de l'expansion et de la disponibilité

Les plages d'adresses IP de Amazon VPC doivent être assez grandes pour répondre aux exigences de charges de travail, y compris en prévision d'une expansion et de l'allocation d'adresses IP aux sous-réseaux sur les zones de disponibilité. Cela inclut les équilibreurs de charge, les instances EC2 et les applications basées sur conteneur.

Lorsque vous planifiez votre topologie de réseau, la première étape consiste à définir l'espace d'adressage IP lui-même. Des plages d'adresses IP privées (conformément aux directives RFC 1918) doivent être allouées pour chaque VPC. Vous devez remplir les exigences suivantes dans le cadre de ce processus :

- Autorisez un espace d'adressage IP pour plus d'un VPC par région.
- Au sein d'un VPC, prévoyez de l'espace pour plusieurs sous-réseaux afin de pouvoir couvrir plusieurs zones de disponibilité.

- Pensez à laisser de l'espace de bloc CIDR non utilisé au sein d'un VPC pour une future expansion.
- Assurez-vous qu'il existe un espace d'adressage IP pour répondre aux besoins de tout parc transitoire d'instances Amazon EC2 que vous pourriez utiliser, comme les parcs d'instances Spot pour le machine learning, les clusters Amazon EMR ou les clusters Amazon Redshift. Une attention similaire doit être accordée aux clusters Kubernetes, tels qu'Amazon Elastic Kubernetes Service (Amazon EKS), car chaque pod Kubernetes se voit attribuer une adresse routable depuis le bloc CIDR du VPC par défaut.
- Remarque : les quatre premières adresses IP et la dernière adresse IP dans le bloc CIDR de chaque sous-réseau sont réservées et ne peuvent pas être utilisées.
- Notez que le bloc CIDR initial du VPC alloué à votre VPC ne peut pas être modifié ou supprimé, mais vous pouvez ajouter des blocs CIDR non superposés au VPC. Les CIDR IPv4 de sous-réseau ne sont pas modifiables, mais les CIDR IPv6 le sont.
- Le plus grand bloc d'adresse CIDR VPC possible est un /16 et le plus petit est un /28.
- Envisagez d'autres réseaux connectés (VPC, sur site ou autres fournisseurs de cloud) et veillez à ce que l'espace d'adressage IP ne se chevauche pas. Pour plus d'informations, consultez REL02-BP05 Appliquer des plages d'adresses IP privées sans chevauchement dans tous les espaces d'adressage privés où ils sont connectés.

Résultat souhaité : Un sous-réseau IP évolutif peut vous aider à faire face à la croissance future et à éviter tout gaspillage inutile.

Anti-modèles courants :

- Ignorer la croissance future et utiliser des blocs CIDR trop petits, qui requièrent une reconfiguration, ce qui peut entraîner des temps d'arrêt.
- Estimation incorrecte du nombre d'adresses IP qu'un Elastic Load Balancer peut utiliser.
- Déploiement de nombreux équilibreurs de charge à trafic élevé dans les mêmes sous-réseaux.
- Utilisation de mécanismes de dimensionnement automatisés sans surveiller la consommation d'adresses IP.
- La définition de plages CIDR excessivement étendues va bien au-delà des prévisions de croissance futures, ce qui peut entraîner des difficultés pour l'appairage avec d'autres réseaux dont les plages d'adresses se chevauchent.

Avantages liés au respect de cette bonne pratique : Ces tailles sont la garantie que vous pouvez prendre en charge la croissance de vos charges de travail et continuer à fournir une disponibilité au fur et à mesure de votre mise à l'échelle.

Niveau d'exposition au risque si cette bonne pratique n'est pas respectée : Moyen

Directives d'implémentation

Planifiez votre réseau en prévision de votre croissance, de la conformité réglementaire et de son intégration avec d'autres composants. La croissance peut être sous-estimée, la conformité réglementaire peut changer, et les acquisitions ou les connexions à des réseaux privés peuvent être difficiles à implémenter sans une planification appropriée.

- Sélectionnez les régions et Comptes AWS pertinents en fonction de vos exigences de services, de la latence, des exigences réglementaires et de reprise après sinistre (DR).
- Identifiez vos besoins pour les déploiements VPC régionaux.
- Identifiez la taille des VPC.
  - Déterminez si vous allez déployer une connectivité multi-VPC.
    - Qu'est-ce que Transit Gateway ?
    - Connectivité multi-VPC dans un seule région
  - Déterminez si vous avez besoin d'une mise en réseau séparée pour les exigences réglementaires.
  - Créez des VPC avec des blocs CIDR de taille appropriée pour répondre à vos besoins actuels et futurs.
    - Si vos prévisions de croissance sont inconnues, il peut être préférable d'opter pour des blocs CIDR plus importants afin de réduire le risque de reconfiguration future.
  - Envisagez d'utiliser l'adressage IPv6 pour les sous-réseaux dans le cadre d'un VPC à double pile. IPv6 convient parfaitement à une utilisation dans des sous-réseaux privés contenant des flottes d'instances éphémères ou des conteneurs qui nécessiteraient autrement un grand nombre d'adresses IPv4.

Ressources

Bonnes pratiques Well-Architected connexes :

- REL02-BP05 Appliquer des plages d'adresses IP privées sans chevauchement dans tous les espaces d'adressage privés où ils sont connectés

Documents connexes :

- Partenaire APN : partenaires pouvant vous aider à planifier votre mise en réseau
- AWS Marketplace pour l'infrastructure réseau
- Livre blanc sur les options de connectivité Amazon Virtual Private Cloud
- Connectivité réseau haute disponibilité de plusieurs centres de données
- Connectivité multi-VPC dans un seule région
- What Is Amazon VPC?
- IPv6 on AWS
- IPv6 on reference architectures
- Amazon Elastic Kubernetes Service launches IPv6 support

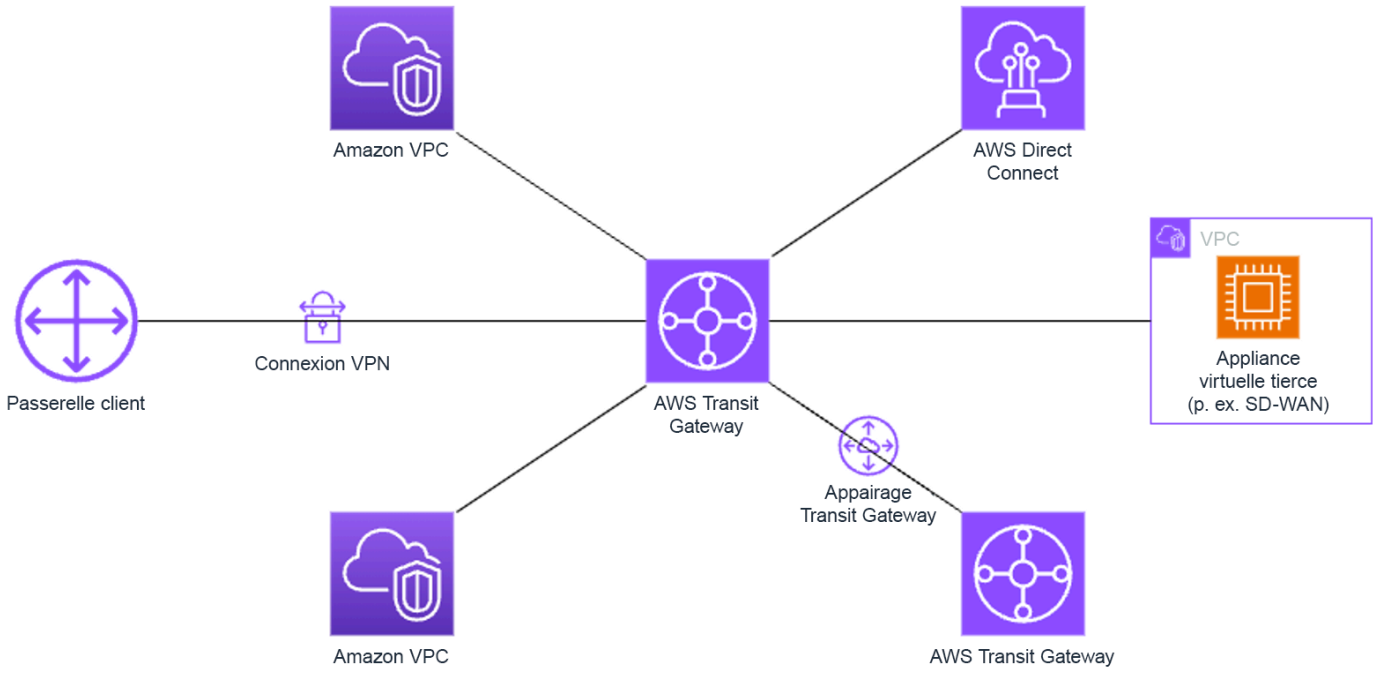
Vidéos connexes :

- AWS re:Invent 2018: Advanced VPC Design and New Capabilities for Amazon VPC (NET303)
- AWS re:Invent 2019: AWS Transit Gateway reference architectures for many VPCs (NET406-R1)
- AWS re:Invent 2023: AWS Ready for what's next? Designing networks for growth and flexibility (NET310)

REL02-BP04 Préférer les topologies en étoile au maillage « many-to-many »

Lorsque vous connectez plusieurs réseaux privés, tels que des clouds privés virtuels (VPC) et des réseaux locaux, optez pour une topologie en étoile plutôt qu'une topologie maillée. Contrairement aux topologies maillées, où chaque réseau se connecte directement aux autres et augmente la complexité et les frais de gestion, l'architecture en étoile centralise les connexions via un hub unique. Cette centralisation simplifie la structure du réseau et améliore son opérabilité, son évolutivité et son contrôle.

AWS Transit Gateway est un service géré, évolutif et hautement disponible conçu pour la construction de réseaux en étoile sur AWS. Il fait office de hub central de votre réseau et assure la segmentation du réseau, le routage centralisé et la connexion simplifiée aux environnements cloud et sur site. La figure suivante montre comment utiliser AWS Transit Gateway pour créer une topologie en étoile.



Anti-modèles courants :

- Vous compliquez à l'excès les politiques de routage dans une architecture en étoile, ce qui réduit l'efficacité du réseau et complique à la fois le dépannage et la gestion proactive.
- Une segmentation insuffisante basée sur le routage au sein du hub pourrait entraîner des vulnérabilités, susceptibles d'exposer le réseau à un accès non autorisé.
- Sans une optimisation minutieuse, le trafic acheminé via le hub peut entraîner des coûts de transfert de données plus élevés, en particulier pour le trafic traversant les zones de disponibilité et les régions. Des stratégies efficaces de gestion du trafic sont essentielles pour contrôler les dépenses.

Avantages de la mise en place de cette bonne pratique : à mesure que le nombre de réseaux connectés augmente, la gestion et l'expansion de la connectivité maillée deviennent de plus en plus difficiles. AWS Transit Gateway propose un hub géré évolutif et fiable pour la construction et l'exploitation de vos topologies en étoile. Lorsque vous utilisezAWS Transit Gateway, vous pouvez établir des connexions et centraliser le routage du trafic sur plusieurs réseaux.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

- Planifiez votre réseau.
- Créez votre AWS Transit Gateway.
- Attachez vos VPC.
- Si nécessaire, créez des connexions VPN ou des passerelles Direct Connect et associez-les à la passerelle Transit Gateway.
- Définissez la façon dont le trafic est acheminé entre les VPC connectés et les autres connexions via la configuration de vos tables de routage Transit Gateway.
- Utilisez Amazon CloudWatch pour surveiller et ajuster les configurations selon les besoins afin d'optimiser les performances et les coûts.

Ressources

Documents connexes :

- Qu'est-ce que Transit Gateway ?
- Création d'une infrastructure réseau AWS multi-VPC évolutive et sécurisée
- Building a global network using AWS Transit Gateway Inter-Region peering
- Amazon Virtual Private Cloud Connectivity Options
- Partenaire APN : partenaires pouvant vous aider à planifier votre mise en réseau
- AWS Marketplace pour l'infrastructure réseau

Vidéos connexes :

- AWS re:Invent 2023 - AWS networking foundations
- AWS re:Invent 2023 - Advanced VPC designs and new capabilities

REL02-BP05 Appliquer des plages d'adresses IP privées sans chevauchement dans tous les espaces d'adressage privés où ils sont connectés

Les plages d'adresses IP de vos VPC ne doivent pas se chevaucher lorsque les réseaux sont appairés, connectés via Transit Gateway ou connectés via un VPN. Évitez les conflits d'adresses IP

entre un VPC et des environnements sur site ou avec d'autres fournisseurs de services cloud que vous utilisez. Vous devez également disposer d'un moyen d'allouer des plages d'adresses IP privées lorsque cela est nécessaire. Un système de gestion des adresses IP (IPAM) peut aider à automatiser cette opération.

Résultat souhaité :

- Aucun conflit de plage d'adresses IP entre les VPC, les environnements sur site ou d'autres fournisseurs de services cloud.
- Une bonne gestion des adresses IP permet d'adapter plus facilement l'infrastructure réseau à la croissance et à l'évolution des exigences en matière de réseau.

Anti-modèles courants :

- Utilisation dans votre VPC d'une plage d'adresses IP identique à celle que vous utilisez sur site, dans votre réseau d'entreprise ou auprès d'autres fournisseurs de services cloud.
- Non suivi des plages d'adresses IP des VPC utilisés pour déployer vos charges de travail.
- Utilisation de processus manuels de gestion des adresses IP, tels que des feuilles de calcul.
- Surdimensionnement ou sous-dimensionnement des blocs CIDR, entraînant un gaspillage d'adresses IP ou un espace d'adressage insuffisant pour votre charge de travail.

Avantages liés au respect de cette bonne pratique : La planification active de votre réseau garantit que vous n'avez pas plusieurs occurrences de la même adresse IP dans des réseaux interconnectés. Cela empêche les problèmes de routage de se produire dans certaines parties de la charge de travail qui utilisent les différentes applications.

Niveau de risque exposé si cette bonne pratique n'est pas établie: moyen

Directives d'implémentation

Utilisez un gestionnaire IPAM, tel que le Amazon VPC IP Address Manager, pour surveiller et gérer votre utilisation du routage CIDR. Plusieurs gestionnaires IPAM sont également disponibles sur AWS Marketplace. Évaluez votre utilisation potentielle sur AWS, ajoutez des plages CIDR à des VPC existants et créez des VPC pour autoriser une croissance d'utilisation planifiée.

Étapes d'implémentation

- Capturez votre consommation CIDR actuelle (par exemple, VPC et sous-réseaux).

    - Utilisez des opérations d'API de service pour collecter la consommation CIDR actuelle.
    - Utilisez le Amazon VPC IP Address Manager pour découvrir des ressources.
- Capturez l'utilisation actuelle de votre sous-réseau.
    - Utilisez des opérations d'API de service pour collecter des sous-réseaux par VPC dans chaque région.
    - Utilisez le Amazon VPC IP Address Manager pour découvrir des ressources.
- Enregistrez l'utilisation actuelle.
- Déterminez si vous avez créé des plages d'adresses IP se chevauchant.
- Calculez la capacité inutilisée.
- Identifiez les plages d'adresses IP qui se chevauchent. Vous pouvez migrer vers une nouvelle plage d'adresses ou envisager d'utiliser des techniques telles que la passerelle NAT privée ou AWS PrivateLink si vous devez connecter les plages qui se chevauchent.

Ressources

Bonnes pratiques associées :

- Protection des réseaux

Documents connexes :

- Partenaire APN : partenaires pouvant vous aider à planifier votre mise en réseau
- AWS Marketplace pour l'infrastructure réseau
- Livre blanc sur les options de connectivité Amazon Virtual Private Cloud
- Connectivité réseau haute disponibilité de plusieurs centres de données
- Connexion de réseaux dont les plages d'adresses IP se chevauchent
- Qu'est-ce qu'Amazon VPC ?
- Qu'est-ce qu'IPAM ?

Vidéos connexes :

- AWS re:Invent 2023 - Advanced VPC designs and new capabilities
- AWS re:Invent 2019: AWS Transit Gateway reference architectures for many VPCs

- AWS re:Invent 2023 - Ready for what’s next? Designing networks for growth and flexibility
- AWS re:Invent 2021 - {New Launch} Manage your IP addresses at scale on AWS

Architecture de charge de travail

Pour garantir la fiabilité d'une charge de travail, il faut commencer par choisir le bon logiciel et la bonne infrastructure. Vos choix d'architecture ont un impact sur le comportement des charges de travail sur les cinq piliers Well-Architected. Pour des raisons de fiabilité, vous devez suivre des modèles spécifiques.

Les sections suivantes expliquent les bonnes pratiques à utiliser avec ces modèles de fiabilité.

Rubriques

Conception de l'architecture de votre service de charge de travail

Créez des charges de travail hautement évolutives et fiables à l'aide d'une architecture orientée service (SOA) ou d'une architecture de microservices. La SOA consiste à rendre les composants logiciels réutilisables via les interfaces de service. L'architecture des microservices va plus loin, en particulier en rendant les composants plus petits et plus simples.

Les interfaces d'architecture orientée service (SOA) utilisent des normes de communication communes permettant leur intégration rapide à de nouvelles charges de travail. La SOA a remplacé la pratique consistant à créer des architectures monolithes, composées d'unités interdépendantes et indivisibles.

Chez AWS, nous avons toujours utilisé la SOA. Nous concevons toutefois désormais nos systèmes à l'aide de microservices. Bien que les micro-services présentent plusieurs qualités attractives, l'avantage le plus important pour la disponibilité est le fait que les microservices sont plus légers et plus simples. Ils vous permettent de différencier la disponibilité requise par les différents services, et ainsi de concentrer plus spécifiquement vos investissements sur les microservices qui présentent les besoins les plus importants en disponibilité. Par exemple, pour fournir des pages d'informations produits sur Amazon.com (« pages de détails »), des centaines de microservices sont appelés pour construire des parties distinctes de la page. Tandis que certains services doivent être disponibles pour fournir le prix et les détails des produits, la grande majorité du contenu de la page peut simplement être exclu si le service n'est pas disponible. Même des éléments tels que les photos et

commentaires ne sont pas requis pour fournir une expérience permettant à un client d'acheter un produit.

Bonnes pratiques

- REL03-BP01 Choisir comment segmenter votre charge de travail
- REL03-BP02 Créer des services axés sur des domaines d'activité et la fonctionnalité
- REL03-BP03 Fournir des contrats de service par API

REL03-BP01 Choisir comment segmenter votre charge de travail

La segmentation de la charge de travail est importante lorsqu'il s'agit de déterminer les exigences de résilience de votre application. L'architecture monolithique doit être évitée dans la mesure du possible. À la place, réfléchissez bien aux composants de l'application capables d'être divisés en microservices. Selon les exigences de votre application, il peut s'agir d'une combinaison d'une architecture orientée services et de microservices dans la mesure du possible. Les charges de travail capables d'absence d'état sont davantage en mesure d'être déployées en tant que microservices.

Résultat souhaité : Les charges de travail doivent être supportables, évolutives et aussi faiblement couplées que possible.

Lorsque vous choisissez comment segmenter votre charge de travail, comparez les avantages aux complexités. Ce qui convient pour un nouveau produit en course pour un premier lancement est différent de ce dont a besoin une charge de travail conçue pour augmenter d'échelle. Lors de la refactorisation d'une architecture monolithique existante, vous devez évaluer comment l'application prendra en charge une décomposition vers l'absence d'état. La division de services en microservices permet aux petites équipes bien définies de les développer et les gérer. Toutefois, les services plus petits peuvent créer des complexités dont une latence supérieure, un débogage plus complexe et une charge opérationnelle accrue.

Anti-modèles courants :

- La version microservice Death Star est une situation dans laquelle les composants atomiques deviennent si interdépendants que l'échec de l'un d'entre eux résulte en un échec encore plus important, ce qui rend les composants aussi rigides et fragiles qu'une architecture monolithique.

Avantages liés au respect de cette pratique :

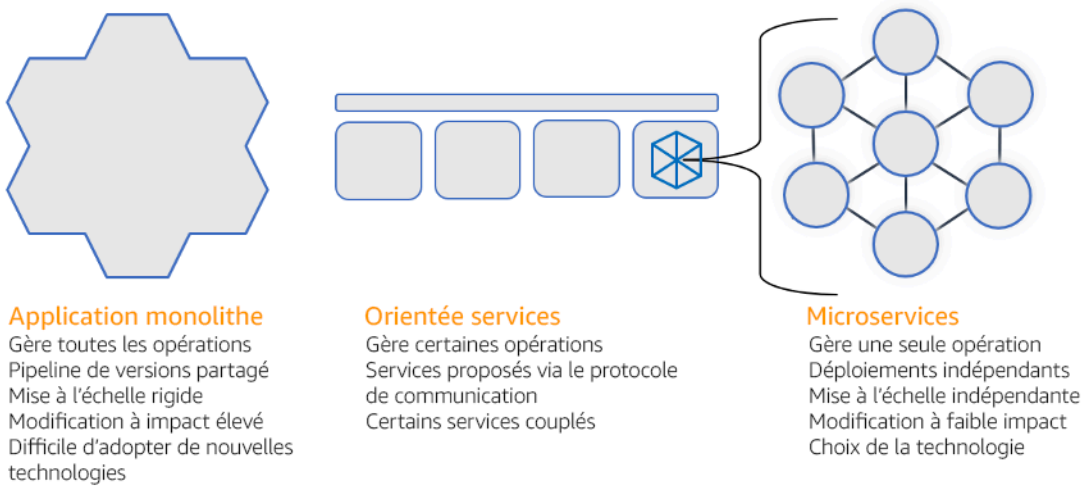
- L'utilisation de segments plus petits permet une plus grande agilité, une plus grande flexibilité organisationnelle et une évolutivité.
- L'impact réduit des interruptions de service.
- Les composants de l'application peuvent avoir différentes exigences de disponibilité, pouvant être pris en charge par une segmentation plus atomique.
- Des responsabilités bien définies pour les équipes prenant en charge la charge de travail.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Choisissez votre type d'architecture en fonction de la façon dont vous segmenterez votre charge de travail. Choisissez une architecture orientée service (SOA) ou une architecture de microservices (ou, dans de rares cas, une architecture monolithique). Même si vous choisissez de commencer avec une architecture monolithe, vous devez vous assurer qu'elle est modulaire et peut évoluer vers une SOA ou vers des microservices à mesure que votre produit se développe avec son adoption par les utilisateurs. Une SOA et une architecture de microservices offrent une segmentation plus petite. Si elle est préférable en tant qu'architecture moderne évolutive et fiable, il faut prendre en compte des compromis, notamment lors du déploiement d'une architecture de microservices.

Le principal compromis est que vous avez maintenant une architecture pour le calcul distribué qui peut compliquer le respect des exigences en matière de latence des utilisateurs et qui complexifie le suivi et le débogage des interactions des utilisateurs. AWS X-Ray peut vous aider à résoudre ce problème. Un autre effet à prendre en compte est la hausse de la complexité opérationnelle à mesure que vous augmentez le nombre d'applications que vous gérez, ce qui nécessite le déploiement de plusieurs composants indépendants.



Architectures monolithique, orientée services et de microservices

Étapes d'implémentation

- Déterminer l'architecture adaptée pour refactoriser ou créer votre application. SOA et les microservices offrent respectivement une segmentation plus petite, ce qui est préférable pour une architecture moderne évolutive et fiable. SOA peut constituer un bon compromis pour parvenir à une segmentation plus réduite tout en évitant certaines des complexités des microservices. Pour en savoir plus, voir Compromis des microservices.
- Si votre charge de travail est appropriée et que votre organisation peut la prendre en charge, vous devez utiliser une architecture de microservices pour obtenir la meilleure agilité et la meilleure fiabilité. Pour en savoir plus, voir Implémentation des microservices sur AWS.
- Tenir compte du modèle Figuier étrangleur pour refactoriser une architecture monolithique en composants plus petits. Cela implique de remplacer petit à petit des composants d'une application spécifique par de nouveaux services et applications. AWS Migration Hub Refactor Spaces agit comme le point de départ de la refactorisation incrémentielle. Pour en savoir plus, voir Migrer sans interruption vers des charges de travail existantes sur site à l'aide d'un modèle Figuier étrangleur.
- L'implémentation d'une architecture de microservices peut exiger un mécanisme de découverte de service pour permettre à ces services distribués de communiquer entre eux. AWS App Mesh peut être utilisé avec des architectures orientées service afin de fournir une découverte et un accès fiables aux services. AWS Cloud Map peut également être utilisé pour la découverte dynamique de service basée sur un DNS.

- Si vous migrez d'une architecture monolithique vers une architecture orientée service, Amazon MQ peut combler le fossé en tant que bus de services lors de la reconception des applications existantes dans le cloud.
- Pour les architectures monolithiques existantes avec une base de données partagée unique, choisissez comment réorganiser les données en segments plus petits. Vous pouvez les réorganiser par unité commerciale, modèle d'accès ou structure de données. À ce stade du processus de refactorisation, vous devez choisir d'utiliser un type de base de données relationnelle ou non relationnelle. Pour en savoir plus, voir De SQL à NoSQL.

Niveau d'effort du plan d'implémentation : Élevé

Ressources

Bonnes pratiques associées :

- REL03-BP02 Créer des services axés sur des domaines d'activité et la fonctionnalité

Documents connexes :

- Amazon API Gateway : configuration d'une API REST à l'aide d'OpenAPI
- Qu'est-ce que l'architecture orientée service ?
- Contexte délimité (modèle central dans la conception pilotée par domaine)
- Implémentation des microservices sur AWS
- Compromis des microservices
- Microservices : une définition de ce nouveau terme architectural
- Microservices sur AWS
- Qu'est-ce qu'AWS App Mesh ?

Exemples connexes :

- Atelier sur la modernisation itérative des applications

Vidéos connexes :

- Offrir l'excellence avec l'architecture de microservices sur AWS

REL03-BP02 Créer des services axés sur des domaines d'activité et la fonctionnalité

Une architecture orientée services (SOA) définit des services avec des fonctions bien déterminées dictées par les besoins métier. Les microservices utilisent des modèles de domaine et un contexte limité pour définir les limites des services en fonction des limites du contexte métier. En se concentrant sur les domaines d'activité et les fonctionnalités, les équipes peuvent définir des exigences de fiabilité indépendantes pour leurs services. Les contextes limités isolent et encapsulent la logique métier, ce qui permet aux équipes de mieux raisonner sur la manière de gérer les défaillances.

Résultat souhaité : les ingénieurs et les parties prenantes de l'entreprise définissent conjointement des contextes délimités et les utilisent pour concevoir des systèmes en tant que services remplissant des fonctions commerciales spécifiques. Ces équipes utilisent des pratiques établies telles que l'event storming pour définir les exigences. Les nouvelles applications sont conçues comme des services dont les limites sont bien définies et qui possèdent un couplage faible. Les monolithes existants sont décomposés en contextes limités et la conception des systèmes évolue vers des architectures SOA ou de microservices. Lorsque les monolithes sont refactorisés, des approches établies telles que les contextes de bulles et les modèles de décomposition des monolithes sont appliquées.

Les services orientés domaine sont exécutés sous la forme d'un ou de plusieurs processus qui ne partagent pas d'état. Ils répondent de manière indépendante aux fluctuations de la demande et gèrent les scénarios de panne à la lumière des exigences spécifiques du domaine.

Anti-modèles courants :

- Les équipes sont constituées autour de domaines techniques spécifiques tels que l'interface utilisateur et l'expérience utilisateur, les intergiciels ou les bases de données plutôt que de domaines commerciaux spécifiques.
- Les applications couvrent des responsabilités de domaine. Les services qui couvrent des contextes limités peuvent être plus difficiles à gérer, nécessiter des efforts de test plus importants et nécessiter la participation de plusieurs équipes de domaine aux mises à jour logicielles.
- Les dépendances de domaine, telles que les bibliothèques d'entités de domaine, sont partagées entre les services de telle sorte que les modifications apportées à un domaine de service nécessitent des modifications apportées à d'autres domaines de service.

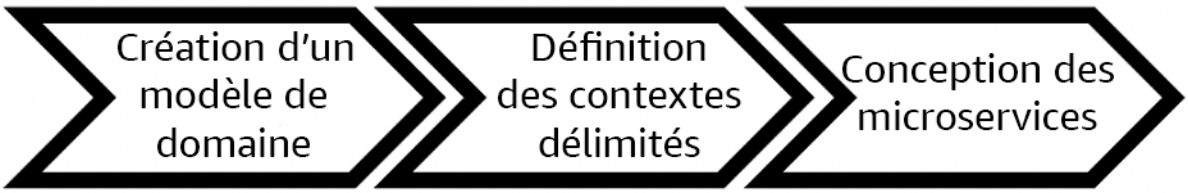
- Les contrats de service et la logique métier n'expriment pas les entités dans un langage de domaine commun et cohérent, ce qui crée des couches de traduction qui compliquent les systèmes et augmentent les efforts de débogage.

Avantages liés au respect de cette bonne pratique : Les applications sont conçues comme des services indépendants délimités par domaines d'activité et utilisent un langage métier commun. Les services peuvent être testés et déployés indépendamment. Les services répondent aux exigences de résilience spécifiques au domaine mis en œuvre.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

La décision pilotée par domaine (DDD) est l'approche fondamentale de la conception et de la création de logiciels autour de domaines métier. Il est utile de travailler avec un cadre existant lorsque vous créez des services axés sur des domaines métier. Lorsque vous travaillez avec des applications monolithiques existantes, vous pouvez tirer parti des modèles de décomposition qui fournissent des techniques éprouvées pour moderniser les applications en services.



Décision pilotée par domaine

Étapes d'implémentation

- Les équipes peuvent organiser des ateliers d'event storming pour identifier rapidement les événements, les commandes, les agrégats et les domaines dans un format léger de notes autocollantes.
- Une fois que les entités et les fonctions de domaine ont été créées dans un contexte de domaine, vous pouvez diviser votre domaine en services en utilisant un contexte limitédans lequel les entités qui partagent des fonctions et des attributs similaires sont regroupées. La division en contextes permet de faire émerger un modèle de délimitation des microservices.
  - Par exemple, les entités du site Amazon.com peuvent inclure le colis, la livraison, le calendrier, le prix, la remise et la devise.

    - Le colis, la livraison et le calendrier sont regroupés dans le contexte d'expédition, tandis que le prix, la remise et la devise sont regroupés dans le contexte de tarification.
- La décomposition des monolithes en microservices décrit les modèles de refactorisation des microservices. L'utilisation de modèles de décomposition par capacité métier, sous-domaine ou transaction s'inscrit parfaitement dans les approches axées sur le domaine.
- Des techniques tactiques telles que le contexte de bulles vous permettent d'introduire le DDD dans des applications existantes ou héritées sans devoir procéder à des réécritures préalables et sans engagement total envers le DDD. Dans une approche de contexte à bulles, un petit contexte limité est établi à l'aide d'un mappage et d'une coordination des services, ou d'une couche anticorruption, qui protège le modèle de domaine nouvellement défini des influences extérieures.

Une fois que les équipes ont effectué une analyse du domaine et défini des entités et des contrats de service, elles peuvent tirer parti des services AWS pour mettre en œuvre leur conception axée sur le domaine sous forme de services basés sur le cloud.

- Commencez votre développement en définissant des tests qui appliquent les règles métier de votre domaine. Le développement piloté par les tests (TDD) et le développement piloté par le comportement (BDD) aident les équipes à concentrer leurs services sur la résolution des problèmes commerciaux.
- Sélectionnez les services AWS qui répondent le mieux aux exigences de votre domaine d'activité et à l'architecture de microservices :
    - AWS sans serveur permet à votre équipe de se concentrer sur une logique de domaine spécifique au lieu de gérer les serveurs et l'infrastructure.
    - Les conteneurs AWS simplifient la gestion de votre infrastructure afin que vous puissiez vous concentrer sur les exigences de votre domaine.
    - Les bases de données sur mesure vous permettent d'adapter les exigences de votre domaine au type de base de données le mieux adapté.
- Création d'architectures hexagonales sur AWS décrit un cadre permettant d'intégrer une logique métier à des services en procédant de manière rétroactive à partir d'un domaine métier afin de répondre à des exigences fonctionnelles, puis d'associer des adaptateurs d'intégration. Les modèles qui séparent les détails de l'interface de la logique métier avec les services AWS aident les équipes à se concentrer sur les fonctionnalités du domaine et à améliorer la qualité des logiciels.

Ressources

Bonnes pratiques associées :

- REL03-BP01 Choisir comment segmenter votre charge de travail
- REL03-BP03 Fournir des contrats de service par API

Documents connexes :

- Microservices AWS
- Implémentation des microservices sur AWS
- Comment convertir un monolithe en microservices
- Premiers pas avec DDD en présence de systèmes hérités
- Conception axée sur le domaine : aborder la complexité au cœur du logiciel
- Création d'architectures hexagonales sur AWS
- Décomposition des monolithes en microservices
- Event Storming
- Messages entre contextes limités
- Microservices
- Développement piloté par les tests
- Développement axé sur le comportement

Exemples connexes :

- Atelier natif cloud en entreprise
- Conception de microservices natifs cloud sur AWS (extrait de DDD/EventStormingWorkshop)

Outils associés :

- Bases de données AWS Cloud
- Sans serveur sur AWS
- Conteneurs AWS

REL03-BP03 Fournir des contrats de service par API

Les contrats de service sont des accords documentés entre les producteurs d'API et les consommateurs définis dans une définition d'API lisible par machine. Une stratégie de gestion des versions permet aux consommateurs de continuer à utiliser l'API existante et de procéder à la migration de leurs applications vers la nouvelle API lorsqu'ils sont prêts. Le déploiement du producteur peut avoir lieu à tout moment tant que le contrat est respecté. Les équipes de service peuvent utiliser la pile technologique de leur choix pour satisfaire aux clauses du contrat d'API.

Résultat souhaité :

Anti-modèles courants : Les applications créées à l'aide d'architectures orientées services ou microservices peuvent fonctionner de manière indépendante tout en bénéficiant d'une dépendance intégrée à l'exécution. Les modifications apportées à un consommateur ou à un producteur d'API n'interrompent pas la stabilité de l'ensemble du système lorsque les deux parties respectent un contrat d'API commun. Les composants qui communiquent via des API de service peuvent exécuter des versions fonctionnelles indépendantes, effectuer des mises à niveau vers des dépendances d'exécution ou effectuer un basculement vers un site de reprise après sinistre (DR) avec peu ou pas d'impact mutuel. En outre, les services discrets sont capables d'évoluer de manière indépendante en absorbant la demande de ressources sans que les autres services évoluent à l'unisson.

- Création d'API de service sans schémas fortement typés. Cela se traduit par des API qui ne peuvent pas être utilisées pour générer des liaisons d'API et des charges utiles qui ne peuvent pas être validées par programmation.
- Absence d'adoption d'une stratégie de gestion des versions, qui oblige les utilisateurs d'API à les mettre à jour et à les publier, sous peine de défaillance lorsque les contrats de service évoluent.
- Messages d'erreur qui divulguent les détails de l'implémentation du service sous-jacent au lieu de décrire les échecs d'intégration dans le contexte et le langage du domaine.
- Absence d'utilisation des contrats d'API pour développer des scénarios de test et simuler des implémentations d'API afin de permettre des tests indépendants des composants du service.

Avantages liés au respect de cette bonne pratique : les systèmes distribués constitués de composants qui communiquent via des contrats de service d'API peuvent améliorer la fiabilité. Les développeurs peuvent détecter les problèmes potentiels au début du processus de développement en vérifiant les types lors de la compilation afin de s'assurer que les demandes et les réponses respectent le contrat d'API et que les champs obligatoires sont présents. Les contrats d'API

fournissent une interface d'autodocumentation claire pour les API et assurent une meilleure interopérabilité entre les différents systèmes et langages de programmation.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyen

Directives d'implémentation

Une fois que vous avez identifié les domaines d'activité et déterminé la segmentation de votre charge de travail, vous pouvez développer vos API de service. Définissez d'abord des contrats de service lisibles par machine pour les API, puis mettez en œuvre une stratégie de gestion des versions des API. Lorsque vous êtes prêt à intégrer des services via des protocoles courants tels que REST, GraphQL ou des événements asynchrones, vous pouvez intégrer des services AWS à votre architecture pour intégrer vos composants à l'aide de contrats d'API clairement typés.

Services AWS pour les contrats d'API de service

Intégrez des services AWS, notamment Amazon API Gateway, AWS AppSyncet Amazon EventBridge dans votre architecture pour utiliser les contrats de service d'API dans votre application. Amazon API Gateway vous aide à intégrer directement des services AWS natifs et d'autres services Web. API Gateway prend en charge la spécification et la gestion des versions OpenAPI. AWS AppSyncest un point de terminaison GraphQL géré que vous configurez en définissant un schéma GraphQL pour définir une interface de service pour les requêtes, les mutations et les abonnements. Amazon EventBridge utilise des schémas d'événements pour définir des événements et générer des liaisons de code pour vos événements.

Étapes d'implémentation

- Tout d'abord, définissez un contrat pour votre API. Un contrat exprimera les fonctionnalités d'une API et définira des objets de données et des champs fortement typés pour l'entrée et la sortie de l'API.
- Lorsque vous configurez des API dans API Gateway, vous pouvez importer et exporter les spécifications OpenAPI pour vos points de terminaison.
  - L'importation d'une définition OpenAPI simplifie la création de votre API et peut être intégrée à l'infrastructure AWS sous forme d'outils de code tels qu' AWS Serverless Application Model et AWS Cloud Development Kit (AWS CDK).
  - L'exportation d'une définition d'API simplifie l'intégration aux outils de test d'API et fournit aux consommateurs de services une spécification d'intégration.

- Vous pouvez définir et gérer les API GraphQL avec AWS AppSync en définissant un fichier de schéma GraphQL afin de générer votre interface de contrat et de simplifier l'interaction avec des modèles REST complexes, plusieurs tables de base de données ou des services existants.
- Les projets AWS Amplify intégrés à AWS AppSync génèrent des fichiers de requête JavaScript fortement typés à utiliser dans votre application ainsi qu'une bibliothèque cliente AWS AppSync GraphQL pour les tables Amazon DynamoDB .
- Lorsque vous consommez des événements de service provenant d'Amazon EventBridge, les événements adhèrent à des schémas qui existent déjà dans le registre des schémas ou que vous définissez à l'aide de la spécification OpenAPI. Avec un schéma défini dans le registre, vous pouvez également générer des liaisons client à partir du contrat de schéma afin d'intégrer votre code aux événements.
- Extension ou gestion des versions de votre API. L'extension d'une API est une option plus simple lorsque vous ajoutez des champs qui peuvent être configurés avec des champs facultatifs ou des valeurs par défaut pour les champs obligatoires.
  - Les contrats basés sur JSON pour des protocoles tels que REST et GraphQL peuvent être une bonne solution pour l'extension de contrat.
  - Les contrats basés sur XML pour des protocoles tels que SOAP doivent être testés auprès des consommateurs de services afin de déterminer la faisabilité d'une extension du contrat.
- Lors de la gestion des versions d'une API, pensez à implémenter la gestion des versions par proxy lorsqu'une façade est utilisée pour prendre en charge les versions afin que la logique puisse être maintenue dans une base de code unique.
  - Avec API Gateway vous pouvez utiliser les mappages de demandes et de réponses afin de simplifier l'absorption des modifications du contrat en établissant une façade permettant de fournir des valeurs par défaut pour les nouveaux champs ou de supprimer les champs retirés d'une demande ou d'une réponse. Avec cette approche, le service sous-jacent peut gérer une base de code unique.

Ressources

Bonnes pratiques associées :

- REL03-BP01 Choisir comment segmenter votre charge de travail
- REL03-BP02 Créer des services axés sur des domaines d'activité et la fonctionnalité
- REL04-BP02 Implémenter des dépendances couplées faiblement
- REL05-BP03 Contrôler et limiter les appels de nouvelle tentative

- REL05-BP05 Définir les délais d'attente du client

Documents connexes :

- Qu'est-ce qu'une API (interface de programmation d'applications) ?
- Implémentation des microservices sur AWS
- Compromis des microservices
- Microservices : une définition de ce nouveau terme architectural
- Microservices sur AWS
- Utilisation des extensions API Gateway pour OpenAPI
- Spécification OpenAPI
- GraphQL : schémas et types
- Liaisons de code Amazon EventBridge

Exemples connexes :

- Amazon API Gateway : configuration d'une API REST à l'aide d'OpenAPI
- Amazon API Gateway vers une application Amazon DynamoDB CRUD à l'aide d'OpenAPI
- Modèles modernes d'intégration des applications à l'ère de l'informatique sans serveur : intégration des services API Gateway
- Implémentation de la gestion des versions API Gateway basée sur les en-têtes avec Amazon CloudFront
- AWS AppSync : création d'une application client

Vidéos connexes :

- Using OpenAPI in AWS SAM to manage API Gateway

Outils associés :

- Amazon API Gateway
- AWS AppSync
- Amazon EventBridge

Concevoir des interactions dans un système distribué pour éviter les défaillances

Les systèmes distribués s'appuient sur des réseaux de communication pour interconnecter des composants comme des serveurs ou des services. Votre charge de travail doit fonctionner de manière fiable malgré la perte de données ou la latence dans ces réseaux. Les composants du système distribué doivent fonctionner d'une manière qui n'a pas d'impact négatif sur les autres composants ou la charge de travail. Ces bonnes pratiques empêchent les défaillances et améliorent le temps moyen entre les défaillances (MTBF).

Bonnes pratiques

- REL04-BP01 Identifier le type de systèmes distribués dont vous dépendez
- REL04-BP02 Implémenter des dépendances couplées faiblement
- REL04-BP03 Effectuer un travail constant
- REL04-BP04 Rendre toutes les réponses idempotentes

REL04-BP01 Identifier le type de systèmes distribués dont vous dépendez

Les systèmes distribués peuvent être synchrones, asynchrones ou par lots. Les systèmes synchrones doivent traiter les demandes le plus rapidement possible et communiquer entre eux en effectuant des appels de demande et de réponse synchrones à l’aide des protocoles HTTP/S, REST ou RPC (Remote Procedure Call). Les systèmes asynchrones communiquent entre eux en échangeant des données de manière asynchrone via un service intermédiaire sans coupler des systèmes individuels. Les systèmes par lots reçoivent un volume important de données d’entrée, exécutent des processus de données automatisés sans intervention humaine et génèrent des données de sortie.

Résultat souhaité : concevez une charge de travail qui interagit efficacement avec les dépendances synchrones, asynchrones et par lots.

Anti-modèles courants :

- La charge de travail attend indéfiniment une réponse de la part de ses dépendances, ce qui peut entraîner une expiration des clients de la charge de travail, qui ne savent pas si leur demande a été reçue.

- La charge de travail utilise une chaîne de systèmes dépendants qui s'appellent les uns les autres de manière synchrone. Cela nécessite que chaque système soit disponible et traite correctement une demande avant que l'ensemble de la chaîne puisse aboutir, ce qui entraîne un comportement et une disponibilité globale potentiellement fragiles.
- La charge de travail communique avec ses dépendances de manière asynchrone et repose sur le concept de livraison garantie unique des messages, alors qu'il est souvent encore possible de recevoir des messages dupliqués.
- La charge de travail n'utilise pas les outils de planification par lots appropriés et permet l'exécution simultanée de la même tâche de traitement par lots.

Avantages de la mise en place de cette bonne pratique : il est courant pour une charge de travail donnée de mettre en œuvre un ou plusieurs styles de communication (synchrone, asynchrone et par lots). Cette bonne pratique vous aide à identifier les différents compromis associés à chaque style de communication afin que votre charge de travail soit capable de tolérer les interruptions liées à toutes ses dépendances.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

Les sections suivantes contiennent des instructions de mise en œuvre générales et spécifiques pour chaque type de dépendance.

Instructions générales

- Assurez-vous que les objectifs de niveau de service (SLO) offerts vos dépendances en matière de performance et de fiabilité répondent aux exigences de performance et de fiabilité de votre charge de travail.
- Utilisez les services d'observabilité AWS pour surveiller les temps de réponse et les taux d'erreur afin de vous assurer que votre dépendance fournit le service aux niveaux requis par votre charge de travail.
- Identifiez les défis potentiels auxquels votre charge de travail peut être confrontée lors de la communication avec ses dépendances. Les systèmes distribués présentent un large éventail de défis susceptibles d'accroître la complexité architecturale, la charge opérationnelle et les coûts. Les défis courants incluent la latence, les perturbations du réseau, la perte de données, la mise à l'échelle et le retard de réplication des données.

- Mettez en œuvre une gestion des erreurs et une journalisation robustes pour faciliter la résolution des problèmes lorsque votre dépendance rencontre des problèmes.

Dépendance synchrone

Dans les communications synchrones, votre charge de travail envoie une demande à sa dépendance et bloque l'opération en attente de réponse. Lorsque sa dépendance reçoit la demande, elle essaie de la traiter le plus rapidement possible et renvoie une réponse à la charge de travail. L'un des principaux défis liés à la communication synchrone est qu'elle entraîne un couplage temporel, ce qui nécessite que la charge de travail et ses dépendances soient disponibles en même temps. Lorsque la charge de travail doit communiquer de manière synchrone avec ses dépendances, suivez les conseils ci-dessous :

- Votre charge de travail ne doit pas reposer sur plusieurs dépendances synchrones pour exécuter une seule fonction. Cette chaîne de dépendances augmente la fragilité globale, car toutes les dépendances sur le chemin doivent être disponibles pour que la demande soit traitée correctement.
- Lorsqu'une dépendance n'est pas saine ou n'est pas disponible, déterminez vos stratégies de gestion des erreurs et de nouvelles tentatives. Évitez d'utiliser un comportement bimodal. On parle de comportement bimodal lorsque la charge de travail présente un comportement différent en mode normal et en mode d'échec. Pour plus d'informations sur le comportement bimodal, consultez REL11-BP05 Utiliser la stabilité statique pour éviter les comportements bimodaux.
- N'oubliez pas qu'il vaut mieux échouer rapidement que faire attendre la charge de travail. Par exemple, le Guide du développeur AWS Lambda décrit comment gérer les nouvelles tentatives et les échecs lorsque vous invoquez des fonctions Lambda.
- Définissez des délais d'expiration lorsque la charge de travail appelle sa dépendance. Cette technique permet d'éviter d'attendre trop longtemps ou d'attendre indéfiniment une réponse. Pour une discussion utile sur ce problème, consultez la section Réglage des paramètres de demande HTTP du kit AWS Java SDK pour les applications Amazon DynamoDB prenant en charge la latence.
- Réduisez le nombre d'appels passés entre la charge de travail et sa dépendance pour répondre à une seule demande. Le fait d'avoir trop d'appels entre elles augmente le couplage et la latence.

Dépendance asynchrone

Pour pouvoir découpler temporellement la charge de travail de sa dépendance, elles doivent communiquer de manière asynchrone. En utilisant une approche asynchrone, la charge de travail

peut poursuivre tout autre traitement sans avoir à attendre que sa dépendance, ou sa chaîne de dépendances, envoie une réponse.

Lorsque la charge de travail doit communiquer de manière asynchrone avec sa dépendance, suivez les conseils ci-dessous :

- Déterminez s'il convient d'utiliser la messagerie ou le streaming d'événements en fonction de votre cas d'utilisation et de vos exigences. La messagerie permet à votre charge de travail de communiquer avec ses dépendances en envoyant et en recevant des messages via un agent de messages. Le streaming d'événements permet à votre charge de travail et à ses dépendances d'utiliser un service de streaming pour publier et s'abonner à des événements, diffusés sous forme de flux de données continus, qui doivent être traités dès que possible.

- La messagerie et le streaming d'événements traitent les messages différemment. Vous devez donc faire un choix en fonction des éléments suivants :
  - Priorité des messages : les agents de messagerie peuvent traiter les messages prioritaires avant les messages normaux. Dans le cadre du streaming d'événements, tous les messages ont la même priorité.
  - Consommation des messages : les agents de messagerie s'assurent que les consommateurs reçoivent le message. Les consommateurs qui diffusent des événements doivent suivre le dernier message qu'ils ont lu.
  - Ordre des messages : avec la messagerie, la réception des messages dans l'ordre exact dans lequel ils ont été envoyés n'est pas garantie, sauf si vous utilisez une approche FIFO (premier entré, premier sorti). Le streaming d'événements préserve toujours l'ordre dans lequel les données ont été produites.
  - Suppression du message : avec la messagerie, le consommateur doit supprimer le message après l'avoir traité. Le service de streaming d'événements ajoute le message à un flux et y reste jusqu'à l'expiration de la période de conservation du message. Cette politique de suppression rend le streaming d'événements adapté à la rediffusion de messages.
- Définissez comment la charge de travail comprend que sa dépendance a mené à bien sa tâche. Par exemple, lorsque votre charge de travail appelle une fonction Lambda de manière asynchrone, Lambda place l'événement dans une file d'attente et renvoie une réponse positive sans informations supplémentaires. Une fois le traitement terminé, la fonction Lambda peut envoyer le résultat vers une destination, configurable en fonction du succès ou de l'échec.
- Augmentez votre charge de travail pour gérer les messages dupliqués en tirant parti de l'idempotence. L'idempotence signifie que les résultats de la charge de travail ne changent pas

même si elle est générée plusieurs fois pour le même message. Il est important de souligner que les services de messagerie ou de streaming rediffusent un message en cas de panne du réseau ou si aucun accusé de réception n'a été reçu.

- Si la charge de travail n'obtient pas de réponse de sa dépendance, elle doit soumettre à nouveau la demande. Envisagez de limiter le nombre de tentatives pour préserver le processeur, la mémoire et les ressources réseau de votre charge de travail afin de gérer d'autres demandes. La documentation AWS Lambda montre comment gérer les erreurs liées à l'invocation asynchrone.
- Tirez parti des outils d'observabilité, de débogage et de suivi appropriés pour gérer et exploiter la communication asynchrone de la charge de travail avec ses dépendances. Vous pouvez l'utiliser Amazon CloudWatch pour surveiller les services de messagerie et de streaming d'événements. Vous pouvez également instrumenter votre charge de travail AWS X-Ray pour obtenir rapidement des informations utiles afin de résoudre les problèmes.

Dépendance par lots

Les systèmes par lots prennent les données d'entrée, lancent une série de tâches pour les traiter et produisent certaines données de sortie, sans intervention manuelle. En fonction de la taille des données, les tâches peuvent s'exécuter pendant une durée allant de quelques minutes à, dans certains cas, plusieurs jours. Lorsque la charge de travail communique avec sa dépendance par lots, suivez les conseils ci-dessous :

- Définissez la fenêtre de temps pendant laquelle la charge de travail doit exécuter le traitement par lots. La charge de travail peut configurer un modèle de récurrence pour invoquer un système de traitement par lots (par exemple, toutes les heures ou à la fin de chaque mois).
- Déterminez l'emplacement de l'entrée des données et de la sortie des données traitées. Choisissez un service de stockage, comme Amazon Simple Storage Services (Amazon S3), Amazon Elastic File System (Amazon EFS), et Amazon FSx for Lustre, qui permet à votre charge de travail de lire et d'écrire des fichiers à grande échelle.
- Si la charge de travail doit invoquer plusieurs tâches par lots, vous pouvez utiliser AWS Step Functions pour simplifier l'orchestration des tâches par lots exécutées sur AWS ou sur site. Cet exemple de projet montre l'orchestration de tâches par lots à l'aide de Step Functions, AWS Batch et Lambda.
- Surveillez les tâches par lots pour détecter d'éventuelles anomalies, telles qu'une tâche qui prend plus de temps que prévu. Vous pouvez utiliser des outils comme CloudWatch Container Insights pour surveiller les environnements et les tâches AWS Batch. Dans ce cas, la charge de travail empêcherait le début de la tâche suivante et informerait le personnel concerné de l'exception.

Ressources

Documents connexes :

- AWS Cloud Operations: Monitoring and Observability
- Amazon’s Builder Library : défis liés aux systèmes distribués
- REL11-BP05 Utiliser la stabilité statique pour éviter les comportements bimodaux
- Guide du développeur AWS Lambda : gestion des erreurs et nouvelles tentatives automatiques dans AWS Lambda
- Réglage des paramètres de demande HTTP du kit AWS Java SDK pour les applications Amazon DynamoDB prenant en charge la latence
- Messagerie AWS
- Qu’est-ce que le streaming de données ?
- Guide du développeur AWS Lambda : invocation asynchrone
- FAQ Amazon Simple Queue Service : files d’attente FIFO
- Guide du développeur Amazon Kinesis Data Streams : gestion des enregistrements dupliqués
- Guide du développeur Amazon Simple Queue Service : métriques CloudWatch disponibles pour Amazon SQS
- Guide du développeur Amazon Kinesis Data Streams : surveillance du service Amazon Kinesis Data Streams avec Amazon CloudWatch
- Guide du développeurAWS X-Ray : concepts AWS X-Ray
- Exemples AWSsur GitHub : AWS Step Functions Complex Orchestrator App
- Guide de l’utilisateurAWS Batch : AWS Batch CloudWatch Container Insights

Vidéos connexes :

- Sommet AWS SF 2022 : L’observabilité et la surveillance des applications avec AWS (COP310)

Outils associés :

- Amazon CloudWatch
- Amazon CloudWatch Logs
- AWS X-Ray
- Amazon Simple Storage Services (Amazon S3)

- Amazon Elastic File System (Amazon EFS)
- Amazon FSx for Lustre
- AWS Step Functions
- AWS Batch

REL04-BP02 Implémenter des dépendances couplées faiblement

Des dépendances telles que des systèmes de file d'attente, des systèmes de streaming, des flux de travail et des équilibreurs de charge sont couplées faiblement. Le couplage faible permet d'isoler le comportement d'un composant des autres composants qui en dépendent, ce qui augmente la résilience et l'agilité.

Dans les systèmes couplés fortement, la modification d'un composant peut nécessiter de modifier d'autres composants qui en dépendent, ce qui entraîne une dégradation des performances de tous les composants. Le couplage faible rompt cette dépendance de sorte que les composants dépendants n'ont besoin que de connaître l'interface publiée et sa version. La mise en œuvre d'un couplage faible entre les dépendances permet d'isoler une défaillance dans l'une afin de ne pas en impacter une autre.

Le couplage faible vous permet de modifier le code ou d'ajouter des fonctionnalités à un composant tout en minimisant les risques pour les autres composants qui en dépendent. Il offre également une résilience granulaire au niveau des composants, ce qui vous permet de mettre à l'échelle voire de modifier la mise en œuvre sous-jacente de la dépendance.

Pour améliorer encore la résilience par un couplage faible, dans la mesure du possible, rendez asynchrones les interactions des composants. Ce modèle convient à toute interaction qui ne nécessite pas une réponse immédiate et pour laquelle une confirmation de l'enregistrement d'une requête suffira. Il implique un composant qui génère des événements et un autre qui les consomme. Les deux composants ne s'intègrent pas via une interaction directe point à point, mais généralement via une couche de stockage durable intermédiaire, telle qu'une file d'attente Amazon SQS ou une plateforme de données de streaming comme Amazon Kinesis ou AWS Step Functions.

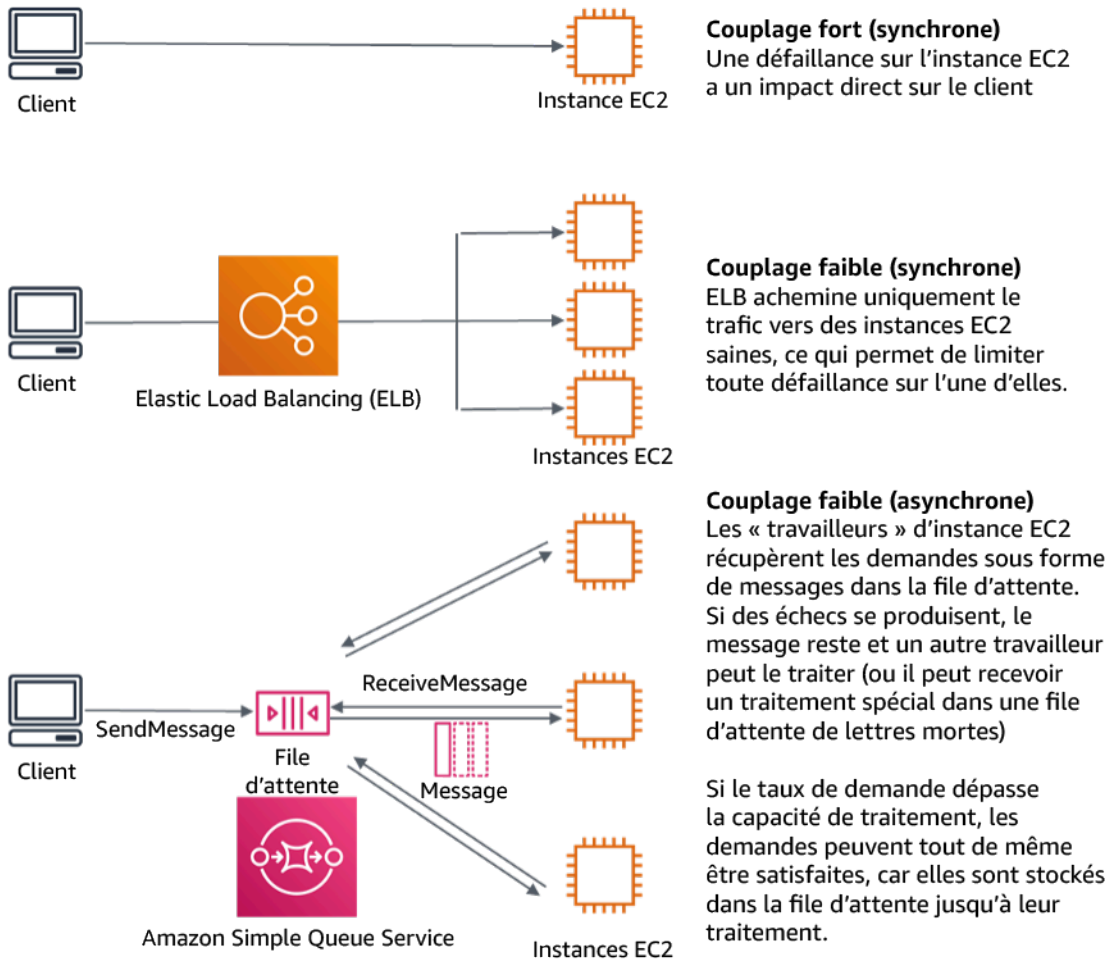


Figure 4 : Les dépendances telles que des systèmes de file d'attente et des équilibreurs de charge sont couplées faiblement

Les files d'attente Amazon SQS et les programmes Elastic Load Balancer ne sont que deux façons d'ajouter une couche intermédiaire pour un couplage faible. Les architectures guidées par les événements peuvent également être conçues dans le AWS Cloud à l'aide d'Amazon EventBridge, qui peut extraire des clients (producteurs d'événements) des services sur lesquels ils s'appuient (clients d'événements). Amazon Simple Notification Service (Amazon SNS) est une solution efficace lorsque vous avez besoin d'une messagerie de type « many-to-many », à haut débit et en mode push. Grâce aux rubriques Amazon SNS, vos systèmes d'édition peuvent diffuser des messages vers un grand nombre de points de terminaison abonnés pour un traitement parallèle.

Bien que les files d'attente offrent plusieurs avantages, dans la plupart des systèmes en temps réel stricts, les requêtes antérieures à un seuil (souvent en secondes) sont considérées comme obsolètes (le client a abandonné et n'attend plus de réponse). En conséquence, elles ne sont pas traitées. De

cette façon, les requêtes plus récentes (et probablement toujours valides) peuvent être traitées à la place.

Résultat souhaité : la mise en œuvre de dépendances couplées faiblement réduit la surface de défaillance au niveau des composants, ce qui permet de diagnostiquer et de résoudre les problèmes. Elle simplifie également les cycles de développement en permettant aux équipes de mettre en œuvre des modifications à un niveau modulaire sans affecter les performances des autres composants qui en dépendent. Avec cette approche, il est possible de mettre à l'échelle un composant en fonction des besoins en ressources et de l'utilisation de ce composant, ce qui contribue à améliorer la rentabilité.

Anti-modèles courants :

- Déploiement d'une charge de travail monolithique.
- Appel direct d'API entre les niveaux de charge de travail sans possibilité de basculement ou de traitement asynchrone de la demande.
- Couplage fort à l'aide de données partagées. Les systèmes couplés faiblement évitent de partager des données par le biais de bases de données partagées ou d'autres formes de stockage de données couplées fortement, qui peuvent réintroduire un couplage fort et entraver la capacité de mise à l'échelle.
- Ignorer la contre-pression. Votre charge de travail doit être capable de ralentir ou d'arrêter les données entrantes lorsqu'un composant ne peut pas les traiter au même rythme.

Avantages liés au respect de cette bonne pratique : le couplage faible permet d'isoler le comportement d'un composant de celui des autres composants qui en dépendent, et d'augmenter ainsi la résilience et l'agilité. La défaillance d'un composant est isolée des autres.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

Implémentez des dépendances couplées faiblement. Différentes solutions permettent de créer des applications couplées faiblement. Ces services incluent notamment la mise en œuvre de files d'attente entièrement gérées, des flux de travail automatisés, des réactions aux événements et des API, qui peuvent aider à isoler un composant des autres composants et, par conséquent, à accroître la résilience et l'agilité.

- Créer des architectures basées sur les événements : Amazon EventBridge vous aide à créer des architectures basées sur les événements, qui sont couplées faiblement et distribuées.
- Mettre en œuvre des files d'attente dans des systèmes distribués : vous pouvez utiliser Amazon Simple Queue Service (Amazon SQS) pour intégrer et découpler des systèmes distribués.
- Conteneuriser les composants en tant que microservices : les microservices permettent aux équipes de créer des applications composées de petits composants indépendants qui communiquent via des API bien définies. Amazon Elastic Container Service(Amazon ECS) et Amazon Elastic Kubernetes Service (Amazon EKS) peuvent vous aider à faire plus rapidement vos premiers pas avec les conteneurs.
- Gérer les flux de travail avec Step Functions : Step Functions vous aide à coordonner plusieurs services AWS dans des flux de travail flexibles.
- Tirer parti des architectures de messagerie pub/sub : Amazon Simple Notification Service (Amazon SNS) transmet les messages des diffuseurs de publication aux abonnés (également appelés producteurs et consommateurs).

Étapes d'implémentation

- Les composants d'une architecture basée sur les événements sont initiés par des événements. Les événements sont des actions qui se produisent dans un système (par exemple, un utilisateur ajoute un article à un panier). Lorsque l'action aboutit, un événement est généré et active le composant suivant du système.
  - Building Event-driven Applications with Amazon EventBridge
  - AWS re:Invent 2022 - Designing Event-Driven Integrations using Amazon EventBridge
- Les systèmes de messagerie distribuée comportent trois parties principales qui doivent être mises en œuvre pour une architecture basée sur des files d'attente : les composants du système distribué, la file d'attente utilisée pour le découplage (distribuée sur des serveurs Amazon SQS) et les messages de la file d'attente. Dans un système classique, les producteurs envoient le message dans la file d'attente et le consommateur reçoit le message de la file d'attente. La file d'attente stocke les messages sur plusieurs serveurs Amazon SQS à des fins de redondance.
  - Basic Amazon SQS architecture
  - Send Messages Between Distributed Applications with Amazon Simple Queue Service
- Lorsqu'ils sont bien utilisés, les microservices améliorent la maintenabilité et la capacité de mise à l'échelle, car les composants couplés faiblement sont gérés par des équipes indépendantes. Ils permettent également d'isoler les comportements d'un composant en cas de changement.

  - Implémentation des microservices sur AWS
  - Let's Architect! Architecting microservices with containers
- Avec AWS Step Functions, vous pouvez notamment créer des applications distribuées, automatiser des processus et orchestrer des microservices. L'orchestration de plusieurs composants dans un flux de travail automatisé vous permet de découpler des dépendances dans votre application.
  - Create a Serverless Workflow with AWS Step Functions and AWS Lambda
  - Mise en route avec AWS Step Functions

Ressources

Documents connexes :

- Amazon EC2: Ensuring Idempotency
- L'Amazon Builders' Library : défis liés aux systèmes distribués
- The Amazon Builders' Library: Reliability, constant work, and a good cup of coffee
- What Is Amazon EventBridge?
- What Is Amazon Simple Queue Service?
- Break up with your monolith
- Orchestrate Queue-based Microservices with AWS Step Functions and Amazon SQS
- Basic Amazon SQS architecture
- Architecture basée sur des files d'attente

Vidéos connexes :

- AWS New York Summit 2019: Intro to Event-driven Architectures and Amazon EventBridge (MAD205)
- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small ARC337 (inclut le couplage faible, le travail constant et la stabilité statique)
- AWS re:Invent 2019: Moving to event-driven architectures (SVS308)
- AWS re:Invent 2019: Scalable serverless event-driven applications using Amazon SQS and Lambda (API304)

- AWS re:Invent 2019: Scalable serverless event-driven applications using Amazon SQS and Lambda
- AWS re:Invent 2022 - Designing event-driven integrations using Amazon EventBridge
- AWS re:Invent 2017: Elastic Load Balancing Deep Dive and Best Practices

REL04-BP03 Effectuer un travail constant

Les systèmes peuvent échouer en cas de modifications importantes et rapides de la charge. Par exemple, si votre charge de travail effectue une surveillance de l'état de milliers de serveurs, elle doit envoyer chaque fois une charge utile de la même taille (un instantané complet de l'état actuel). Qu'aucun des serveurs ne présente de problème ou qu'ils en connaissent tous, le système de surveillance de l'état effectue un travail constant sans modifications importantes ni rapides.

Par exemple, si le système de vérification de l'état surveille 100 000 serveurs, la charge sur celui-ci est nominale sous le taux de défaillance normalement faible du serveur. En revanche, si un événement majeur rendait la moitié de ces serveurs défectueux, le système de vérification de l'état serait submergé en tentant de mettre à jour les systèmes de notification et de communiquer l'état à ses clients. Le système de vérification de l'état doit donc plutôt envoyer à chaque fois l'instantané complet de l'état actuel. 100 000 états d'intégrité du serveur, chacun représenté par un bit, ne seraient qu'une charge utile de 12,5 Ko. Qu'aucun des serveurs ne présente de problème ou qu'ils en connaissent tous, le système de vérification de l'état effectue un travail constant, et les modifications importantes et rapides ne menacent pas la stabilité du système. C'est ainsi qu'Amazon Route 53 gère les vérifications de l'état des points de terminaison (tels que les adresses IP) pour déterminer comment les utilisateurs finaux sont acheminés vers eux.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Faible

Directives d'implémentation

- Effectuer un travail constant : les systèmes peuvent échouer lorsque la charge connaît des changements rapides et importants.
- Implémentez des dépendances couplées faiblement. Des dépendances telles que des systèmes de file d'attente, des systèmes de streaming, des flux de travail et des équilibreurs de charge sont couplées faiblement. Le couplage faible permet d'isoler le comportement d'un composant des autres composants qui en dépendent, ce qui augmente la résilience et l'agilité.
  - L'Amazon Builders' Library : fiabilité, travail constant et une bonne tasse de café

- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small ARC337 (includes constant work)
  - Pour l'exemple d'un système de vérification de l'état surveillant 100 000 serveurs, concevez les charges de travail de manière à ce que les tailles de charge utile restent constantes, quel que soit le nombre de réussites ou d'échecs.

Ressources

Documents connexes :

- Amazon EC2 : garantir l'idempotence
- L'Amazon Builders' Library : défis liés aux systèmes distribués
- L'Amazon Builders' Library : fiabilité, travail constant et une bonne tasse de café

Vidéos connexes :

- AWS New York Summit 2019: Intro to Event-driven Architectures and Amazon EventBridge (MAD205)
- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small ARC337 (includes constant work)
- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small ARC337 (inclut un couplage faible, un travail constant et une stabilité statique)
- AWS re:Invent 2019: Moving to event-driven architectures (SVS308)

REL04-BP04 Rendre toutes les réponses idempotentes

Un service idempotent promet que chaque demande est traitée une seule fois et exactement de la même façon, de sorte que l'exécution de plusieurs demandes identiques ait le même effet qu'une seule demande. Un service idempotent permet à un client d'implémenter plus facilement les nouvelles tentatives sans craindre qu'une demande soit traitée plusieurs fois par erreur. Pour ce faire, les clients peuvent émettre des demandes d'API avec un jeton d'idempotence. Le même jeton est utilisé chaque fois que la demande est répétée. Une API de service idempotente utilise le jeton pour renvoyer une réponse identique à la réponse qui a été renvoyée la première fois que la demande a été traitée.

Dans un système distribué, il est facile d'effectuer une action au maximum une fois (le client n'effectue qu'une seule demande) ou au moins une fois (continuer à demander jusqu'à ce que le client reçoive la confirmation de la réussite). En revanche, il est difficile de garantir qu'une action est idempotente, c'est-à-dire exécutée une seule fois, de sorte que l'exécution de plusieurs demandes identiques a le même effet qu'une seule demande. En utilisant des jetons d'idempotence dans les API, les services peuvent recevoir une demande de mutation une ou plusieurs fois sans créer d'enregistrements dupliqués ou induire des effets secondaires.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

- Rendez toutes les réponses idempotentes. Un service idempotent promet que chaque demande est traitée une seule fois et exactement de la même façon, de sorte que l'exécution de plusieurs demandes identiques ait le même effet qu'une seule demande.
  - Les clients peuvent émettre des demandes d'API avec un jeton d'idempotence. Le même jeton est utilisé chaque fois que la demande est répétée. Une API de service idempotente utilise le jeton pour renvoyer une réponse identique à la réponse qui a été renvoyée la première fois que la demande a été traitée.
    - Garantir l'idempotence Amazon EC2

Ressources

Documents connexes :

- Garantir l'idempotence Amazon EC2
- L'Amazon Builders' Library : défis liés aux systèmes distribués
- L'Amazon Builders' Library : fiabilité, travail constant et une bonne tasse de café

Vidéos connexes :

- AWS New York Summit 2019: Intro to Event-driven Architectures and Amazon EventBridge (MAD205)
- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small ARC337 (inclut un couplage faible, un travail constant et une stabilité statique)
- AWS re:Invent 2019: Moving to event-driven architectures (SVS308)

Concevoir des interactions dans un système distribué pour résister aux défaillances ou les atténuer

Les systèmes distribués s'appuient sur des réseaux de communication pour interconnecter des composants (tels que des serveurs ou des services). Votre charge de travail doit fonctionner de manière fiable malgré la perte de données ou la latence sur ces réseaux. Les composants du système distribué doivent fonctionner d'une manière qui n'a pas d'impact négatif sur les autres composants ou la charge de travail. Ces bonnes pratiques permettent aux charges de travail de résister aux contraintes ou aux défaillances, de s'en remettre plus rapidement et d'atténuer l'impact de ces déficiences. Il en résulte une amélioration du temps moyen de récupération (MTTR).

Ces bonnes pratiques empêchent les défaillances et améliorent le temps moyen entre les défaillances (MTBF).

Bonnes pratiques

- REL05-BP01 Implémenter une dégradation appropriée pour transformer les dépendances matérielles applicables en dépendances logicielles
- REL05-BP02 Limiter les demandes
- REL05-BP03 Contrôler et limiter les appels de nouvelle tentative
- REL05-BP04 Procéder à une interruption immédiate et limiter les files d'attente
- REL05-BP05 Définir les délais d'attente du client
- REL05-BP06 Rendre les systèmes sans état dans la mesure du possible
- REL05-BP07 Mettre en place des leviers d'urgence

REL05-BP01 Implémenter une dégradation appropriée pour transformer les dépendances matérielles applicables en dépendances logicielles

Les composants de l'application doivent continuer à exécuter leur fonction principale même si les dépendances deviennent indisponibles. Ils peuvent fournir des données légèrement obsolètes, des données alternatives ou même aucune donnée. Cela garantit que le fonctionnement global du système n'est que très peu entravé par des défaillances localisées tout en fournissant une valeur commerciale centrale.

Résultat souhaité : Lorsque les dépendances d'un composant ne sont pas saines, le composant lui-même peut continuer de fonctionner, bien que de manière dégradée. Les modes de défaillance des

composants doivent être considérés comme un fonctionnement normal. Les flux de travail doivent être conçus de manière à ce que ces défaillances n'aboutissent pas à une défaillance complète ou qu'elles aboutissent au moins à des états prévisibles et récupérables.

Anti-modèles courants :

- Ne pas identifier les fonctionnalités métier essentielles nécessaires. Ne pas tester le fonctionnement des composants, même en cas de défaillance des dépendances.
- Aucune donnée n'est diffusée en cas d'erreur ou lorsqu'une seule dépendance parmi plusieurs n'est pas disponible et que des résultats partiels peuvent toujours être renvoyés.
- Création d'un état incohérent lorsqu'une transaction échoue partiellement.
- Ne pas disposer d'un autre moyen d'accéder à un magasin de paramètres central.
- Invalider ou vider l'état local à la suite d'un échec d'actualisation sans prendre en compte les conséquences d'une telle opération.

Avantages liés au respect de cette bonne pratique : une dégradation progressive améliore la disponibilité du système dans son ensemble et maintient la fonctionnalité des fonctions les plus importantes même en cas de panne.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

La mise en œuvre d'une dégradation progressive permet de minimiser l'impact des défaillances de dépendance sur le fonctionnement des composants. Idéalement, un composant détecte les défaillances liées aux dépendances et les contourne de manière à avoir un impact minimal sur les autres composants ou les clients.

L'architecture permettant une dégradation progressive implique de prendre en compte les modes de défaillance potentiels lors de la conception des dépendances. Pour chaque mode de défaillance, déterminez un moyen de fournir la plupart des fonctionnalités, ou les plus critiques d'entre elles, du composant aux appelants ou aux clients. Ces considérations peuvent devenir des exigences supplémentaires qui peuvent être testées et vérifiées. Idéalement, un composant est capable d'exécuter sa fonction principale de manière acceptable, même en cas de défaillance d'une ou de plusieurs dépendances.

Il s'agit tout autant d'une discussion commerciale que technique. Toutes les exigences commerciales sont importantes et doivent être satisfaites dans la mesure du possible. Cependant, il est tout de

même logique de se demander ce qui doit se passer lorsque toutes les exigences ne peuvent pas être satisfaites. Un système peut être conçu pour être disponible et cohérent, mais lorsqu'une exigence doit être supprimée, laquelle est la plus importante ? Pour le traitement des paiements, il peut s'agir de la cohérence. Pour une application en temps réel, il peut s'agir de la disponibilité. Pour un site Web orienté client, la réponse peut dépendre des attentes du client.

Ce que cela signifie dépend des exigences du composant et de ce qui doit être considéré comme sa fonction principale. Par exemple :

- un site Web d'e-commerce peut afficher des données provenant de plusieurs systèmes différents, par exemple des recommandations personnalisées, les produits les mieux classés et l'état des commandes des clients sur la page de destination. Lorsqu'un système en amont est défaillant, il est tout de même judicieux d'afficher tout le reste au lieu d'afficher une page d'erreur à un client.
- Un composant effectuant des écritures par lots peut toujours continuer à traiter un lot si l'une des opérations individuelles échoue. La mise en œuvre d'un mécanisme de nouvelle tentative doit être simple. Cela peut être fait en renvoyant à l'appelant des informations indiquant quelles opérations ont réussi, lesquelles ont échoué et pourquoi elles ont échoué, ou en plaçant les demandes ayant échoué dans une file d'attente de lettres mortes pour implémenter des tentatives asynchrones. Les informations relatives aux opérations ayant échoué doivent également être consignées.
- Un système qui traite les transactions doit vérifier que toutes les mises à jour individuelles sont exécutées ou qu'aucune d'entre elles ne l'est. Pour les transactions distribuées, le modèle Saga peut être utilisé pour annuler les opérations précédentes en cas d'échec d'une opération ultérieure de la même transaction. Ici, la fonction principale est de maintenir la cohérence.
- Les systèmes soumis à des contraintes de temps doivent être en mesure de gérer les dépendances qui ne répondent pas en temps voulu. Dans ces cas de figure, le modèle du disjoncteur peut être utilisé. Lorsque les réponses d'une dépendance commencent à expirer, le système peut passer à un état fermé où aucun appel supplémentaire n'est effectué.
- Une application peut lire des paramètres à partir d'un magasin de paramètres. Il peut être utile de créer des images de conteneur avec un ensemble de paramètres par défaut et de les utiliser si le magasin de paramètres n'est pas disponible.

Notez que les chemins empruntés en cas de défaillance d'un composant doivent être testés et doivent être nettement plus simples que le chemin principal. D'une manière générale, les stratégies de repli doivent être évitées.

Étapes d'implémentation

Identifiez les dépendances externes et internes. Déterminez quels types de défaillances peuvent y survenir. Réfléchissez à des moyens de minimiser l'impact négatif sur les systèmes en amont et en aval, ainsi que sur les clients lors de ces défaillances.

Vous trouverez ci-dessous une liste des dépendances et la manière de les dégrader de façon appropriée en cas d'échec :

1. Défaillance partielle des dépendances : un composant peut adresser plusieurs demandes à des systèmes en aval, soit sous la forme de demandes multiples adressées à un système, soit sous celle d'une demande adressée à plusieurs systèmes. Selon le contexte métier, différentes méthodes de gestion peuvent être appropriées (pour plus de détails, voir les exemples précédents dans le guide de mise en œuvre).
2. Un système en aval est incapable de traiter les demandes en raison d'une charge élevée : si les demandes adressées à un système en aval échouent régulièrement, il n'est pas logique de continuer à réessayer. Cela peut créer une charge supplémentaire sur un système déjà surchargé et rendre la récupération plus difficile. Le modèle du disjoncteur peut être utilisé ici afin de surveiller les appels en échec vers un système en aval. Si un grand nombre d'appels échouent, il cessera d'envoyer d'autres demandes au système en aval et n'autorisera les appels qu'occasionnellement pour vérifier si le système en aval est à nouveau disponible.
3. Un magasin de paramètres n'est pas disponible : pour transformer un magasin de paramètres, vous pouvez utiliser la mise en cache des dépendances souples ou des valeurs par défaut saines incluses dans les images de conteneur ou de machine. Notez que ces valeurs par défaut doivent être tenues à jour et incluses dans les suites de tests.
4. Aucun service de surveillance ou aucune autre dépendance non fonctionnelle n'est disponible : si un composant ne peut pas envoyer par intermittence des journaux, des métriques ou des traces à un service de surveillance central, il est souvent préférable de continuer à exécuter les fonctions métier comme d'habitude. Il est souvent inacceptable de ne pas enregistrer ni de pousser des métriques pendant une longue période. En outre, certains cas d'utilisation peuvent nécessiter des entrées d'audit complètes pour répondre aux exigences de conformité.
5. Une instance principale d'une base de données relationnelle n'est peut-être pas disponible : Amazon Relational Database Service, comme presque toutes les bases de données relationnelles, ne peut comporter qu'une seule instance d'écriture principale. Cela crée un point de défaillance unique pour les charges de travail d'écriture et complique la mise à l'échelle. Ce problème peut être partiellement atténué en utilisant une configuration multi-AZ pour une haute disponibilité ou Amazon Aurora sans serveur pour une meilleure évolutivité. Pour des exigences de très

haute disponibilité, il peut être judicieux de ne pas se fier du tout au rédacteur principal. Pour les requêtes qui se limitent à la lecture, des répliques de lecture peuvent être utilisées, ce qui assure la redondance et la possibilité d'une mise à l'échelle, et pas seulement d'une augmentation. Les écritures peuvent être mises en mémoire tampon, par exemple dans une file d'attente Amazon Simple Queue Service, afin que les demandes d'écriture des clients puissent toujours être acceptées même si le serveur principal est temporairement indisponible.

Ressources

Documents connexes :

- Amazon API Gateway : limiter les demandes d'API pour un meilleur débit
- Coupe-circuit (présentation du coupe-circuit, ouvrage « Release It! »)
- Nouvelles tentatives après erreur et interruptions exponentielles dans AWS
- Michael Nygard « Release It! Design and Deploy Production-Ready Software »
- L'Amazon Builders' Library : éviter le basculement dans les systèmes distribués
- L'Amazon Builders' Library : éviter les retards de file d'attente insurmontables
- L'Amazon Builders' Library : défis et stratégies de mise en cache
- L'Amazon Builders' Library : délais d'attente, nouvelles tentatives et backoff avec instabilité

Vidéos connexes :

- Retry, backoff, and jitter: AWS re:Invent 2019: Introducing The Amazon Builders' Library (DOP328)

Exemples connexes :

- Atelier Well-Architected : niveau 300 : implémentation de la surveillance de l'état et gestion des dépendances pour améliorer la fiabilité

REL05-BP02 Limiter les demandes

Limitez les demandes pour atténuer l'épuisement des ressources en cas d'augmentation imprévue de la demande. Les demandes inférieures à la limite sont traitées tandis que celles dépassant la limite définie sont rejetées et présentent un message de retour indiquant que la demande a dépassé la limite.

Résultat souhaité : les pics de volume importants, qu'ils soient dus à une augmentation soudaine du trafic client, à des inondations ou à des tempêtes de nouvelles tentatives, sont atténués par la limitation des demandes, ce qui permet aux charges de travail de poursuivre le traitement normal du volume de demandes pris en charge.

Anti-modèles courants :

- Les limitations des points de terminaison de l'API ne sont pas implémentées ou leurs valeurs par défaut sont conservées sans tenir compte des volumes attendus.
- Les points de terminaison de l'API ne sont pas testés en termes de charge ou les limites de régulation ne sont pas testées.
- Limiter les taux de demandes sans tenir compte de la taille ou de la complexité des demandes.
- Tester les taux de demande maximaux ou la taille maximale des demandes, mais pas les deux simultanément.
- Les ressources ne sont pas provisionnées selon les mêmes limites établies lors des tests.
- Aucun plan d'utilisation n'a été configuré ni envisagé pour les utilisateurs d'API d'application à application (A2A).
- Les utilisateurs de files d'attente qui redimensionnent horizontalement ne disposent pas de paramètres de simultanéité maximaux configurés.
- La limitation du débit par adresse IP n'a pas été mise en œuvre.

Avantages liés au respect de cette bonne pratique : Les charges de travail qui fixent des limites peuvent fonctionner normalement et traiter correctement le chargement des demandes acceptées en cas de pics de volume inattendus. Les pics soudains ou soutenus de demandes adressées aux API et aux files d'attente sont limités et n'épuisent pas les ressources de traitement des demandes. Les limites de débit limitent les requêtes individuelles afin que les volumes élevés de trafic provenant d'une seule adresse IP ou d'un seul utilisateur d'API n'épuisent pas les ressources et n'aient pas d'impact sur les autres consommateurs.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Les services doivent être conçus pour traiter une capacité connue de demandes ; cette capacité peut être établie par des tests de charge. Si les taux d'arrivée des demandes dépassent les limites, la réponse appropriée indique qu'une demande a été limitée. Cela permet au consommateur de gérer l'erreur et de réessayer ultérieurement.

Lorsque votre service nécessite une implémentation de limitation, pensez à implémenter l'algorithme du compartiment à jetons, dans lequel un jeton compte pour une demande. Les jetons sont rechargés à une vitesse limitée par seconde et vidés de manière asynchrone à raison d'un jeton par demande.

L'algorithme du compartiment à jetons.

Amazon API Gateway implémente l'algorithme du compartiment à jetons en fonction des limites du compte et de la région et il peut être configuré par client avec des plans d'utilisation. En outre, Amazon Simple Queue Service (Amazon SQS) et Amazon Kinesis peuvent mettre en mémoire tampon les demandes afin de lisser le taux de demandes et de permettre des taux de limitation plus élevés pour les demandes pouvant être traitées. Enfin, vous pouvez implémenter la limitation du débit avec AWS WAF afin de limiter les utilisateurs d'API spécifiques qui génèrent une charge anormalement élevée.

Étapes d'implémentation

Vous pouvez configurer API Gateway avec des limites de régulation pour vos API et retourner `des erreurs 429 Demandes trop nombreuses` lorsque les limites sont dépassées. Vous pouvez utiliser AWS WAF avec vos points de terminaison API Gateway et AWS AppSync pour activer la limitation du débit par adresse IP. En outre, lorsque votre système peut tolérer un traitement asynchrone, vous pouvez placer les messages dans une file d'attente ou un flux pour accélérer les réponses aux clients du service, ce qui vous permet d'atteindre des taux d'accélération plus élevés.

Avec le traitement asynchrone, lorsque vous avez configuré Amazon SQS en tant que source d'événements pourAWS Lambda, vous pouvez configurer la simultanéité maximale afin d'éviter que des taux d'événements élevés n'épuisent le quota d'exécution simultanée disponible du compte requis pour d'autres services de votre charge de travail ou de votre compte.

Bien qu'API Gateway propose une implémentation gérée du compartiment à jetons, lorsque vous ne pouvez pas utiliserAPI Gateway, vous pouvez tirer parti des implémentations open source spécifiques à la langue (voir les exemples associés dans Ressources) du compartiment à jetons pour vos services.

- Comprenez et configurez les limitations API Gateway au niveau du compte par région, de l'API par étape et de la clé d'API par niveau de plan d'utilisation.
- Appliquez les règles de limitation de débit AWS WAF vers les points de terminaison API Gateway et AWS AppSync pour vous protéger contre les inondations et bloquer les adresses IP malveillantes. Les règles de limitation de débit peuvent également être configurées sur les clés d'API AWS AppSync pour les consommateurs A2A.
- Déterminez si vous avez besoin d'un contrôle plus limitant qu'une limitation du débit pour les API AWS AppSync et, si c'est le cas, configurez un API Gateway devant votre point de terminaison AWS AppSync.
- Lorsque des files d'attente Amazon SQS sont configurées comme déclencheurs pour les utilisateurs de files d'attente Lambda, définissez la simultanéité maximale sur une valeur capable d'effectuer un traitement suffisant pour vous permettre d'atteindre vos objectifs de niveau de service sans pour autant dépasser les limites de simultanéité affectant les autres fonctions Lambda. Envisagez de définir une concurrence réservée pour d'autres fonctions Lambda du même compte et de la même région lorsque vous utilisez des files d'attente avec Lambda.
- Utilisez API Gateway avec des intégrations de services natives vers Amazon SQS ou Kinesis pour mettre des demandes en mémoire tampon.
- Si vous ne pouvez pas utiliser API Gateway, examinez les bibliothèques spécifiques à la langue pour implémenter l'algorithme de compartiment à jetons adapté à votre charge de travail. Consultez la section des exemples et faites vos propres recherches pour trouver une bibliothèque appropriée.
- Testez les limites que vous envisagez de définir ou d'autoriser à augmenter, et documentez les limites testées.
- N'augmentez pas les limites au-delà de ce que vous avez établi lors des tests. Lorsque vous augmentez une limite, vérifiez que les ressources provisionnées sont déjà équivalentes ou supérieures à celles des scénarios de test avant d'appliquer l'augmentation.

Ressources

Bonnes pratiques associées :

- REL04-BP03 Effectuer un travail constant

- REL05-BP03 Contrôler et limiter les appels de nouvelle tentative

Documents connexes :

- Amazon API Gateway : limiter les demandes d'API pour un meilleur débit
- AWS WAF : déclaration de règle basée sur le débit
- Introduction d'une simultanéité maximale de AWS Lambda en utilisant Amazon SQS comme source d'événements
- AWS Lambda : simultanéité maximale

Exemples connexes :

- Les trois principales règles AWS WAF basées sur le débit
- Bucket4j Java
- Jeton-compartiment Python
- Nœud jeton-compartiment
- Limitation du débit de threading du système .NET

Vidéos connexes :

- Implementing GraphQL API security best practices with AWS AppSync

Outils associés :

- Amazon API Gateway
- AWS AppSync
- Amazon SQS
- Amazon Kinesis
- AWS WAF

REL05-BP03 Contrôler et limiter les appels de nouvelle tentative

Utilisez le backoff exponentiel pour relancer les demandes à des intervalles de plus en plus longs entre chaque nouvelle tentative. Introduisez un décalage entre les tentatives afin de randomiser les intervalles entre les tentatives. Limitez le nombre maximal de tentatives.

Résultat souhaité : les composants types d'un système logiciel distribué incluent les serveurs, les équilibreurs de charge, les bases de données et les serveurs DNS. Pendant le fonctionnement normal, ces composants peuvent répondre aux demandes par des erreurs temporaires ou limitées, ainsi que par des erreurs qui persisteraient indépendamment des nouvelles tentatives. Lorsque des clients adressent des demandes à des services, celles-ci consomment des ressources, notamment de la mémoire, des threads, des connexions, des ports ou toute autre ressource limitée. Le contrôle et la limitation des nouvelles tentatives constituent une stratégie visant à libérer et à minimiser la consommation de ressources afin que les composants du système soumis à des contraintes ne soient pas surchargés.

Lorsque le client demande une expiration du délai ou reçoit des réponses d'erreur, il doit décider de réessayer ou non. S'il recommence, il le fait avec un backoff exponentiel avec une instabilité et une valeur de nouvelle tentative maximale. Par conséquent, les services et processus back-end sont moins sollicités et le temps nécessaire pour s'autoréparer est réduit, ce qui se traduit par une récupération plus rapide et un traitement efficace des demandes.

Anti-modèles courants :

- Implémentation de nouvelles tentatives sans ajouter de valeurs de backoff exponentiel, d'instabilité et de nouvelles tentatives maximales. Le backoff et l'instabilité permettent d'éviter les pics de trafic artificiels dus à des tentatives involontaires coordonnées à intervalles réguliers.
- Implémentation de nouvelles tentatives sans tester leurs effets ou en supposant que les nouvelles tentatives sont déjà intégrées dans un kit SDK sans tester de scénarios de nouvelles tentatives.
- Incapacité à comprendre les codes d'erreur publiés à partir des dépendances, ce qui entraîne une nouvelle tentative pour toutes les erreurs, y compris celles dont la cause claire indique un manque d'autorisation, une erreur de configuration ou toute autre condition qui, comme on pouvait s'y attendre, ne sera pas résolue sans intervention manuelle.
- Ne pas aborder les pratiques d'observabilité, notamment la surveillance et l'envoi d'alertes en cas de pannes de service répétées afin que les problèmes sous-jacents soient connus et puissent être résolus.
- Développement de mécanismes de nouvelle tentative personnalisés lorsque des fonctionnalités de nouvelle tentative intégrées ou tierces suffisent.

- Réessayer à plusieurs couches de votre pile d'applications d'une manière qui complique les nouvelles tentatives et augmente la consommation de ressources lors d'une tempête de nouvelles tentatives. Assurez-vous de comprendre comment ces erreurs affectent votre application et les dépendances sur lesquelles vous vous appuyez, puis implémentez les nouvelles tentatives à un seul niveau.
- Réessayer les appels de service qui ne sont pas idempotents, ce qui peut entraîner des effets secondaires inattendus tels que des résultats dupliqués.

Avantages liés au respect de cette bonne pratique : les nouvelles tentatives aident les clients à obtenir les résultats souhaités lorsque les requêtes échouent, mais elles font également perdre plus de temps au serveur pour obtenir les réponses souhaitées. Lorsque les défaillances sont rares ou transitoires, les nouvelles tentatives fonctionnent bien. Lorsque les défaillances sont causées par une surcharge de ressources, les nouvelles tentatives peuvent aggraver la situation. L'ajout d'un backoff exponentiel avec instabilité aux nouvelles tentatives des clients permet aux serveurs de se rétablir en cas de défaillance provoquée par une surcharge de ressources. L'instabilité permet d'éviter l'alignement des demandes en pics, tandis que le backoff réduit l'escalade de charge provoquée par l'ajout de nouvelles tentatives au chargement normal des demandes. Enfin, il est important de configurer un nombre maximal de nouvelles tentatives ou un temps écoulé afin d'éviter de créer des backlogs susceptibles d'entraîner des échecs métastables.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Contrôler et limiter les appels de nouvelle tentative. Utilisez le backoff exponentiel pour réessayer après des intervalles progressivement plus longs. Introduisez l'instabilité pour randomiser les intervalles de nouvelle tentative et limiter le nombre maximal de nouvelles tentatives.

Les kits AWS SDK implémentent les nouvelles tentatives et le backoff exponentiel par défaut. Utilisez ces implémentations AWS intégrées là où cela s'avère nécessaire dans votre charge de travail. Implémentez une logique similaire dans votre charge de travail lorsque vous appelez des services qui sont idempotents et où les nouvelles tentatives améliorent la disponibilité de vos clients. Déterminez quels sont les délais d'expiration et quand les nouvelles tentatives doivent s'arrêter en fonction de votre cas d'utilisation. Créez et mettez en pratique des scénarios de test pour ces cas d'utilisation impliquant de nouvelles tentatives.

Étapes d'implémentation

- Déterminez la couche optimale de votre pile d'applications pour implémenter de nouvelles tentatives pour les services sur lesquels repose votre application.
- Soyez conscient des SDK existants qui mettent en œuvre des stratégies de nouvelle tentative éprouvées avec un backoff exponentiel et une instabilité pour la langue de votre choix, et privilégiez-les par rapport à l'écriture de vos propres implémentations de nouvelles tentatives.
- Vérifiez que les services sont idempotents avant de mettre en œuvre de nouvelles tentatives. Une fois les nouvelles tentatives mises en œuvre, assurez-vous qu'elles sont à la fois testées et régulièrement mises en œuvre en production.
- Lorsque vous appelez des API de service AWS, utilisez les kits AWS SDK et AWS CLI et comprenez les options de configuration d'une nouvelle tentative. Déterminez si les valeurs par défaut conviennent à votre cas d'utilisation, testez-les et ajustez-les si nécessaire.

Ressources

Bonnes pratiques associées :

- REL04-BP04 Rendre toutes les réponses idempotentes
- REL05-BP02 Limiter les demandes
- REL05-BP04 Procéder à une interruption immédiate et limiter les files d'attente
- REL05-BP05 Définir les délais d'attente du client
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances

Documents connexes :

- Nouvelles tentatives après erreur et backoff exponentiel dans AWS
- Amazon Builders' Library : délais d'attente, nouvelles tentatives et backoff avec instabilité
- Backoff exponentiel et instabilité
- Sécuriser les nouvelles tentatives avec des API idempotentes

Exemples connexes :

- Nouvelle tentative Spring

- Nouvelle tentative Resilience4j

Vidéos connexes :

- Retry, backoff, and jitter: AWS re:Invent 2019: Introducing The Amazon Builders' Library (DOP328)

Outils associés :

- Kits AWS SDK et outils : comportement en cas de nouvelle tentative
- AWS Command Line Interface : nouvelles tentatives AWS CLI

REL05-BP04 Procéder à une interruption immédiate et limiter les files d'attente

Lorsqu'un service n'est pas en mesure de répondre correctement à une demande, procédez à son interruption immédiate. Cela permet la libération des ressources associées à une demande et donne la possibilité au service de récupérer s'il lui manque des ressources. L'interruption immédiate est un modèle de conception logicielle bien établi qui peut être exploité pour créer des charges de travail hautement fiables dans le cloud. La mise en file d'attente est également un modèle d'intégration d'entreprise bien établi qui permet de faciliter le chargement et de permettre aux clients de libérer des ressources lorsque le traitement asynchrone peut être toléré. Lorsqu'un service est capable de répondre correctement dans des conditions normales, mais échoue lorsque le taux de demandes est trop élevé, utilisez une file d'attente pour mettre les demandes en mémoire tampon. Toutefois, ne permettez pas l'accumulation de longs backlogs de files d'attente susceptibles d'entraîner le traitement de demandes obsolètes auxquelles un client a déjà renoncé.

Résultat souhaité : lorsque les systèmes sont confrontés à des problèmes de ressources, à des dépassements de délai, à des exceptions ou à des pannes grises qui rendent les objectifs de niveau de service irréalisables, les stratégies d'interruption immédiate permettent d'accélérer la récupération du système. Les systèmes qui doivent absorber les pics de trafic et peuvent prendre en charge le traitement asynchrone peuvent améliorer la fiabilité en permettant aux clients de lancer rapidement des demandes grâce à l'utilisation des files d'attente pour mettre en mémoire tampon les demandes envoyées aux services back-end. Lors de la mise en mémoire tampon des demandes dans des files d'attente, des stratégies de gestion des files d'attente sont mises en œuvre pour éviter des backlogs insurmontables.

Anti-modèles courants :

- Mise en œuvre de files de messages sans configurer de files d'attente de lettres mortes (DLQ) ni d'alarmes sur les volumes DLQ pour détecter les défaillances d'un système.
- Il ne s'agit pas de mesurer l'ancienneté des messages dans une file d'attente, mais de mesurer la latence pour comprendre quand les utilisateurs prennent du retard ou si des erreurs entraînent de nouvelles tentatives.
- Conservation des messages en attente dans une file d'attente, alors qu'il n'est plus utile de traiter ces messages si l'entreprise n'en a plus besoin.
- La configuration de files d'attente du premier entré, premier sorti (FIFO) au moment du dernier entré, premier sorti (LIFO) permettrait de mieux répondre aux besoins des clients, par exemple lorsqu'un ordre strict n'est pas requis et que le traitement du backlog retarde toutes les nouvelles demandes urgentes, ce qui entraîne une violation des niveaux de service pour tous les clients.
- Exposition des files d'attente internes aux clients au lieu d'exposer les API qui gèrent la prise de travail et placent les demandes dans les files d'attente internes.
- La combinaison d'un trop grand nombre de types de demandes de travail dans une seule file d'attente peut aggraver les problèmes de backlog en répartissant la demande de ressources entre les types de demandes.
- Traitement de demandes complexes et simples dans la même file d'attente, malgré la nécessité d'une surveillance, de délais d'expiration et d'allocations de ressources différents.
- Absence de validation des entrées ou utilisation des assertions pour implémenter des mécanismes rapides dans les logiciels qui génèrent des exceptions vers des composants de niveau supérieur capables de gérer les erreurs de façon appropriée.
- Absence de suppression des ressources défectueuses du routage des requêtes, en particulier lorsque les défaillances sont grises, ce qui indique à la fois des réussites et des échecs en raison d'un plantage et d'un redémarrage, d'une panne de dépendance intermittente, d'une capacité réduite ou d'une perte de paquets réseau.

Avantages liés au respect de cette bonne pratique : les systèmes qui tombent rapidement en panne sont plus faciles à déboguer et à corriger, et présentent souvent des problèmes de codage et de configuration avant la publication des versions en production. Les systèmes qui intègrent des stratégies de mise en file d'attente efficaces offrent une résilience et une fiabilité accrues face aux pics de trafic et aux pannes intermittentes du système.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Les stratégies infaillibles peuvent être codées dans des solutions logicielles ou configurées dans l'infrastructure. En plus de leur capacité d'interruption immédiate, les files d'attente constituent une technique architecturale simple mais puissante qui permet de découpler les composants du système en douceur. Amazon CloudWatch fournit des fonctionnalités de surveillance et d'alerte en cas de défaillance. Une fois que l'on sait qu'un système est défaillant, des stratégies d'atténuation peuvent être invoquées, notamment en cas de défaillance de ressources altérées. Lorsque les systèmes implémentent des files d'attente avec Amazon SQS et d'autres technologies de file d'attente pour faciliter le chargement, ils doivent tenir compte de la gestion des backlogs de files d'attente, ainsi que des défaillances de consommation de messages.

Étapes d'implémentation

- Implémentez des assertions par programmation ou des métriques spécifiques dans votre logiciel et utilisez-les pour émettre des alertes explicites en cas de problèmes système. Amazon CloudWatch vous permet de créer des métriques et des alarmes en fonction du modèle de journal des applications et de l'instrumentation du kit SDK.
- Utilisez les métriques et les alarmes CloudWatch pour éviter les problèmes liés à l'altération des ressources qui ajoutent de la latence au traitement ou qui échouent à plusieurs reprises à traiter les demandes.
- Utilisez le traitement asynchrone en concevant des API pour accepter les demandes et les ajouter aux files d'attente internes en utilisant Amazon SQS, puis en répondant au client émetteur du message par un message de réussite afin que le client puisse libérer des ressources et passer à autre chose pendant que les utilisateurs de la file d'attente back-end traitent les demandes.
- Mesurez et surveillez la latence de traitement des files d'attente en produisant une métrique CloudWatch chaque fois que vous retirez un message d'une file d'attente en comparant l'heure actuelle à l'horodatage du message.
- Lorsque des défaillances empêchent le bon traitement des messages ou que des pics de trafic concernent des volumes qui ne peuvent pas être traités conformément aux contrats de niveau de service, mettez de côté le trafic ancien ou excédentaire vers une file d'attente de débordement. Cela permet de traiter en priorité les nouvelles tâches et les tâches plus anciennes lorsque la capacité est disponible. Cette technique est une approximation du traitement LIFO et permet un traitement normal du système pour toutes les nouvelles tâches.

- Utilisez des lettres mortes ou réadaptez des files d'attente afin de déplacer les messages qui ne peuvent pas être traités hors du backlog vers un emplacement pouvant faire l'objet de recherches et de résolutions ultérieures.
- Réessayez ou, si cela est acceptable, supprimez les anciens messages en comparant l'heure actuelle à l'horodatage du message et en supprimant les messages qui ne sont plus pertinents pour le client demandeur.

Ressources

Bonnes pratiques associées :

- REL04-BP02 Implémenter des dépendances couplées faiblement
- REL05-BP02 Limiter les demandes
- REL05-BP03 Contrôler et limiter les appels de nouvelle tentative
- REL06-BP02 Définir et calculer des métriques (agrégation)
- REL06-BP07 Surveiller la traçabilité complète des demandes via votre système

Documents connexes :

- Éviter les backlogs insurmontables dans les files d'attente
- Interruption immédiate
- Comment puis-je empêcher la croissance d'un backlog de messages dans ma file d'attente Amazon SQS ?
- Elastic Load Balancing : changement de zone
- Amazon Route 53Application Recovery Controller : contrôle du routage pour le basculement du trafic

Exemples connexes :

- Modèles d'intégration d'entreprise : canal des lettres mortes

Vidéos connexes :

- AWS re:Invent 2022 - Operating highly available Multi-AZ applications

Outils associés :

- Amazon SQS
- Amazon MQ
- AWS IoT Core
- Amazon CloudWatch

REL05-BP05 Définir les délais d'attente du client

Définissez les délais d'expiration de manière appropriée pour les connexions et les demandes, vérifiez-les systématiquement et ne vous fiez pas aux valeurs par défaut, car elles ne tiennent pas compte des spécificités de la charge de travail.

Résultat souhaité : les délais d'expiration du client doivent prendre en compte le coût pour le client, le serveur et la charge de travail associés à l'attente des demandes dont le traitement prend un temps anormal. Dans la mesure où il est impossible de connaître la cause exacte d'un délai d'attente, les clients doivent utiliser leur connaissance des services pour établir des attentes relatives aux causes probables et aux délais d'expiration appropriés.

Le délai d'expiration des connexions client dépend des valeurs configurées. Après avoir dépassé le délai imparti, les clients décident de revenir en arrière et de réessayer ou d'ouvrir un disjoncteur. Ces modèles évitent d'émettre des demandes susceptibles d'exacerber un problème d'erreur sous-jacent.

Anti-modèles courants :

- Ne pas connaître les délais d'expiration du système ou les délais d'expiration par défaut.
- Ne pas connaître le délai normal d'exécution des demandes.
- Ne pas connaître les raisons pour lesquelles les demandes peuvent prendre un temps anormalement long à traiter, ni les coûts pour le client, le service ou les performances de la charge de travail associés à l'attente de ces traitements.
- Ne pas connaître la probabilité qu'un réseau défaillant entraîne l'échec d'une requête uniquement lorsque le délai d'expiration est atteint, ainsi que les coûts pour les performances du client et de la charge de travail si l'on n'adopte pas un délai d'expiration plus court.
- Ne pas tester les scénarios de délai d'expiration à la fois pour les connexions et les demandes.
- Définir des délais d'expiration trop élevés, ce qui peut entraîner de longs temps d'attente et augmenter l'utilisation des ressources.

- Définir des délais d'attente trop bas, ce qui entraîne des défaillances artificielles.
- Oublier les modèles pour gérer les erreurs de temporisation des appels distants, par exemple les disjoncteurs et les nouvelles tentatives.
- Ne pas envisager de surveiller les taux d'erreur des appels de service, les objectifs de niveau de service en matière de latence et les valeurs aberrantes en matière de latence. Ces métriques peuvent fournir des informations sur les délais d'attente agressifs ou permissifs.

Avantages liés au respect de cette bonne pratique : les délais d'expiration des appels distants sont configurés et les systèmes sont conçus pour gérer les délais d'expiration de façon appropriée afin de préserver les ressources lorsque les appels distants répondent de manière anormalement lente et que les erreurs de délai d'expiration sont gérées de façon appropriée par les clients du service.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Définissez un délai d'expiration de la connexion et un délai d'expiration de la demande pour tout appel de dépendance de service et, plus généralement, pour tout appel entre processus. De nombreux cadres proposent des capacités de délai d'expiration intégrées, mais soyez prudent, car certains ont des valeurs par défaut infinies ou supérieures à ce qui est acceptable pour vos objectifs de service. Une valeur trop élevée réduit l'utilité du délai d'attente, car les ressources continuent d'être consommées pendant que le client attend l'expiration du délai. Une valeur trop faible peut générer un trafic accru sur le back-end et une latence accrue en raison du nombre excessif de demandes réessayées. Dans certains cas, cela peut entraîner des interruptions complètes, car toutes les demandes font l'objet d'une nouvelle tentative.

Tenez compte des points suivants lorsque vous déterminez des stratégies de délai d'expiration :

- Le traitement des demandes peut prendre plus de temps que d'habitude en raison de leur contenu, de défaillances d'un service cible ou d'une panne de partition réseau.
- Les demandes dont le contenu est anormalement coûteux peuvent consommer des ressources inutiles du serveur et du client. Dans ce cas, le fait d'avoir un délai d'expiration pour ces demandes et de ne pas réessayer peut préserver les ressources. Les services doivent également se protéger contre les contenus anormalement coûteux avec des limites et des délais d'expiration côté serveur.
- Les demandes qui prennent anormalement longtemps en raison d'une défaillance du service peuvent être interrompues et réessayées. Il convient de tenir compte des coûts de service liés à la demande et à la nouvelle tentative, mais si la cause est une déficience localisée, une nouvelle

tentative ne sera probablement pas coûteuse et réduira la consommation de ressources du client. Le délai d'expiration peut également libérer des ressources du serveur en fonction de la nature de la déficience.

- Les demandes dont l'exécution prend beaucoup de temps parce que la demande ou la réponse n'a pas été envoyée par le réseau peuvent être interrompues et réessayées. La demande ou la réponse n'ayant pas été envoyée, il en aurait résulté un échec indépendamment de la durée du délai imparti. Dans ce cas, l'expiration du délai ne libérera pas les ressources du serveur, mais des ressources client et cela améliorera les performances de la charge de travail.

Tirez parti de modèles de conception bien établis, tels que les nouvelles tentatives et les disjoncteurs, pour gérer les délais d'expiration de façon appropriée et adopter des approches d'interruption immédiate. Les kits AWS SDK et AWS CLI permettent de configurer les délais d'expiration de la connexion et des demandes, ainsi que d'effectuer de nouvelles tentatives avec des backoffs exponentiels et des instabilités. Les fonctions AWS Lambda prennent en charge la configuration des délais d'expiration et, avec AWS Step Functions, vous pouvez créer des disjoncteurs low-code qui tirent parti des intégrations prédéfinies avec des services AWS et des kits de développement logiciel (SDK). AWS App Mesh Envoy fournit des fonctionnalités de délai d'expiration et de disjoncteur.

Étapes d'implémentation

- Configurez les délais d'expiration pour les appels de service à distance et profitez des fonctionnalités de délai d'expiration spécifique à la langue intégrées ou des bibliothèques de délai d'expiration open source.
- Lorsque votre charge de travail passe des appels avec un kit AWS SDK, consultez la documentation pour la configuration du délai d'expiration spécifique à la langue.
  - Python
  - PHP
  - .NET
  - Ruby
  - Java
  - Go
  - Node.js
  - C++
- Lorsque vous utilisez des kits AWS SDK ou des commandes AWS CLI dans votre charge de travail, configurez les valeurs de délai d'expiration par défaut en définissant

les valeurs de configuration AWS par défaut pour `connectTimeoutInMillis` et `tlsNegotiationTimeoutInMillis`.

- Appliquez les options de ligne de commande `cli-connect-timeout` et `cli-read-timeout` pour contrôler les commandes AWS CLI ponctuelles destinées à des services AWS.
- Surveillez les appels de service à distance pour détecter les délais d'expiration et définissez des alarmes en cas d'erreurs persistantes afin de pouvoir gérer de manière proactive les scénarios d'erreur.
- Implémentez les métriques CloudWatch et la détection des anomalies CloudWatch sur les taux d'erreur d'appel, les objectifs de niveau de service en matière de latence et les valeurs aberrantes pour la latence afin de fournir des informations sur la gestion des délais d'attente trop agressifs ou permissifs.
- Configurez les délais d'expiration sur les fonctions Lambda.
- Les clients API Gateway doivent implémenter leurs propres tentatives lors de la gestion des délais d'expiration. API Gateway prend en charge un délai d'expiration de l'intégration de 50 millisecondes à 29 secondes pour les intégrations en aval et ne réessaie pas lorsque l'intégration demande un délai d'expiration.
- Implémentez le modèle de disjoncteur pour éviter de passer des appels à distance lorsque le délai d'expiration est écoulé. Ouvrez le circuit pour éviter les échecs d'appels et fermez-le lorsque les appels répondent normalement.
- Pour les charges de travail basées sur des conteneurs, consultez les fonctionnalités App Mesh Envoy permettant de tirer parti des délais d'expiration et des disjoncteurs intégrés.
- Utilisez AWS Step Functions pour créer des disjoncteurs low-code pour les appels de service à distance, en particulier lorsque vous appelez des kits AWS SDK natifs et des intégrations Step Functions compatibles afin de simplifier votre charge de travail.

Ressources

Bonnes pratiques associées :

- REL05-BP03 Contrôler et limiter les appels de nouvelle tentative
- REL05-BP04 Procéder à une interruption immédiate et limiter les files d'attente
- REL06-BP07 Surveiller la traçabilité complète des demandes via votre système

Documents connexes :

- Kits SDK AWS : nouvelles tentatives et délais d'attente
- Amazon Builders' Library : délais d'attente, nouvelles tentatives et backoff avec instabilité
- Quotas Amazon API Gateway et notes importantes
- AWS Command Line Interface : options de ligne de commande
- AWS SDK for Java 2.x : configuration des délais d'expiration de l'API
- AWSBotocore utilisant l'objet de configuration et la référence de configuration
- AWS SDK for .NET : nouvelles tentatives et délais d'expiration
- AWS Lambda : configuration des options de fonction Lambda

Exemples connexes :

- Utilisation du modèle de disjoncteur avec AWS Step Functions et Amazon DynamoDB
- Martin Fowler : disjoncteur

Outils associés :

- Les kits AWS SDK
- Les fonctions AWS Lambda
- Amazon SQS
- AWS Step Functions
- AWS Command Line Interface

REL05-BP06 Rendre les systèmes sans état dans la mesure du possible

Les systèmes ne doivent pas exiger d'état ou doivent décharger un état de telle sorte qu'entre les différentes demandes client, il n'y ait pas de dépendance vis-à-vis des données stockées localement sur disque et en mémoire. Cela permet le remplacement à volonté des serveurs sans impact sur la disponibilité.

Lorsque des utilisateurs ou des services interagissent avec une application, ils exécutent souvent une série d'interactions qui forment une session. Une session correspond aux données uniques des utilisateurs qui persistent entre les requêtes pendant l'utilisation de l'application. Une application sans état n'a pas besoin de connaître les interactions précédentes et ne stocke pas d'informations de session.

Lorsqu'une application est conçue pour être sans état, vous pouvez utiliser des services de calcul sans serveur, comme AWS Lambda ou AWS Fargate.

Outre le remplacement du serveur, les applications sans état ont également pour avantage de pouvoir être mises à l'échelle horizontalement, car toutes les ressources de calcul disponibles (telles que les instances EC2 et les fonctions AWS Lambda) peuvent répondre à n'importe quelle requête.

Avantages de la mise en place de cette bonne pratique : les systèmes conçus pour être sans état sont plus adaptables à une mise à l'échelle horizontale, ce qui permet d'ajouter ou de supprimer de la capacité en fonction des fluctuations du trafic et de la demande. Ils sont également intrinsèquement résilients aux défaillances et offrent flexibilité et agilité dans le cadre du développement d'applications.

Niveau de risque exposé si cette bonne pratique n'est pas établie: moyen

Directives d'implémentation

Rendre vos applications sans état. Les applications sans état permettent une mise à l'échelle horizontale et tolèrent la défaillance d'un nœud individuel. Analysez et comprenez les composants de votre application qui conservent leur état au sein de l'architecture. Cela vous permet d'évaluer l'impact potentiel de la transition vers une conception sans état. Une architecture sans état découple les données utilisateur et décharge les données de session. Cela permet de mettre à l'échelle chaque composant indépendamment pour répondre à des demandes variables de charge de travail et optimiser l'utilisation des ressources.

Étapes d'implémentation

- Identifiez et comprenez les composants avec état dans votre application.
- Découplez les données en séparant et en gérant les données utilisateur de la logique principale de l'application.
  - Amazon Cognito peut découpler les données utilisateur du code de l'application à l'aide de fonctionnalités telles que les groupes d'identités, les groupes d'utilisateurs et Amazon Cognito Sync.
  - Vous pouvez utiliser AWS Secrets Manager pour découpler les données des utilisateurs en stockant les secrets dans un emplacement sécurisé et centralisé. Cela signifie que le code de l'application n'a pas besoin de stocker de secrets, ce qui le rend plus sécurisé.
  - Envisagez d'utiliser Amazon S3 pour stocker des données volumineuses non structurées, telles que des images ou des documents. Votre application peut récupérer ces données en cas de besoin, évitant ainsi d'avoir à les stocker en mémoire.

  - Utilisez Amazon DynamoDB pour stocker des informations telles que des profils utilisateurs. Votre application peut interroger ces données en temps quasi réel.
- Déchargez les données de session dans une base de données, un cache ou des fichiers externes.
  - Amazon ElastiCache, Amazon DynamoDB, Amazon Elastic File System (Amazon EFS) et Amazon MemoryDB for Redis sont des exemples de services AWS que vous pouvez utiliser pour décharger les données de session.
- Concevez une architecture sans état après avoir identifié les données d'état et d'utilisateur qui doivent être conservées avec la solution de stockage de votre choix.

Ressources

Bonnes pratiques associées :

- REL11-BP03 Automatiser la réparation sur toutes les couches

Documents connexes :

- L'Amazon Builders' Library : éviter le basculement dans les systèmes distribués
- L'Amazon Builders' Library : éviter les retards de file d'attente insurmontables
- L'Amazon Builders' Library : défis et stratégies de mise en cache
- Best Practices for Stateless Web Tier on AWS

REL05-BP07 Mettre en place des leviers d'urgence

Les leviers d'urgence sont des processus rapides qui peuvent réduire l'impact sur la disponibilité de votre charge de travail.

Les leviers d'urgence fonctionnent en désactivant, en limitant ou en modifiant le comportement des composants ou des dépendances à l'aide de mécanismes connus et testés. Ils permettent d'atténuer les perturbations de la charge de travail causées par l'épuisement des ressources dû à une augmentation inattendue de la demande et de réduire l'impact des défaillances des composants non stratégiques de votre charge de travail.

Résultat souhaité : en mettant en place des leviers d'urgence, vous pouvez établir des processus dont le fonctionnement a été vérifié pour préserver la disponibilité des composants stratégiques de votre charge de travail. La charge de travail devrait se dégrader de manière appropriée et continuer

à remplir ses fonctions stratégiques durant l'activation d'un levier d'urgence. Pour plus d'informations sur la dégradation appropriée, consultez REL05-BP01 Implémenter une dégradation appropriée pour transformer les dépendances matérielles applicables en dépendances logicielles.

Anti-modèles courants :

- La défaillance des dépendances non stratégiques a un impact sur la disponibilité de votre charge de travail principale.
- Le comportement des composants stratégiques n'est pas testé ou vérifié lors d'une défaillance d'un composant non stratégique.
- Aucun critère clair et déterministe n'a été défini pour l'activation ou la désactivation d'un levier d'urgence.

Avantages liés au respect de cette bonne pratique : la mise en place de leviers d'urgence peut améliorer la disponibilité des composants stratégiques de votre charge de travail en fournissant à vos résolveurs des processus établis pour répondre à des pics de demande imprévus ou à des défaillances des dépendances non stratégiques.

Niveau de risque exposé si cette bonne pratique n'est pas établie: moyen

Directives d'implémentation

- Identifier les composants stratégiques de votre charge de travail.
- Concevoir et construire les composants stratégiques de votre charge de travail de manière à ce qu'ils résistent aux défaillances des composants non stratégiques.
- Effectuer des tests pour valider le comportement de vos composants stratégiques en cas de défaillance des composants non stratégiques.
- Définir et surveiller des métriques ou des déclencheurs pertinents pour lancer des procédures de levier d'urgence.
- Définir les procédures (manuelles ou automatisées) qui comprennent le levier d'urgence.

Étapes d'implémentation

- Identifier les composants stratégiques de votre charge de travail.
  - Chaque composant technique de votre charge de travail doit être associé à la fonction commerciale correspondante et classé comme stratégique ou non stratégique. Pour des exemples de fonctionnalités stratégiques et non stratégiques chez Amazon, consultez Any Day

Can Be Prime Day : How Amazon.com Search Uses Chaos Engineering to handle over 84K requests per second.

  - Il s'agit d'une décision à la fois technique et commerciale, qui varie en fonction de l'organisation et de la charge de travail.
- Concevoir et construire les composants stratégiques de votre charge de travail de manière à ce qu'ils résistent aux défaillances des composants non stratégiques.
  - Lors de l'analyse des dépendances, tenez compte de tous les modes de défaillance potentiels et vérifiez que vos mécanismes de levier d'urgence fournissent les fonctionnalités stratégiques aux composants en aval.
- Effectuer des tests pour valider le comportement de vos composants stratégiques pendant l'activation de vos leviers d'urgence.
  - Éviter les comportements bimodaux. Pour plus d'informations, consultez REL11-BP05 Utiliser la stabilité statique pour éviter les comportements bimodaux.
- Définir et surveiller des métriques pertinentes pour lancer des procédures de levier d'urgence.
  - La recherche des bonnes métriques à surveiller dépend de votre charge de travail. Parmi les métriques, citons la latence ou le nombre de demandes infructueuses à une dépendance.
- Définir les procédures (manuelles ou automatisées) qui comprennent le levier d'urgence.
  - Il peut s'agir de mécanismes tels que le délestage de charge, la limitation des requêtes ou la mise en œuvre d'une dégradation appropriée.

Ressources

Bonnes pratiques associées :

- REL05-BP01 Implémenter une dégradation appropriée pour transformer les dépendances matérielles applicables en dépendances logicielles
- REL05-BP02 Limiter les demandes
- REL11-BP05 Utiliser la stabilité statique pour éviter les comportements bimodaux

Documents connexes :

- Automatisation de déploiements sécurisés sans intervention
- Any Day Can Be Prime Day: How Amazon.com Search Uses Chaos Engineering to Handle Over 84K Requests Per Second

Vidéos connexes :

- AWS re:Invent 2020: Reliability, consistency, and confidence through immutability

Gestion des modifications

Les modifications apportées à votre charge de travail ou à son environnement doivent être anticipées et prises en compte pour assurer un fonctionnement fiable de la charge de travail. Les modifications incluent celles imposées à votre charge de travail telles que les pics de demande, ainsi que celles de l'intérieur comme les déploiements de fonctions et les correctifs de sécurité.

Les sections suivantes expliquent les bonnes pratiques en matière de gestion des modifications :

Rubriques

Surveiller les ressources de charge de travail

Les journaux et les métriques sont de puissants outils pour obtenir informations sur l'état de votre charge de travail. Vous pouvez configurer votre charge de travail de sorte à surveiller les journaux et les métriques et envoyer des notifications lorsque les seuils sont franchis ou que des événements significatifs se produisent. La surveillance permet à votre charge de travail de reconnaître quand des seuils de faibles performances sont franchis ou quand des défaillances se produisent, afin d'y répondre par une récupération automatique.

La surveillance est essentielle pour s'assurer que vous respectez vos exigences en matière de disponibilité. Votre surveillance doit détecter efficacement les pannes. La pire des pannes « silencieuse ». La fonctionnalité devient inopérante, mais il est impossible de détecter la panne, sinon indirectement. Vos clients sont informés avant vous. La fonction d'alerte en cas de problèmes est l'une des principales raisons de votre surveillance. Vos alertes doivent être séparées autant que possible de vos systèmes. Si votre interruption de service supprime votre capacité d'alerte, cela rallongera votre période d'interruption.

Chez AWS, nous instrumentons nos applications à plusieurs niveaux. Nous enregistrons la latence, les taux d'erreurs et la disponibilité pour chaque requête, pour toutes les dépendances et pour les opérations clés du processus. Nous enregistrons également des métriques sur le bon fonctionnement. Cela nous permet de voir les problèmes imminents avant qu'ils se produisent. Nous ne prenons pas seulement en compte la latence moyenne. Nous nous concentrons davantage sur

la latence hors norme, comme les 99,9e et 99,99e centiles. En effet, si une requête sur 1 000 ou 10 000 est lente, cela reste une mauvaise expérience. Ainsi, si votre moyenne est acceptable, mais qu'une requête sur 100 entraîne une latence extrême, un problème peut survenir lorsque votre trafic augmente.

La surveillance au sein d'AWS comporte quatre phases distinctes :

1. Génération : surveillance de tous les composants de la charge de travail
2. Agrégation : définition et calcul des métriques
3. Traitement et alarmes en temps réel : envoi de notifications et automatisation des réponses
4. Stockage et analyse

Bonnes pratiques

- REL06-BP01 Surveiller tous les composants de la charge de travail (génération)
- REL06-BP02 Définir et calculer des métriques (agrégation)
- REL06-BP03 Envoyer des notifications (traitement et alarmes en temps réel)
- REL06-BP04 Automatiser les réponses (traitement et alarmes en temps réel)
- REL06-BP05 Analytique
- REL06-BP06 Procéder à des examens réguliers
- REL06-BP07 Surveiller la traçabilité complète des demandes via votre système

REL06-BP01 Surveiller tous les composants de la charge de travail (génération)

Surveillez les composants de la charge de travail avec Amazon CloudWatch ou des outils tiers. Surveillez les services AWS avec le tableau de bord AWS Health.

Tous les composants de votre charge de travail doivent être surveillés, y compris le côté utilisateur, la logique métier et les niveaux de stockage. Au besoin, définissez des métriques clés, décrivez leur procédure d'extraction des journaux, puis spécifiez des seuils de déclenchement pour les événements d'alarme correspondants Assurez-vous que les métriques sont pertinentes pour les indicateurs clés de performance (KPI) de votre charge de travail, et utilisez des métriques et des journaux pour identifier les signes avant-coureurs de la dégradation du service. Par exemple, une métrique liée aux résultats commerciaux, telle que le nombre de commandes traitées avec succès

par minute, peut indiquer des problèmes de charge de travail plus rapidement qu'une métrique technique, telle que l'utilisation du processeur. Utilisez le tableau de bord AWS Health pour obtenir une vue personnalisée des performances et de la disponibilité des services AWS sous-jacents à vos ressources AWS.

La surveillance dans le cloud offre de nouvelles opportunités. La plupart des fournisseurs de cloud ont développé des hooks personnalisables et peuvent fournir des informations pour vous aider à surveiller plusieurs couches de votre charge de travail. Des services AWS comme Amazon CloudWatch appliquent des algorithmes statistiques et de machine learning pour analyser en continu les métriques des systèmes et des applications, déterminer les points de référence normaux et détecter les anomalies avec une intervention minimale de l'utilisateur. Les algorithmes de détection d'anomalies tiennent compte de la saisonnalité et des changements de tendance des métriques.

AWS met à disposition une multitude d'informations de surveillance et de journalisation qui peuvent être utilisées pour définir des métriques spécifiques à la charge de travail et des processus de changement de la demande, et pour adopter des techniques de machine learning, quelle que soit l'expertise en ML.

En outre, surveillez l'ensemble de vos points de terminaison externes afin de vous assurer qu'ils sont indépendants de votre implémentation de base. Cette surveillance active peut être effectuée avec des transactions synthétiques (parfois appelées tests canary utilisateur, mais à ne pas confondre avec les déploiements canary), qui exécutent périodiquement plusieurs tâches communes correspondant aux actions effectuées par les clients de la charge de travail. Maintenez ces tâches de courte durée et veillez à ne pas surcharger votre charge de travail pendant les tests. Amazon CloudWatch Synthetics vous permet de créer des tests canary synthétiques pour surveiller vos points de terminaison et vos API. Vous pouvez également combiner les nœuds de clients synthétiques Canari avec la console AWS X-Ray pour identifier les scripts Canari synthétiques qui rencontrent des erreurs, des pannes ou des taux de limitation au cours de la période sélectionnée.

Résultat souhaité :

Collectez et utilisez des métriques critiques de tous les composants de la charge de travail pour garantir la fiabilité de la charge de travail et une expérience utilisateur optimale. Détecter qu'une charge de travail n'atteint pas les résultats vous permet de déclarer rapidement un sinistre et de vous remettre d'un incident.

Anti-modèles courants :

- Surveillance limitée aux interfaces externes de votre charge de travail.

- Ne pas générer de métriques spécifiques à la charge de travail et s'appuyer uniquement sur les métriques qui vous sont fournies par les services AWS utilisés par votre charge de travail.
- Utiliser uniquement des métriques techniques dans votre charge de travail et ne surveiller aucune métrique liée aux KPI non techniques auxquels la charge de travail contribue.
- S'appuyer sur le trafic de production et de simples vérifications de l'état pour surveiller et évaluer l'état de la charge de travail.

Avantages liés au respect de cette bonne pratique : La surveillance à tous les niveaux de votre charge de travail vous permet d'anticiper et de résoudre plus rapidement les problèmes dans les composants qui constituent la charge de travail.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

1. Activer la journalisation si disponible. Les données de surveillance doivent être obtenues à partir de tous les composants des charges de travail. Activez la journalisation supplémentaire, telle que les journaux d'accès S3, et autorisez votre charge de travail à consigner des données qui lui sont spécifiques. Collectez des métriques pour les moyennes d'UC, d'E/S réseau et d'E/S disque à partir de services comme Amazon ECS, Amazon EKS, Amazon EC2, Elastic Load Balancing, AWS Auto Scaling et Amazon EMR. Consulter Services AWS publiant des métriques CloudWatch pour obtenir une liste des services AWS qui publient des métriques CloudWatch.
2. Passez en revue toutes les métriques par défaut et explorez toutes les lacunes de collecte de données. Chaque service génère des métriques par défaut. La collecte des métriques par défaut vous permet de mieux comprendre les dépendances entre les composants de charge de travail et sur la manière dont la fiabilité et les performances des composants affectent la charge de travail. Vous pouvez également créer et publier vos propres métriques vers CloudWatch à l'aide d'AWS CLI ou d'une API. Évaluez
3. toutes les métriques pour décider celles pour lesquelles envoyer des alertes pour chaque service AWS de votre charge de travail. Vous pouvez choisir de sélectionner un sous-ensemble de métriques qui ont un impact majeur sur la fiabilité de la charge de travail. Se concentrer sur les métriques critiques et le seuil vous permet d'affiner le nombre d'information et peut aider à minimiser les faux positifs.
4. Définissez des alertes et le processus de récupération de votre charge de travail après le déclenchement de l'alerte. La définition d'alertes vous permet de notifier, de faire remonter et de suivre rapidement les étapes nécessaires pour vous remettre d'un incident et atteindre votre

objectif de temps de récupération (RTO) prescrit. Vous pouvez utiliser des alarmes Amazon CloudWatch pour appeler des flux de travail automatisés et lancer des procédures de récupération en fonction de seuils définis.

5. Explorez l'utilisation de transactions synthétiques pour collecter des données pertinentes sur l'état des charges de travail. La surveillance synthétique suit les mêmes routes et effectue les mêmes actions qu'un client, ce qui vous permet de vérifier en permanence l'expérience client même lorsque vous n'avez aucun trafic client sur vos charges de travail. En utilisant les transactions synthétiques, vous pouvez découvrir les problèmes avant vos clients.

Ressources

Bonnes pratiques associées :

- REL11-BP03 Automatiser la réparation sur toutes les couches

Documents connexes :

- Premiers pas avec le tableau de bord AWS Health : état de votre compte
- Services AWS publiant des métriques CloudWatch
- Journaux d'accès pour votre Network Load Balancer
- Journaux d'accès pour votre Application Load Balancer
- Accès à Amazon CloudWatch Logs pour AWS Lambda
- Journalisation de l'accès au serveur Amazon S3
- Activer les journaux d'accès pour votre Classic Load Balancer
- Exportation de données de journal vers Amazon S3
- Installer l'agent CloudWatch sur une instance Amazon EC2
- Publication des métriques personnalisées
- Fonctionnement des tableaux de bord Amazon CloudWatch
- Utilisation des métriques Amazon CloudWatch
- Utilisation de scripts Canary (Amazon CloudWatch Synthetics)
- Que sont les Amazon CloudWatch Logs ?

Guides de l'utilisateur :

- Création d'un journal d'activité
- Surveillance des métriques de mémoire et de disque pour les instances Linux Amazon Amazon EC2
- Utiliser CloudWatch Logs avec des instances de conteneur
- Journaux de flux VPC
- Qu'est-ce qu'Amazon DevOps Guru ?
- Qu'est-ce que AWS X-Ray ?

Blogs connexes :

- Débogage avec Amazon CloudWatch Synthetics et AWS X-Ray

Exemples et ateliers connexes :

- Ateliers AWS Well-Architected : Excellence opérationnelle - Surveillance des dépendances
- L'Amazon Builders' Library : Instrumentation des systèmes distribués au profit de la visibilité opérationnelle
- Atelier sur l'observabilité

REL06-BP02 Définir et calculer des métriques (agrégation)

Stockez les données des journaux et appliquez des filtres si nécessaire pour calculer les métriques, en particulier le décompte d'un événement de journal spécifique ou la latence calculée à partir des horodatages des événements de journaux.

Amazon CloudWatch et Amazon S3 servent de couches principales pour l'agrégation et le stockage. Pour certains services, comme AWS Auto Scaling et Elastic Load Balancing, les métriques par défaut sont fournies par défaut pour la charge du processeur ou la latence moyenne des demandes au sein d'un cluster ou d'une instance. Pour les services de streaming, comme les journaux de flux VPC et AWS CloudTrail, les données d'événement sont transmises à CloudWatch Logs et vous devez définir et appliquer des filtres de métrique pour extraire des métriques à partir des données d'événement. Cela vous donne des données de séries chronologiques, qui peuvent servir d'entrées aux alarmes CloudWatch que vous définissez pour déclencher les alertes.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

- Définir et calculer des métriques (agrégation). Stockez les données des journaux et appliquez des filtres si nécessaire pour calculer les métriques, en particulier le décompte d'un événement de journal spécifique ou la latence calculée à partir des horodatages des événements de journaux
  - Les filtres de métrique définissent les termes et les modèles à rechercher dans les données de journal lorsqu'elles sont envoyées à CloudWatch Logs. CloudWatch Logs utilise ces filtres de métriques pour transformer les données de journal en métriques numériques CloudWatch que vous pouvez représenter graphiquement ou au niveau desquelles vous pouvez définir une alarme.
    - Recherche et filtrage des données de journaux
  - Utiliser un agent tiers de confiance pour agréger les journaux
    - Suivez les instructions de l'agent tiers. La plupart des produits tiers s'intègrent à CloudWatch et Amazon S3.
  - Certains services AWS peuvent publier des journaux directement dans Amazon S3. Ainsi, si votre principale exigence pour les journaux est le stockage dans Amazon S3, vous pouvez facilement faire en sorte que le service produisant les journaux les envoie directement à Amazon S3 sans configurer d'infrastructure supplémentaire.
    - Envoi des journaux directement à Amazon S3

Ressources

Documents connexes :

- Exemples de requêtes Amazon CloudWatch Logs Insights
- Débogage avec Amazon CloudWatch Synthetics et AWS X-Ray
- Un atelier sur l'observabilité
- Recherche et filtrage des données de journaux
- Envoi des journaux directement à Amazon S3
- L'Amazon Builders' Library : Instrumentation des systèmes distribués au profit de la visibilité opérationnelle

REL06-BP03 Envoyer des notifications (traitement et alarmes en temps réel)

Lorsque les organisations détectent des problèmes potentiels, elles envoient des notifications et des alertes en temps réel au personnel et aux systèmes appropriés afin de résoudre rapidement et efficacement ces problèmes.

Résultat souhaité : Il est possible d'obtenir des réponses rapides aux événements opérationnels en configurant des alarmes pertinentes basées sur les métriques de service et d'application. Lorsque les seuils d'alarme sont dépassés, le personnel et les systèmes appropriés sont avertis afin de résoudre les problèmes sous-jacents.

Anti-modèles courants :

- Vous configurez les alarmes avec un seuil trop élevé, ce qui fait échouer l'envoi des notifications vitales.
- Vous configurez les alarmes avec un seuil trop bas, ce qui empêche la prise en compte des alertes importantes à cause du bruit généré par un trop grand nombre de notifications.
- Vous ne mettez pas à jour les alarmes et leur seuil en cas de changement d'utilisation.
- Pour les alarmes qu'il est préférable de traiter par des actions automatisées, vous envoyez la notification au personnel au lieu de générer l'action automatisée, ce qui entraîne l'envoi d'un trop grand nombre de notifications.

Avantages liés au respect de cette bonne pratique : En envoyant des notifications et des alertes en temps réel au personnel et aux systèmes appropriés, vous pouvez détecter rapidement les problèmes et réagir rapidement face aux incidents opérationnels.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Les charges de travail doivent être équipées d'un système de traitement et d'avertissement en temps réel afin d'améliorer la détectabilité des problèmes susceptibles d'affecter la disponibilité de l'application et de déclencher une réponse automatique. Les organisations peuvent procéder au traitement et à l'avertissement en temps réel en créant des alertes avec des métriques définies afin de recevoir des notifications chaque fois que des événements importants se produisent ou qu'une métrique dépasse un seuil.

Amazon CloudWatch vous permet de créer des alarmes de métrique et des alarmes composites au moyen d'alarmes CloudWatch basées sur le seuil statique, la détection des anomalies et d'autres critères. Pour plus de détails sur les types d'alarmes que vous pouvez configurer avec CloudWatch, consultez la section relative aux alarmes dans la documentation CloudWatch. »

Vous pouvez créer des vues personnalisées des métriques et des alertes de vos ressources AWS pour vos équipes en utilisant les tableaux de bord CloudWatch. » Les pages d'accueil personnalisables de la console CloudWatch vous permettent de surveiller vos ressources dans une vue unique sur plusieurs régions.

Les alarmes peuvent effectuer une ou plusieurs actions, comme envoyer une notification à une rubrique Amazon SNS, exécuter une action Amazon EC2 ou une action Amazon EC2 Auto Scaling , ou créer un OpsItem ou de réponse dans AWS Systems Manager.

Amazon CloudWatch utilise Amazon SNS pour envoyer des notifications lorsque l'alarme change d'état, ce qui permet de transmettre les messages des diffuseurs de publication (producteurs) aux abonnés (consommateurs). Pour plus de détails sur la configuration des notifications Amazon SNS, consultez Configuration d'Amazon SNS. »

CloudWatch envoie des événements EventBridge 43 % chaque fois qu'une alarme CloudWatch est créée, mise à jour, supprimée ou que son état change. Vous pouvez utiliser EventBridge avec ces événements pour créer des règles qui exécutent des actions, comme vous avertir chaque fois que l'état d'une alarme change ou déclencher automatiquement des événements sur votre compte à l'aide de l'automatisation Systems Manager. »

Quand utiliser EventBridge ou Amazon SNS ?

EventBridge et Amazon SNS peuvent être utilisés pour développer des applications pilotées par des événements. Votre choix dépendra de vos besoins spécifiques.

Amazon EventBridge est recommandé lorsque vous souhaitez créer une application qui réagit aux événements de vos propres applications, des applications SaaS et des services AWS. EventBridge est le seul service basé sur les événements qui s'intègre directement aux partenaires SaaS tiers. EventBridge ingère également automatiquement les événements de plus de 200 services AWS sans que les développeurs n'aient à créer de ressources sur leur compte.

EventBridge utilise une structure définie basée sur JSON pour les événements et vous aide à créer des règles qui s'appliquent à l'ensemble du corps de l'événement afin de sélectionner les événements à transférer vers une cible. EventBridge prend actuellement en charge plus de

20 services AWS en tant que cibles, y compris AWS Lambda, Amazon SQS, Amazon SNS, Amazon Kinesis Data Streamset Amazon Data Firehose. »

Amazon SNS est recommandé pour les applications qui nécessitent un niveau de diffusion élevé (des milliers, voire des millions de points de terminaison). Il est courant d'observer que les clients utilisent Amazon SNS comme cible pour leur règle afin de filtrer les événements dont ils ont besoin et de les diffuser sur plusieurs points de terminaison.

Les messages ne sont pas structurés et peuvent être de n'importe quel format. Amazon SNS prend en charge le transfert de messages vers six types de cibles différents, notamment Lambda, Amazon SQS, les points de terminaison HTTP/S, les SMS, les notifications push mobiles et les e-mails. La latence type de Amazon SNS est inférieure à 30 millisecondes. » Un large éventail de services AWS envoie des messages Amazon SNS en configurant le service dans cet objectif (plus de 30, y compris Amazon EC2, Amazon S3et Amazon RDS).

Étapes d'implémentation

1. Créez une alarme à l'aide d'alarmes Amazon CloudWatch. »
    a. Une alarme de métrique surveille une seule métrique CloudWatch ou une expression qui dépend de métriques CloudWatch. L'alarme déclenche une ou plusieurs actions en fonction de la valeur de la métrique ou de l'expression par rapport à un seuil sur un certain nombre d'intervalles de temps. L'action peut consister à envoyer une notification à une rubrique Amazon SNS, exécuter une action Amazon EC2 ou une action Amazon EC2 Auto Scaling , ou créer un OpsItem ou de réponse dans AWS Systems Manager.
    b. Une alarme composite est une expression de règle qui prend en compte les conditions d'alarme des autres alarmes que vous avez créées. L'alarme composite ne passe en état d'alarme que si toutes les conditions de la règle sont satisfaites. Les alarmes spécifiées dans l'expression de règle d'une alarme composite peuvent inclure des alarmes de métrique et des alarmes composites supplémentaires. Les alarmes composites peuvent envoyer des notifications Amazon SNS lorsque leur état change et peuvent créer des Systems Manager OpsItems ou des incidents lorsqu'ils entrent en état d'alarme, mais qu'ils ne peuvent effectuer aucune action Amazon EC2 ou Auto Scaling.
2. Configurez les notifications Amazon SNS. » Lorsque vous créez une alarme CloudWatch, vous pouvez inclure une rubrique Amazon SNS pour envoyer une notification lorsque l'alarme change d'état.
3. Créez des règles dans EventBridge qui correspondent aux alarmes CloudWatch spécifiées. Chaque règle prend en charge plusieurs cibles, y compris des fonctions Lambda. Par exemple,

vous pouvez définir une alarme qui se déclenche lorsque l'espace disque disponible est insuffisant, ce qui déclenche une fonction Lambda par le biais d'une règle EventBridge permettant de libérer de l'espace. Pour plus de détails sur les cibles EventBridge, consultez Cibles EventBridge (langue française non garantie).. »

Ressources

Bonnes pratiques Well-Architected connexes :

- REL06-BP01 Surveiller tous les composants de la charge de travail (génération)
- REL06-BP02 Définir et calculer des métriques (agrégation)
- REL12-BP01 Utiliser des playbooks pour enquêter sur les causes des défaillances

Documents connexes :

- Amazon CloudWatch
- Informations CloudWatch Logs
- Utilisation des alarmes Amazon CloudWatch
- Fonctionnement des tableaux de bord Amazon CloudWatch
- Utilisation des métriques Amazon CloudWatch
- Configuration de notifications Amazon SNS (langue française non garantie)
- la détection des anomalies CloudWatch
- Protection des données CloudWatch Logs
- Amazon EventBridge
- Amazon Simple Notification Service

Vidéos connexes :

- reinvent 2022 observability videos
- AWS re:Invent 2022 - Observability best practices at Amazon

Exemples connexes :

- Un atelier sur l'observabilité

- Amazon EventBridge à AWS Lambda avec contrôle du feedback par des alarmes Amazon CloudWatch

REL06-BP04 Automatiser les réponses (traitement et alarmes en temps réel)

utilisez l'automatisation pour agir en cas de détection d'événement, par exemple, pour remplacer les composants défectueux.

Un traitement automatique en temps réel des alarmes est mis en œuvre afin que les systèmes puissent prendre rapidement des mesures correctives et tenter d'éviter les pannes ou une dégradation du service lorsque les alarmes se déclenchent. Les réponses automatisées aux alarmes peuvent inclure le remplacement des composants défaillants, l'ajustement de la capacité de calcul, la redirection du trafic vers des hôtes, des zones de disponibilité ou d'autres régions en bonne santé, et la notification des opérateurs.

Résultat souhaité : des alarmes en temps réel sont identifiées et un traitement automatisé des alarmes est mis en place pour déclencher les actions appropriées afin de maintenir les objectifs de niveau de service et les contrats de niveau de service (SLA). L'automatisation peut aller de l'autoréparation de composants individuels au basculement complet du site.

Anti-modèles courants :

- Pas d'inventaire ou de catalogue clair des principales alarmes en temps réel.
- Aucune réponse automatique aux alarmes critiques (par exemple, lorsque la capacité de calcul est presque épuisée, une mise à l'échelle automatique se produit).
- Réponses aux alarmes contradictoires.
- Pas de procédure opérationnelle standard (SOP) que les opérateurs doivent suivre lorsqu'ils reçoivent des notifications d'alerte.
- Pas de surveillance des modifications de configuration, alors que des changements de configuration non détectés peuvent entraîner des temps d'arrêt pour les charges de travail.
- Pas de stratégie pour annuler les modifications de configuration involontaires.

Avantages liés au respect de cette bonne pratique : l'automatisation du traitement des alarmes peut améliorer la résilience du système. Le système prend automatiquement des mesures correctives, réduisant ainsi les activités manuelles qui nécessitent des interventions humaines sujettes aux

erreurs. L'exécution de la charge de travail permet d'atteindre les objectifs de disponibilité et de réduire les interruptions de service.

Niveau de risque exposé si cette bonne pratique n'est pas établie: moyen

Directives d'implémentation

Pour gérer efficacement les alertes et automatiser leur réponse, classez les alertes en fonction de leur criticité et de leur impact, documentez les procédures de réponse et planifiez les réponses avant de classer les tâches.

Identifiez les tâches nécessitant des actions spécifiques (souvent détaillées dans les runbooks) et examinez tous les runbooks et playbooks pour déterminer les tâches qui peuvent être automatisées. Si les actions peuvent être définies, alors elles sont souvent automatisables. Si elles ne le sont pas, documentez les étapes manuelles dans une SOP et formez les opérateurs à ces étapes. Remettez continuellement en question les processus manuels pour trouver des opportunités d'automatisation où vous pouvez établir et maintenir un plan d'automatisation des réponses aux alertes.

Étapes d'implémentation

1. Dresser un inventaire des alarmes : pour obtenir la liste de toutes les alarmes, vous pouvez utiliser AWS CLI à l'aide de la commande Amazon CloudWatch `describe-alarms`. Selon le nombre d'alarmes que vous avez configurées, vous devrez peut-être utiliser la pagination pour récupérer un sous-ensemble d'alarmes pour chaque appel, ou vous pouvez utiliser le SDK AWS pour obtenir les alarmes à l'aide d'un appel d'API.
2. Documenter les actions de toutes les alarmes : tenez à jour un runbook avec toutes les alarmes et leurs actions, qu'elles soient manuelles ou automatisées. AWS Systems Manager fournit des runbooks prédéfinis. Pour plus d'informations sur les runbooks, consultez Créer vos propres runbooks. Pour plus d'informations sur l'affichage du contenu du runbook, consultez Afficher le contenu du runbook.
3. Configurer et gérer les actions des alarmes : pour toutes les alarmes nécessitant une action, spécifiez l'action automatisée à l'aide du SDK CloudWatch. Par exemple, vous pouvez modifier automatiquement l'état de vos instances Amazon EC2 basées sur une alarme CloudWatch en créant des actions sur l'alarme et en les activant ou désactivant.

   Vous pouvez également utiliser Amazon EventBridge pour répondre automatiquement aux événements du système (problèmes de disponibilité d'une application ou modifications de ressources, par exemple). Vous pouvez créer des règles pour indiquer les événements qui vous

intéressent et les mesures à prendre lorsqu'un événement correspond à une règle. Les actions qui peuvent être lancées automatiquement incluent l'appel d'une fonction AWS Lambda, l'appel d'Amazon EC2 `Run Command`, le relais de l'événement à Amazon Kinesis Data Streams et l'affichage d'Automatiser Amazon EC2 en utilisant EventBridge.

4. Procédures opérationnelles standard (SOP) : en fonction des composants de votre application, AWS Resilience Hub recommande plusieurs modèles de SOP. Vous pouvez utiliser ces SOP pour documenter tous les processus qu'un opérateur doit suivre en cas d'alerte. Vous pouvez également créer une SOP en fonction des recommandations d'Resilience Hub, lorsque vous avez besoin d'une application Resilience Hub avec une stratégie de résilience associée, ainsi que d'une évaluation historique de la résilience de cette application. Les recommandations de SOP sont générées par l'évaluation de la résilience.

   Resilience Hub travaille de pair avec Systems Manager pour automatiser les étapes de vos SOP en fournissant un certain nombre de documents SSM que vous pouvez utiliser comme référence pour ces SOP. Par exemple, Resilience Hub peut recommander une procédure SOP pour ajouter de l'espace disque sur la base d'un document d'automatisation SSM existant.
5. Effectuer des actions automatisées avec Amazon DevOps Guru : vous pouvez utiliser Amazon DevOps Guru pour surveiller automatiquement les ressources de votre application afin de détecter des comportements anormaux et fournir des recommandations ciblées afin d'accélérer l'identification des problèmes et les délais de résolution. Avec DevOps Guru, vous pouvez surveiller les flux de données opérationnelles de plusieurs sources en temps quasi réel, notamment des métriques Amazon CloudWatch, AWS ConfigAWS CloudFormation et AWS X-Ray. Vous pouvez également utiliser DevOps Guru pour créer automatiquement des OpsItems dans OpsCenter et envoyer des événements à EventBridge pour une automatisation supplémentaire.

Ressources

Bonnes pratiques associées :

- REL06-BP01 Surveiller tous les composants de la charge de travail (génération)
- REL06-BP02 Définir et calculer des métriques (agrégation)
- REL06-BP03 Envoyer des notifications (traitement et alarmes en temps réel)
- REL08-BP01 Utiliser des runbooks pour les activités standard telles que le déploiement

Documents connexes :

- AWS Systems Manager Automation
- Creating an EventBridge Rule That Triggers on an Event from an AWS Resource
- One Observability Workshop
- Amazon Builders' Library : Instrumenter les systèmes distribués pour une visibilité opérationnelle
- What is Amazon DevOps Guru?
- Utilisation des documents d'automatisation (playbooks)

Vidéos connexes :

- AWS re:Invent 2022 - Observability best practices at Amazon
- AWS re:Invent 2020: Automate anything with AWS Systems Manager
- Introduction to AWS Resilience Hub
- Create Custom Ticket Systems for Amazon DevOps Guru Notifications
- Enable Multi-Account Insight Aggregation with Amazon DevOps Guru

Exemples connexes :

- Ateliers sur la fiabilité
- Amazon CloudWatch and Systems Manager Workshop

REL06-BP05 Analytique

collectez les fichiers journaux et les historiques de métriques, puis analysez-les pour obtenir des informations plus générales sur les tendances et la charge de travail.

Amazon CloudWatch Logs Insights prend en charge un langage de requête simple et puissant que vous pouvez utiliser pour analyser les données des journaux. Amazon CloudWatch Logs prend également en charge les abonnements qui permettent aux données de circuler en toute transparence vers Amazon S3, où vous pouvez utiliser Amazon Athena pour interroger les données. Il prend également en charge les requêtes dans une grande variétés de formats. Consulter Formats de données et SerDes pris en charge dans le guide de l'utilisateur Amazon Athena pour plus d'informations. Pour analyser des ensembles de fichiers journaux volumineux, vous pouvez exécuter un cluster Amazon EMR pour exécuter des analyses à l'échelle du pétaoctet.

Il existe divers outils fournis par les partenaires AWS et les tiers qui permettent l'agrégation, le traitement, le stockage et l'analyse. Parmi ces outils figurent New Relic, Splunk, Loggly, Logstash, CloudHealth et Nagios. Cependant, la génération en dehors du système et des journaux d'applications est propre à chaque fournisseur de cloud, et généralement, spécifique à chaque service.

Un partie souvent négligée de la surveillance des processus concerne la gestion des données. Vous devez déterminer les exigences de rétention des données de surveillance, puis appliquer des stratégies de cycle de vie en conséquence. Amazon S3 prend en charge la gestion du cycle de vie au niveau du compartiment S3. Cette gestion du cycle de vie peut être appliquée différemment à d'autres chemins dans le compartiment. Vers la fin du cycle de vie, vous pouvez transférer des données dans Amazon S3 Glacier pour un stockage à long terme, puis les laisser expirer une fois la fin de la période de rétention terminée. La classe de stockage S3 Intelligent-Tiering est conçue pour optimiser les coûts en transférant automatiquement les données vers le niveau d'accès le plus économique, sans impact sur les performances ni surcharge opérationnelle.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

- CloudWatch Logs vous permet de rechercher et d'analyser de manière interactive vos données de journaux dans Amazon CloudWatch Logs.
  - Analyse des données des journaux avec CloudWatch Logs Insights
  - Exemples de requêtes Amazon CloudWatch Logs Insights
- Utiliser Amazon CloudWatch Logs pour envoyer des journaux vers Amazon S3 où vous pouvez les exploiter ou utiliser Amazon Athena pour interroger les données
  - Comment analyser mes journaux d'accès au serveur Amazon S3 à l'aide d'Athena ?
    - Créez une stratégie de cycle de vie S3 pour votre compartiment de journaux d'accès au serveur. Configurez la stratégie de cycle de vie de sorte à supprimer régulièrement les fichiers journaux. Cette suppression permet de réduire la quantité de données analysées par Athena pour chaque requête.
      - Comment créer une stratégie de cycle de vie pour un compartiment S3 ?

Ressources

Documents connexes :

- Exemples de requêtes Amazon CloudWatch Logs Insights
- Analyse des données des journaux avec CloudWatch Logs Insights
- Débogage avec Amazon CloudWatch Synthetics et AWS X-Ray
- Comment créer une stratégie de cycle de vie pour un compartiment S3 ?
- Comment analyser mes journaux d'accès au serveur Amazon S3 à l'aide d'Athena ?
- Un atelier sur l'observabilité
- L'Amazon Builders' Library : Instrumentation des systèmes distribués au profit de la visibilité opérationnelle

REL06-BP06 Procéder à des examens réguliers

Examinez fréquemment comment la surveillance de la charge de travail est mise en œuvre et mettez-la à jour en fonction des événements et des modifications majeurs.

Une surveillance efficace repose sur des métriques commerciales clés. Assurez-vous que ces métriques sont prises en compte dans votre charge de travail au fur et à mesure que les priorités de l'entreprise évoluent.

Auditer votre surveillance vous permet de savoir avec certitude quand une application est conforme à ses objectifs de disponibilité. Pour pouvoir analyser les causes premières, il faut pouvoir découvrir ce qui se passe lorsque des défaillances se produisent. AWS fournit des services qui vous permettent de suivre l'état de vos services lors d'un incident :

- Amazon CloudWatch Logs : vous pouvez stocker vos journaux dans ce service et inspecter leur contenu.
- Amazon CloudWatch Logs Insights : service entièrement géré qui vous permet d'analyser des journaux volumineux en quelques secondes. Il permet des requêtes et des visualisations rapides et interactives.
- AWS Config : permet de voir quelle infrastructure AWS a été utilisée à différents moments.
- AWS CloudTrail : permet de voir quelles API AWS ont été appelées à quel moment et par quel mandataire.

Chez AWS, nous organisons des réunions hebdomadaires pour examiner les performances opérationnelles et partager les enseignements entre les équipes. Compte tenu du nombre

conséquent d'équipes AWS, nous avons créé The Wheel pour choisir de façon aléatoire une charge de travail à examiner. Le respect d'un rythme régulier pour l'examen des performances opérationnelles et le partager des connaissances améliore la capacité de vos équipes opérationnelles à atteindre des performances supérieures.

Anti-modèles courants :

- Collecte limitée aux métriques par défaut.
- Définition d'une stratégie de surveillance et sans examen.
- Absence de discussion relative à la surveillance lorsque des modifications majeures sont déployées.

Avantages liés au respect de cette bonne pratique : La vérification régulière de votre surveillance permet d'anticiper les problèmes potentiels, au lieu de réagir aux notifications lorsqu'un problème anticipé se produit réellement.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

- Créer plusieurs tableaux de bord pour la charge de travail. Vous devez disposer d'un tableau de bord de niveau supérieur qui contient les principales métriques commerciales, ainsi que les métriques techniques que vous avez identifiées comme étant les plus pertinentes pour l'état projeté de la charge de travail au fil de la variation de l'utilisation. Vous devez également avoir des tableaux de bord pour différents niveaux et dépendances d'application qui peuvent être inspectés.
  - Fonctionnement des tableaux de bord Amazon CloudWatch
- Planifier et effectuer des vérifications régulières des tableaux de bord de charge de travail. Effectuez une inspection régulière des tableaux de bord. Vous pouvez avoir des cadences différentes selon la profondeur à laquelle vous inspectez.
  - Inspecter les tendances dans les métriques. Comparez les valeurs des métriques aux valeurs historiques pour voir s'il existe des tendances qui peuvent indiquer que quelque chose doit faire l'objet d'une enquête. Voici quelques exemples : augmentation de la latence, diminution de la fonction principale de l'entreprise et augmentation des réponses aux échecs.
  - Inspecter les valeurs atypiques ou les anomalies dans vos métriques. Les moyennes ou les médianes peuvent masquer des valeurs atypiques et des anomalies. Observez les valeurs les plus élevées et les plus faibles pendant la période et examinez les causes des scores extrêmes. L'abaissement de votre définition de l'extrême vous permet de continuer à améliorer

l'homogénéité des performances de votre charge de travail au fur et à mesure que vous continuez à éliminer ces causes.

- Rechercher des changements importants de comportement. Un changement immédiat de quantité ou de direction d'une métrique peut indiquer qu'il y a eu un changement dans l'application ou la présence de facteurs externes pour le suivi desquels vous devez ajouter des métriques supplémentaires.

Ressources

Documents connexes :

- Exemples de requêtes Amazon CloudWatch Logs Insights
- Débogage avec Amazon CloudWatch Synthetics et AWS X-Ray
- Un atelier sur l'observabilité
- L'Amazon Builders' Library : Instrumentation des systèmes distribués au profit de la visibilité opérationnelle
- Fonctionnement des tableaux de bord Amazon CloudWatch

REL06-BP07 Surveiller la traçabilité complète des demandes via votre système

Suivez les demandes au fur et à mesure qu'elles sont traitées dans les composants du service afin que les équipes produits puissent plus facilement analyser et résoudre les problèmes et améliorer les performances.

Résultat souhaité : les charges de travail dotées d'un suivi complet de tous les composants sont faciles à déboguer, ce qui permet d'améliorer le temps moyen de résolution (MTTR) des erreurs et la latence en simplifiant la découverte de la cause première. Le traçage de bout en bout réduit le temps nécessaire à la découverte des composants concernés et à l'analyse détaillée des causes profondes des erreurs ou de la latence.

Anti-modèles courants :

- Le traçage est utilisé pour certains composants, mais pas pour tous. Par exemple, sans traçage pour AWS Lambda, les équipes risquent de ne pas comprendre clairement la latence provoquée par les démarrages à froid dans le cadre d'une charge de travail irrégulière.

- Les canaris synthétiques ou la surveillance des utilisateurs réels (RUM) ne sont pas configurés avec le traçage. Sans canaris ni RUM, la télémétrie des interactions avec le client est omise de l'analyse des traces, ce qui donne un profil de performance incomplet.
- Les charges de travail hybrides incluent à la fois des outils de suivi natifs du cloud et des outils tiers, mais aucune mesure n'a été prise pour intégrer pleinement une solution de traçage unique. En fonction de la solution de traçage sélectionnée, les kits SDK de traçage natifs du cloud doivent être utilisés pour instrumenter des composants qui ne sont pas natifs du cloud ou des outils tiers doivent être configurés pour ingérer la télémétrie de suivi native du cloud.

Avantages liés au respect de cette bonne pratique : lorsque les équipes de développement sont alertées de problèmes, elles peuvent obtenir une image complète des interactions entre les composants du système, y compris la corrélation composant par composant avec la journalisation, les performances et les défaillances. Dans la mesure où le traçage permet d'identifier visuellement les causes profondes, vous passez moins de temps à les étudier. Les équipes qui comprennent en détail les interactions entre les composants prennent de meilleures décisions plus rapidement lors de la résolution des problèmes. L'analyse des traces des systèmes permet d'améliorer la prise de décisions, par exemple quand il convient de recourir à la reprise après sinistre (DR) ou de choisir le meilleur endroit pour mettre en œuvre des stratégies d'autoréparation, ce qui permet d'améliorer la satisfaction des clients envers vos services.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyen

Directives d'implémentation

Les équipes qui exploitent des applications distribuées peuvent utiliser des outils de traçage pour établir un identifiant de corrélation, collecter des traces de demandes et créer des cartes de service pour les composants connectés. Tous les composants de l'application doivent être inclus dans les traces des demandes, notamment les clients de service, les passerelles d'intergiciels et les bus d'événements, les composants de calcul et le stockage, y compris les magasins de clés-valeurs et les bases de données. Incluez des canaris synthétiques et une surveillance des utilisateurs réels dans votre configuration de traçage de bout en bout pour mesurer les interactions avec les clients distants et la latence afin d'évaluer avec précision les performances de vos systèmes par rapport à vos contrats et objectifs de niveau de service.

Vous pouvez utiliser les services d'instrumentation d'AWS X-Ray et de la surveillance des applications Amazon CloudWatch pour fournir une vue complète des demandes au fur et à mesure qu'elles transitent par votre application. X-Ray collecte la télémétrie des applications et vous permet

de la visualiser et de la filtrer en fonction des charges utiles, des fonctions, des traces, des services et des API. Il peut être activé pour les composants du système sans code ou avec peu de code. La surveillance des applications CloudWatch inclut ServiceLens pour intégrer vos traces aux métriques, aux journaux et aux alarmes. La surveillance des applications CloudWatch inclut également des outils synthétiques pour surveiller vos points de terminaison et vos API, ainsi que la surveillance des utilisateurs réels pour instrumenter vos clients d'applications Web.

Étapes d'implémentation

- Utilisez AWS X-Ray sur tous les services natifs pris en charge, comme Amazon S3, AWS Lambda et Amazon API Gateway. Ces services AWS permettent l'utilisation de X-Ray grâce à des options de configuration utilisant l'infrastructure sous forme de code, de kits AWS SDK ou de la AWS Management Console.
- Applications des instruments AWS Distro for Open Telemetry et X-Ray ou des agents de collecte tiers.
- Consultez le Guide du développeur AWS X-Ray pour plus d'informations sur l'implémentation spécifique au langage de programmation. Ces sections de la documentation expliquent comment instrumenter les requêtes HTTP, les requêtes SQL et d'autres processus spécifiques à votre langage de programmation d'application.
- Utilisez le traçage X-Ray pour les canaris synthétiques Amazon CloudWatch et Amazon CloudWatch RUM afin d'analyser le chemin de requête depuis votre client utilisateur final via votre infrastructure AWS en aval.
- Configurez les métriques et les alarmes CloudWatch en fonction de l'intégrité des ressources et de la télémétrie canari afin que les équipes soient rapidement alertées des problèmes, puis qu'elles puissent analyser en profondeur les traces et les cartographies de services avec ServiceLens.
- Activez l'intégration de X-Ray pour les outils de traçage tiers tels que Datadog, New Relicou Dynatrace si vous utilisez des outils tiers pour votre solution de traçage principale.

Ressources

Bonnes pratiques associées :

- REL06-BP01 Surveiller tous les composants de la charge de travail (génération)
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances

Documents connexes :

- Qu'est-ce qu'AWS X-Ray ?
- Amazon CloudWatch : surveillance des applications
- Débogage avec Amazon CloudWatch Synthetics et AWS X-Ray
- Amazon Builders' Library : Instrumentation des systèmes distribués au profit de la visibilité opérationnelle
- Intégration d'AWS X-Ray à d'autres services AWS
- AWS Distro for OpenTelemetry et AWS X-Ray
- Amazon CloudWatch : utilisation de la surveillance synthétique
- Amazon CloudWatch : utilisation de CloudWatch RUM
- Configuration d'un canari synthétique Amazon CloudWatch et d'une alarme Amazon CloudWatch
- Disponibilité et plus encore : comprendre et améliorer la résilience des systèmes distribués sur AWS

Exemples connexes :

- Un atelier sur l'observabilité

Vidéos connexes :

- AWS re:Invent 2022 - How to monitor applications across multiple accounts
- How to Monitor your AWS Applications

Outils associés :

- d'AWS X-Ray
- Amazon CloudWatch
- Amazon Route 53

Concevoir votre charge de travail de sorte qu'elle s'adapte aux changements de demande

Une charge de travail scalable fournit l'élasticité nécessaire pour ajouter ou supprimer automatiquement des ressources de telle sorte qu'elles correspondent étroitement à tout moment à la demande en cours.

Bonnes pratiques

- REL07-BP01 Utiliser l'automatisation lors de l'obtention des ressources ou de leur mise à l'échelle
- REL07-BP02 Obtenir des ressources après la détection d'un problème sur une charge de travail
- REL07-BP03 Obtenir des ressources après avoir réalisé qu'un plus grand nombre de ressources est nécessaire pour une charge de travail
- REL07-BP04 Effectuer un test de charge de votre charge de travail

REL07-BP01 Utiliser l'automatisation lors de l'obtention des ressources ou de leur mise à l'échelle

Lorsque vous remplacez des ressources compromises ou que vous mettez à l'échelle votre charge de travail, automatisez le processus à l'aide de services AWS gérés comme Amazon S3 et AWS Auto Scaling. Vous pouvez également utiliser des outils tiers et les kits SDK AWS pour automatiser la mise à l'échelle.

Les services AWS gérés comprennent Amazon S3, Amazon CloudFront, AWS Auto Scaling, AWS Lambda, Amazon DynamoDB, AWS Fargate et Amazon Route 53.

AWS Auto Scaling vous permet de détecter et de remplacer les instances dégradées. Il offre également la possibilité de créer des plans de mise à l'échelle pour les ressources, notamment les instances Amazon EC2 et les parcs Spot, les tâches Amazon ECS les tables et index Amazon DynamoDB et les réplicas Amazon Aurora .

Lors de la mise à l'échelle d'instances EC2, veillez à utiliser plusieurs zones de disponibilité (de préférence au moins trois) et à ajouter ou supprimer de la capacité pour maintenir l'équilibre entre ces zones de disponibilité. Les tâches ECS ou les pods Kubernetes (lors de l'utilisation d'Amazon Elastic Kubernetes Service) doivent également être répartis sur plusieurs zones de disponibilité.

Lorsque vous utilisez AWS Lambda, la mise à l'échelle est automatique. Chaque fois qu'une notification d'événement est reçue pour votre fonction, AWS Lambda localise rapidement la capacité

disponible dans son parc de calcul et exécute votre code jusqu'à la simultanéité allouée. Vous devez vous assurer que la simultanéité nécessaire est configurée sur la fonction Lambda spécifique et dans Service Quotas.

Amazon S3 se met automatiquement à l'échelle pour gérer les débits de requêtes élevés. Par exemple, votre application peut obtenir au moins 3 500 demandes PUT/COPY/POST/DELETE ou 5 500 requêtes GET/HEAD par seconde et par préfixe partitionné dans un compartiment. Le nombre de préfixes dans un compartiment est illimité. Vous pouvez augmenter vos performances de lecture ou d'écriture en parallélisant les lectures. Par exemple, si vous créez 10 préfixes dans un compartiment Amazon S3 pour paralléliser les lectures, vous pouvez mettre à l'échelle vos performances de lecture sur 55 000 requêtes de lecture par seconde.

Configurez et utilisez Amazon CloudFront ou un réseau de diffusion de contenus (CDN) de confiance Un CDN fournit des temps de réponse plus rapides à l'utilisateur final et répond aux demandes de contenu à partir du cache, ce qui vous évite (dans une certaine mesure) de devoir adapter votre charge de travail.

Anti-modèles courants :

- Implémentation de groupes Auto Scaling pour la réparation automatique sans implémentation de l'élasticité
- Utilisation de la mise à l'échelle automatique pour répondre aux pics importants du trafic.
- Déploiement d'applications hautement dynamiques avec élimination de l'option d'élasticité.

Avantages liés au respect de cette bonne pratique : L'automatisation élimine le risque d'erreur manuelle lors du déploiement et de la mise hors service des ressources. Elle élimine aussi le risque de dépassement de coûts et de déni de service en raison d'une réponse lente aux besoins de déploiement ou de mise hors service.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

- Configurer et utiliser AWS Auto Scaling. AWS Auto Scaling permet de surveiller vos applications et d'ajuster automatiquement la capacité pour maintenir des performances stables et prévisibles au coût le plus bas possible. Avec AWS Auto Scaling, vous pouvez configurer la mise à l'échelle des applications pour plusieurs ressources sur plusieurs services.
  - Qu'est-ce qu'AWS Auto Scaling ?

    - Configurez Auto Scaling sur vos instances Amazon EC2 et vos parcs d'instances Spot, vos tâches Amazon ECS, vos tables et index Amazon DynamoDB, vos réplicas Amazon Aurora et vos appliances AWS Marketplace, le cas échéant.
        - Gestion automatique de la capacité de débit avec DynamoDB Auto Scaling.
            - Utiliser les opérations d'API de service pour spécifier les alarmes, les stratégies de mise à l'échelle, ainsi que les temps de montée et de baisse de charge
- Utiliser Elastic Load Balancing. Les équilibreurs de charge peuvent répartir la charge par chemin d'accès ou par connectivité réseau.
    - Qu'est-ce qu'Elastic Load Balancing ?
        - Les Application Load Balancers peuvent répartir la charge par chemin.
            - Qu'est-ce qu'un Application Load Balancer ?
                - Configurer un Application Load Balancer pour répartir le trafic sur différentes charges de travail selon le chemin d'accès du nom de domaine
                - Les Application Load Balancers peuvent être utilisés pour répartir les charges d'une manière qui s'intègre à AWS Auto Scaling pour gérer la demande.
                    - Utiliser un équilibreur de charge avec un groupe Auto Scaling
        - Les Network Load Balancers peuvent répartir la charge de travail par connexion.
            - Qu'est-ce qu'un Network Load Balancer ?
                - Configurer un Network Load Balancer pour répartir le trafic sur différentes charges de travail à l'aide du TCP ou disposer constamment d'un jeu d'adresses IP pour votre charge de travail
                - Les Network Load Balancers peuvent être utilisés pour répartir les charges d'une manière qui s'intègre à AWS Auto Scaling pour gérer la demande.
- Utiliser un fournisseur DNS à haut niveau de disponibilité. Les noms DNS permettent à vos utilisateurs de saisir des noms plutôt que des adresses IP pour accéder à vos charges de travail et distribuer ces informations sur une portée précise, en général mondiale, pour les utilisateurs de ces charges de travail.
    - Utiliser Amazon Route 53 ou un fournisseur DNS de confiance.
        - Qu'est-ce qu'Amazon Route 53 ?
    - Utilisez Route 53 pour gérer vos distributions CloudFront et vos équilibreurs de charge.
        - Déterminer les domaines et les sous-domaines que vous allez à gérer
        - Créez des jeux d'enregistrements appropriés à l'aide d'enregistrements ALIAS ou CNAME.

        - Utilisation des enregistrements
- Utilisez le réseau mondial AWS pour optimiser le chemin de vos utilisateurs vers vos applications. AWS Global Accelerator surveille en permanence l'état des points de terminaison de votre application et redirige le trafic vers des points de terminaison sains en moins de 30 secondes.
    - AWS Global Accelerator est un service qui améliore la disponibilité et les performances de vos applications auprès d'utilisateurs locaux ou internationaux. Il fournit des adresses IP statiques qui font office de point d'entrée fixe aux points de terminaison de votre application dans une ou plusieurs Régions AWS, telles que vos Application Load Balancers, vos Network Load Balancers ou vos instances Amazon EC2.
        - Qu'est-ce qu'AWS Global Accelerator ?
- Configurez et utilisez Amazon CloudFront ou un réseau de diffusion de contenus (CDN) de confiance Un réseau de diffusion de contenus peut fournir des temps de réponse des utilisateurs finaux plus rapides et traiter les requêtes de contenu susceptibles de causer une mise à l'échelle inutile de vos charges de travail.
    - Qu'est-ce que Amazon CloudFront ?
        - Configurez les distributions Amazon CloudFront pour vos charges de travail ou utilisez un CDN tiers.
            - Vous pouvez limiter l'accès à vos charges de travail de sorte qu'elles ne soient accessibles qu'à partir de CloudFront. Pour ce faire, vous pouvez utiliser les plages d'adresses IP pour CloudFront dans vos groupes de sécurité ou vos politiques d'accès des points de terminaison.

Ressources

Documents connexes :

- Partenaire APN : partenaires pouvant vous aider à créer des solutions de calcul automatisées
- AWS Auto Scaling : Fonctionnement des plans de mise à l'échelle
- AWS Marketplace : produits utilisables avec Auto Scaling
- Gestion automatique de la capacité de débit avec DynamoDB Auto Scaling
- Utiliser un équilibreur de charge avec un groupe Auto Scaling
- Qu'est-ce qu'AWS Global Accelerator ?
- Qu'est-ce que Amazon EC2 Auto Scaling ?
- Qu'est-ce qu'AWS Auto Scaling ?

- Qu'est-ce que Amazon CloudFront ?
- Qu'est-ce qu'Amazon Route 53 ?
- Qu'est-ce qu'Elastic Load Balancing ?
- Qu'est-ce qu'un Network Load Balancer ?
- Qu'est-ce qu'un Application Load Balancer ?
- Utilisation des enregistrements

REL07-BP02 Obtenir des ressources après la détection d'un problème sur une charge de travail

Si la disponibilité est affectée, mettez à l'échelle les ressources de manière réactive si nécessaire, afin de restaurer la disponibilité de la charge de travail.

Vous devez commencer par configurer les vérifications de l'état et les critères de ces vérifications pour indiquer quand la disponibilité est affectée par le manque de ressources. Informez ensuite le personnel approprié qu'il doit mettre à l'échelle manuellement la ressource ou lancer l'automatisation pour procéder à une mise à l'échelle automatique.

La mise à l'échelle peut être ajustée manuellement en fonction de votre charge de travail. Par exemple, vous pouvez modifier le nombre d'instances EC2 dans un groupe Auto Scaling ou le débit d'une table DynamoDB via la AWS Management Console ou AWS CLI). Toutefois, l'automatisation doit être utilisée à chaque fois que c'est possible (voir Utiliser l'automatisation lors de l'obtention des ressources ou de leur mise à l'échelle).

Résultat souhaité : des opérations de mise à l'échelle (automatique ou manuelle) sont lancées pour rétablir la disponibilité dès la détection d'une panne ou d'une dégradation de l'expérience client.

Niveau de risque exposé si cette bonne pratique n'est pas établie: moyen

Directives d'implémentation

Mettez en œuvre l'observabilité et la surveillance de tous les composants de votre charge de travail, afin de surveiller l'expérience client et de détecter les défaillances. Définissez les procédures, manuelles ou automatisées, de mise à l'échelle des ressources requises. Pour plus d'informations, consultez REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances.

Étapes d'implémentation

- Définissez les procédures, manuelles ou automatisées, de mise à l'échelle des ressources requises.
  - Les procédures de mise à l'échelle dépendent de la conception des différents composants de votre charge de travail.
  - Les procédures de mise à l'échelle varient également en fonction de la technologie sous-jacente utilisée.
    - Les composants qui utilisent AWS Auto Scaling peuvent utiliser des plans de dimensionnement pour configurer un ensemble d'instructions pour la mise à l'échelle de vos ressources. Si vous travaillez avec AWS CloudFormation ou que vous ajoutez des balises à des ressources AWS, vous pouvez configurer des plans de dimensionnement pour différents ensembles de ressources par application. Auto Scaling offre des recommandations de stratégies de mise à l'échelle personnalisées pour chaque ressource. Une fois que vous avez créé votre plan de dimensionnement, Auto Scaling combine des méthodes de mise à l'échelle dynamique et prédictive pour prendre en charge votre stratégie de mise à l'échelle. Pour plus d'informations, consultez Fonctionnement des plans de dimensionnement.
    - Amazon EC2 Auto Scaling vérifie que vous disposez du nombre adéquat d'instances Amazon EC2 disponibles pour gérer la charge de votre application. Vous créez des collections d'instances EC2, appelées groupes Auto Scaling. Vous pouvez spécifier le nombre minimal et maximal d'instances dans chaque groupe Auto Scaling et Amazon EC2 Auto Scaling s'assure que votre groupe ne dépasse jamais ces limites. Pour plus d'informations, consultez What is Amazon EC2 Auto Scaling?
    - La mise à l'échelle automatique d'Amazon DynamoDB utilise le service Application Auto Scaling pour ajuster de manière dynamique la capacité de débit alloué en votre nom, en fonction des modèles de trafic réels. Cela permet à une table ou à un index secondaire global d'augmenter sa capacité de lecture et d'écriture allouée afin de gérer sans limitations les augmentations soudaines du trafic. Pour plus d'informations, consultez Gestion automatique de la capacité de débit avec la scalabilité automatique de DynamoDB.

Ressources

Bonnes pratiques associées :

- REL07-BP01 Utiliser l'automatisation lors de l'obtention des ressources ou de leur mise à l'échelle
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances

Documents connexes :

- AWS Auto Scaling: Fonctionnement des plans de dimensionnement
- Gestion automatique de la capacité de débit avec la scalabilité automatique de DynamoDB
- What Is Amazon EC2 Auto Scaling?

REL07-BP03 Obtenir des ressources après avoir réalisé qu'un plus grand nombre de ressources est nécessaire pour une charge de travail

Mettez à l'échelle les ressources de manière proactive pour répondre à la demande et éviter l'impact sur la disponibilité.

De nombreux services AWS sont automatiquement mis à l'échelle pour répondre à la demande. Si vous utilisez des instances Amazon EC2 ou des clusters Amazon ECS, vous pouvez configurer la mise à l'échelle automatique de ces instances pour qu'elle intervienne en fonction des métriques d'utilisation qui correspondent à la demande de votre charge de travail. Pour Amazon EC2, l'utilisation moyenne du CPU, le nombre de requêtes de l'équilibreur de charge ou la bande passante du réseau peuvent être utilisés pour augmenter (ou diminuer) les instances EC2. Pour Amazon ECS, l'utilisation moyenne du CPU, le nombre de requêtes de l'équilibreur de charge et l'utilisation de la mémoire peuvent être utilisés pour augmenter (ou diminuer) les tâches ECS. En utilisant Target Auto Scaling sur AWS, l'Autoscaler agit comme un thermostat domestique, en ajoutant ou en supprimant des ressources pour maintenir la valeur cible (par exemple, 70 % d'utilisation du CPU) que vous spécifiez.

AWS Auto Scaling peut également exécuter Predictive Auto Scaling, qui s'appuie sur le machine learning pour analyser la charge de travail historique de chaque ressource et anticipe régulièrement la charge future des deux prochains jours.

Little's Law permet de calculer le nombre d'instances de calcul (instances EC2, fonctions Lambda simultanées, etc.) dont vous avez besoin.

$L = \lambda W$

L = nombre d'instances (ou simultanéité moyenne dans le système)

λ = vitesse moyenne à laquelle les requêtes arrivent (demande/s.)

W = temps moyen que chaque requête passe dans le système (s.)

Par exemple, à 100 rps, si le traitement de chaque requête prend 0,5 seconde, vous aurez besoin de 50 instances pour prendre en charge la requête.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

- Obtenir des ressources après avoir réalisé qu'un plus grand nombre de ressources est nécessaire pour une charge de travail. Mettez à l'échelle les ressources de manière proactive pour répondre à la demande et éviter l'impact sur la disponibilité.
  - Déterminez le nombre de ressources de calcul dont vous aurez besoin (simultanéité de calcul) pour gérer un débit de demandes donné.
    - Telling Stories About Little's Law
  - Configurez la mise à l'échelle planifiée pour Amazon EC2 Auto Scaling lorsque vous disposez d'un modèle d'utilisation historique.
    - Mise à l'échelle planifiée pour Amazon EC2 Auto Scaling
  - Utilisez la mise à l'échelle prédictive AWS.
    - Scalabilité prédictive pour EC2 alimentée par le machine learning

Ressources

Documents connexes :

- AWS Auto Scaling : Fonctionnement des plans de mise à l'échelle
- AWS Marketplace : produits utilisables avec Auto Scaling
- Gestion automatique de la capacité de débit avec DynamoDB Auto Scaling
- Scalabilité prédictive pour EC2 alimentée par le machine learning
- Mise à l'échelle planifiée pour Amazon EC2 Auto Scaling
- Telling Stories About Little's Law
- Qu'est-ce que Amazon EC2 Auto Scaling ?

REL07-BP04 Effectuer un test de charge de votre charge de travail

Adoptez une méthodologie de test de charge pour déterminer si la mise à l’échelle répond aux exigences de la charge de travail.

Il est important d'exécuter régulièrement des tests de charge. Les tests de charge devraient découvrir le point de rupture et tester les performances de votre charge de travail. AWS facilite la configuration d'environnements de test temporaires qui modélisent l'échelle de votre charge de travail de production. Dans le Cloud, vous pouvez créer un environnement d'essai à l'échelle de la production et à la demande, exécuter les tests, puis désactiver les ressources. Puisque vous ne payez l'environnement de test que lorsqu'il s'exécute, vous pouvez simuler votre environnement réel pour une fraction du coût d'un test sur site.

Les tests de charge en production doivent également être intégrés aux tests de simulation de pannes, lors desquels le système de production est mis sous tension pendant les périodes où le client est moins utilisé et tout le personnel est disponible pour interpréter les résultats et résoudre les problèmes qui surviennent.

Anti-modèles courants :

- Exécution de tests de charge sur des déploiements qui ne n'ont pas la même configuration que votre production.
- Effectuer un test de charge uniquement sur des éléments individuels de votre charge de travail, et non sur l'ensemble de la charge de travail.
- Exécution de tests de charge avec un sous-ensemble de demandes et non un ensemble représentatif de demandes réelles.
- Exécution de tests de charge avec un faible facteur de sécurité au-dessus de la charge prévue.

Avantages liés au respect de cette bonne pratique : Vous savez quels composants de votre architecture échouent sous charge et vous pouvez identifier les métriques à surveiller qui indiquent suffisamment à temps que vous approchez de cette charge pour que vous résolviez le problème et empêchiez ainsi l'impact de cette défaillance.

Niveau d'exposition au risque si cette bonne pratique n'est pas respectée : Moyen

Directives d'implémentation

- Exécutez des tests de charge pour identifier l'aspect de votre charge de travail qui indique que vous devez ajouter ou supprimer de la capacité. Les tests de charge doivent avoir un trafic représentatif similaire à ce que vous recevez en production. Augmentez la charge tout en surveillant les métriques que vous avez instrumentées pour déterminer quelle métrique indique quand vous devez ajouter ou supprimer des ressources.
  - Test de charge distribuée sur AWS : simulation de milliers d'utilisateurs connectés

- Identifiez le mélange de demandes. Comme vous pouvez avoir divers mélanges de demandes, vous devez examiner les différentes périodes lors de l'identification de la combinaison de trafic.
- Implémentez un pilote de charge. Vous pouvez utiliser un code personnalisé, un logiciel open source ou un logiciel commercial pour implémenter un pilote de charge.
- Effectuez un test de charge initial avec une faible capacité. Vous constatez des effets immédiats en entraînant une charge moindre, éventuellement aussi petite qu'une instance ou un conteneur.
- Effectuez un test de charge par rapport à une capacité plus importante. Étant donné que les effets seront différents sur une charge distribuée, vous devez procéder à des essais dans un environnement aussi proche que possible de celui du produit.

Ressources

Documents connexes :

- Test de charge distribuée sur AWS : simulation de milliers d'utilisateurs connectés
- Applications de test de charge

Vidéos connexes :

- AWS Summit ANZ 2023: Accelerate with confidence through AWS Distributed Load Testing

Implémentation de la modification

Des modifications contrôlées sont nécessaires pour déployer de nouvelles fonctionnalités et garantir que les charges de travail et l'environnement d'exploitation exécutent des logiciels connus et correctement corrigés. Si les modifications ne sont pas contrôlées, il est difficile de prédire leur effet ou de résoudre les problèmes qui en découlent.

Bonnes pratiques

- REL08-BP01 Utiliser des runbooks pour les activités standard telles que le déploiement
- REL08-BP02 Intégrer les tests fonctionnels dans le cadre de votre déploiement
- REL08-BP03 Intégrer les tests de résilience dans le cadre de votre déploiement
- REL08-BP04 Effectuer le déploiement à l'aide d'une infrastructure immuable

- REL08-BP05 Déployer les modifications avec l'automatisation

REL08-BP01 Utiliser des runbooks pour les activités standard telles que le déploiement

Les runbooks sont les procédures prédéfinies destinées à parvenir à un résultat spécifique. Utilisez des runbooks pour effectuer des tâches manuelles ou automatiques standard. Il peut s'agir du déploiement d'une charge de travail, de l'application de correctifs à une charge de travail ou de la modification du DNS.

Par exemple, mettez en place des processus pour assurer la sécurité des restaurations pendant les déploiements. Pour garantir la fiabilité d'un service, il est essentiel de s'assurer que vous pouvez restaurer un déploiement sans interruption pour vos clients.

Concernant les procédures de runbook, commencez par un processus manuel efficace valide, mettez-le en œuvre dans le code et, le cas échéant, déclenchez son exécution automatique.

Même pour les charges de travail sophistiquées hautement automatisées, les runbooks restent utiles pour exécuter des tests de simulation de pannes ou répondre à des exigences rigoureuses en matière de rapports et d'audit.

Notez que les playbooks sont utilisés en réponse à des incidents spécifiques et que les runbooks le sont pour obtenir des résultats spécifiques. En règle générale, les runbooks sont destinés aux activités de routine, tandis que les playbooks sont utilisés pour répondre à des événements non réguliers.

Anti-modèles courants :

- Exécution de modifications imprévues de la configuration en production.
- Ignorer les étapes de votre plan afin d'accélérer le déploiement, ce qui entraîne un échec du déploiement.
- Effectuez des modifications sans tester l'annulation de la modification.

Avantages liés au respect de cette bonne pratique : Une planification efficace des modifications augmente votre capacité à exécuter correctement la modification, car vous êtes conscient de tous les systèmes concernés. Vous gagnez en confiance si vous réussissez à valider des modifications que vous apportez aux environnements de test.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

- Obtenez des réponses cohérentes et rapides à des événements bien compris en documentant les procédures dans des runbooks.
  - Concepts AWS Well-Architected Framework : runbook
- Utilisez le principe de l'infrastructure en tant que code pour définir votre infrastructure. En ayant recours à AWS CloudFormation ou à un tiers de confiance pour définir votre infrastructure, vous pouvez utiliser le contrôle de version et suivre les modifications apportées à la version du logiciel.
  - Utilisez AWS CloudFormation ou un fournisseur tiers de confiance pour définir votre infrastructure.
    - Qu'est-ce qu'AWS CloudFormation ?
  - Créez des modèles qui sont singuliers et découplés, en utilisant de bons principes de conception de logiciels.
    - Déterminez les autorisations, les modèles et les responsables de l'implémentation
      - Contrôle de l'accès avec AWS Identity and Access Management
    - Utilisez le contrôle de code source, comme AWS CodeCommit ou un outil tiers de confiance, pour le contrôle de version.
      - Qu'est-ce qu'AWS CodeCommit ?

Ressources

Documents connexes :

- Partenaire APN : partenaires pouvant vous aider à créer des solutions de déploiement automatisées
- AWS Marketplace : produits pouvant être utilisés pour automatiser vos déploiements
- Concepts AWS Well-Architected Framework : runbook
- Qu'est-ce qu'AWS CloudFormation ?
- Qu'est-ce qu'AWS CodeCommit ?

Exemples connexes :

- Automatisation des opérations avec les playbooks et les runbooks

REL08-BP02 Intégrer les tests fonctionnels dans le cadre de votre déploiement

Les tests fonctionnels sont exécutés dans le cadre du déploiement automatisé. Si les critères de réussite ne sont pas respectés, le pipeline est arrêté ou annulé. Ces tests sont exécutés dans un environnement de préproduction, qui est mis en place avant la production dans le pipeline. Idéalement, cela s'effectue dans le cadre d'un pipeline de déploiement.

Résultat souhaité : vous utilisez l'automatisation pour effectuer des tests fonctionnels, et les données de test associées réduisent la durée et les dépenses des tests et améliorent la précision des résultats des tests. Vous intégrez des tests fonctionnels dans le cadre de votre processus de déploiement, ce qui vous permet d'automatiser vos pipelines de publication pour des mises à jour rapides et fiables des applications et de l'infrastructure.

Anti-modèles courants :

- Vous effectuez manuellement des tests en dehors du pipeline de déploiement.
- Vous sautez les étapes de test de votre automatisation grâce à des flux de travail d'urgence manuels.
- Vous ne suivez pas les plans et processus de test établis au profit de délais accélérés.

Avantages de la mise en place de cette bonne pratique : les tests fonctionnels confirment que le système fonctionne conformément aux exigences spécifiées. Il est utilisé pour vérifier de manière cohérente l'état de fonctionnement prévu des composants tels que les interfaces utilisateur, les API, les bases de données et le code source. Lorsque vous examinez ces composants du système, les tests fonctionnels vérifient que chaque fonctionnalité se comporte comme prévu, ce qui protège à la fois les attentes des utilisateurs et l'intégrité du logiciel. Intégrez des tests fonctionnels dans le cadre de votre déploiement régulier et utilisez l'automatisation pour déployer toutes les modifications, ce qui réduit le risque d'introduction d'erreurs humaines.

Niveau de risque exposé si cette bonne pratique n'est pas établie : élevé

Directives d'implémentation

Intégrez les tests fonctionnels dans le cadre de votre déploiement. Les tests fonctionnels sont exécutés dans le cadre du déploiement automatisé. Si les critères de réussite ne sont pas remplis, le pipeline est arrêté ou annulé. AWS CodePipeline fournit un pipeline de livraison continue pour les tests automatisés, ce qui permet aux testeurs d'automatiser l'ensemble du processus de test et

de déploiement. Il s'intègre à des services AWS comme AWS CodeBuild et AWS CodeDeploy pour automatiser les phases de création, de test et de déploiement du cycle de vie du développement logiciel.

Étapes d'implémentation

- Configurez votre pipeline : configurez les étapes de récupération de code source, de génération, de test et de déploiement à l'aide de la console AWS CodePipeline ou de l'AWS Command Line Interface (CLI).
  - Définissez votre source : avec AWS CodePipeline, vous pouvez récupérer automatiquement le code source à partir de systèmes de contrôle de version tels que GitHub, AWS CodeCommit ou Bitbucket, ce qui permet de confirmer que le code le plus récent est toujours utilisé pour les tests.
  - Automatisez les builds et les tests : AWS CodeBuild peut automatiquement créer et tester votre code et générer des rapports de test. Il prend en charge les frameworks de test populaires tels que JUnit, NUnit et TestNG.
  - Déployez votre code : une fois que le code a été créé et testé, AWS CodeDeploy peut le déployer dans votre environnement de test, y compris des instances Amazon EC2, des fonctions AWS Lambda ou des serveurs sur site.
  - Surveillez les pipelines : AWS CodePipeline peut suivre la progression de votre pipeline et l'état de chaque étape. Vous pouvez également utiliser des contrôles de qualité pour bloquer le pipeline en fonction de l'état d'exécution du test. Vous pouvez aussi recevoir des notifications en cas d'échec ou d'achèvement d'une étape du pipeline.

Ressources

Documents connexes :

- Utilisez AWS CodePipeline avec AWS CodeBuild pour tester le code et exécuter des générations
- Journalisation et surveillance dans AWS CodeBuild
- Indicateurs pour les tests fonctionnels

REL08-BP03 Intégrer les tests de résilience dans le cadre de votre déploiement

Intégrez des tests de résilience en introduisant consciemment des défaillances dans le système afin de mesurer sa fonctionnalité en cas de scénarios perturbateurs. Les tests de résilience sont différents

des tests unitaires et fonctionnels qui sont généralement intégrés dans les cycles de déploiement, car ils se concentrent sur l'identification des défaillances imprévues de votre système. Bien qu'il soit prudent de commencer par l'intégration des tests de résilience en pré-production, fixez-vous comme objectif d'implémenter ces tests en production dans le cadre de vos journées de simulation.

Résultat souhaité : les tests de résilience contribuent à renforcer la confiance dans la capacité du système à résister à la dégradation en cours de production. Les expériences identifient les points faibles susceptibles d'entraîner une défaillance, ce qui vous permet d'améliorer le système afin d'atténuer automatiquement et efficacement les défaillances et la dégradation.

Anti-modèles courants :

- Manque d'observabilité et de surveillance dans les processus de déploiement
- Dépendance à l'humain pour résoudre les défaillances du système
- Mécanismes d'analyse de mauvaise qualité
- Accent mis sur les problèmes connus d'un système et manque d'expérimentation pour identifier les problèmes inconnus
- Identification des défaillances, mais pas de solution
- Aucune documentation sur les résultats ni aucun runbook

Avantages de la mise en place de cette bonne pratique : les tests de résilience intégrés à vos déploiements permettent d'identifier les problèmes système inconnus qui, autrement, passeraient inaperçus, ce qui pourrait entraîner des interruptions en production. L'identification de ces problèmes système inconnus vous permet de documenter les résultats, d'intégrer les tests dans votre processus CI/CD et de créer des runbooks, ce qui simplifie leur atténuation grâce à des mécanismes efficaces et reproductibles.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

Les formes de test de résilience les plus courantes pouvant être intégrées dans les déploiements de votre système sont la reprise après sinistre et l'ingénierie du chaos.

- Incluez des mises à jour de vos plans de reprise après sinistre et de vos procédures opérationnelles standard (SOP) lors de tout déploiement important.

- Intégrez des tests de fiabilité à vos pipelines de déploiement automatisés. Des services comme AWS Resilience Hub peuvent être intégrés à votre pipeline CI/CD pour établir des évaluations de résilience continues qui sont automatiquement évaluées dans le cadre de chaque déploiement.
- Définissez vos applications dans AWS Resilience Hub. Les évaluations de résilience génèrent des extraits de code qui vous aident à créer des procédures de restauration sous forme de documents AWS Systems Manager pour vos applications et fournissent une liste de moniteurs et d'alarmes Amazon CloudWatch recommandés.
- Une fois vos plans de reprise après sinistre et vos SOP mis à jour, effectuez des tests de reprise après sinistre pour vérifier leur efficacité. Les tests de reprise après sinistre contribuent à déterminer si vous pouvez restaurer votre système après un événement et revenir à un fonctionnement normal. Vous pouvez simuler différentes stratégies de reprise après sinistre et déterminer si votre planification est suffisante pour répondre à vos exigences de disponibilité. Les stratégies courantes de reprise après sinistre incluent la sauvegarde et la restauration, l'environnement en veille, la veille à froid, la veille à chaud, la veille permanente et la veille active/active. Elles diffèrent toutes en termes de coût et de complexité. Avant les tests de reprise après sinistre, nous vous recommandons de définir votre objectif de temps de restauration (RTO) et votre objectif de point de reprise (RPO) afin de simplifier le choix de la stratégie à simuler. AWS propose des outils de reprise après sinistre comme AWS Elastic Disaster Recovery pour vous aider à démarrer votre planification et vos tests.
- Les expériences d'ingénierie du chaos introduisent des perturbations dans le système, telles que des pannes de réseau et des pannes de service. En simulant le système avec des pannes contrôlées, vous pouvez en découvrir les vulnérabilités tout en limitant les impacts des pannes injectées. Tout comme les autres stratégies, exécutez des simulations de pannes contrôlées dans des environnements hors production à l'aide de services comme AWS Fault Injection Service pour gagner en confiance avant le déploiement en production.

Ressources

Documents connexes :

- Experiment with failure using resilience testing to build recovery preparedness
- Continually assessing application resilience with AWS Resilience Hub and AWS CodePipeline
- Architecture de reprise après sinistre (DR) sur AWS, première partie : stratégies de reprise dans le cloud
- Verify the resilience of your workloads using Chaos Engineering

- Principes de l’ingénierie du chaos
- Atelier sur l’ingénierie du chaos

Vidéos connexes :

- AWS re:Invent 2020: Testing Resilience using Chaos Engineering
- Improve Application Resilience with AWS Fault Injection Service
- Prepare & Protect Your Applications From Disruption With AWS Resilience Hub

REL08-BP04 Effectuer le déploiement à l'aide d'une infrastructure immuable

Une infrastructure immuable est un modèle qui exige qu'aucune mise à jour, aucune application de correctifs de sécurité ni aucun changement de configuration ne se produise sur place sur les charges de travail de production. Lorsqu'un changement est nécessaire, l'architecture est intégrée à la nouvelle infrastructure et déployée en production.

Suivez une stratégie de déploiement d'infrastructure immuable pour améliorer la fiabilité, la cohérence et la reproductibilité de vos déploiements de charges de travail.

Résultat souhaité : avec une infrastructure immuable, aucune modification sur place n'est autorisée pour exécuter les ressources d'infrastructure au sein d'une charge de travail. Lorsqu'une modification est nécessaire, un nouvel ensemble de ressources d'infrastructure contenant toutes les modifications nécessaires est déployé parallèlement à vos ressources existantes. Ce déploiement est validé automatiquement et, en cas de succès, le trafic est progressivement transféré vers ce nouvel ensemble de ressources.

Cette stratégie de déploiement s'applique notamment aux mises à jour logicielles, aux correctifs de sécurité, aux modifications de l'infrastructure, ainsi qu'aux mises à jour de la configuration et des applications.

Anti-modèles courants :

- Modifications sur place des ressources d'infrastructure en cours d'exécution.

Avantages liés à l'instauration de cette bonne pratique :

- Plus grande cohérence entre les environnements : comme les ressources d'infrastructure ne diffèrent pas d'un environnement à l'autre, la cohérence est renforcée et les tests sont simplifiés.

- Réduction des écarts de configuration : en remplaçant les ressources d'infrastructure par une configuration connue et dont la version est contrôlée, l'infrastructure se trouve dans un état connu, testé et fiable, ce qui permet d'éviter les écarts de configuration.
- Déploiements atomiques fiables : soit les déploiements se déroulent avec succès, soit rien ne change, ce qui accroît la cohérence et la fiabilité du processus de déploiement.
- Déploiements simplifiés : les déploiements sont simplifiés, car ils n'ont pas besoin de prendre en charge les mises à niveau. Les mises à niveau sont simplement de nouveaux déploiements.
- Déploiements plus sûrs avec des processus de restauration et de récupération rapides : les déploiements sont plus sûrs, car la version de travail précédente n'est pas modifiée. Vous pouvez la restaurer si des erreurs sont détectées.
- Niveau de sécurité renforcé : l'impossibilité de modifier l'infrastructure permet de désactiver les mécanismes d'accès à distance (comme SSH). Vous pouvez ainsi réduire les vecteurs d'attaque tout en renforçant la sécurité de votre organisation.

Niveau de risque exposé si cette bonne pratique n'est pas établie: moyen

Directives d'implémentation

Automatisation

Lors de la définition d'une stratégie de déploiement d'infrastructure immuable, il est recommandé d'utiliser l'automatisation autant que possible afin d'améliorer la reproductibilité et de minimiser le risque d'erreur humaine. Pour plus d'informations, consultez REL08-BP05 Déployer les modifications avec l'automatisation et Automatisation de déploiements sécurisés sans intervention.

Avec l'infrastructure en tant que code (IaC), les étapes de provisionnement, d'orchestration et de déploiement de l'infrastructure sont définies de manière programmatique, descriptive et déclarative et stockées dans un système de contrôle de source. L'utilisation de l'infrastructure en tant que code simplifie l'automatisation du déploiement de l'infrastructure et contribue à garantir l'immuabilité de cette dernière.

Schémas de déploiement

Lorsqu'une modification de la charge de travail est requise, la stratégie de déploiement d'infrastructure immuable impose le déploiement d'un nouvel ensemble de ressources d'infrastructure comprenant toutes les modifications nécessaires. Il est important que ce nouvel ensemble de ressources suive un schéma de déploiement qui minimise l'impact sur les utilisateurs. Il existe deux stratégies principales pour ce type de déploiement :

Déploiement canary : il consiste à diriger un petit nombre de vos clients vers la nouvelle version, généralement exécutée sur une seule instance de service (canary). Examinez ensuite en profondeur les modifications de comportement ou les erreurs générées. Vous pouvez supprimer le trafic du canary si vous rencontrez des problèmes critiques et faire basculer les utilisateurs vers la version précédente. Si le déploiement réussit, vous pouvez le poursuivre à la vitesse souhaitée, tout en surveillant les modifications afin de détecter les erreurs, jusqu'à ce qu'il soit terminé. AWS CodeDeploy peut être configuré avec une configuration de déploiement autorisant un déploiement canary.

Déploiement bleu/vert : il est semblable au déploiement canary, à la différence qu'un parc complet de l'application est déployé en parallèle. Vos déploiements alternent entre deux piles (bleu et vert). Une fois encore, vous pouvez faire basculer le trafic vers la nouvelle version et revenir à l'ancienne si vous rencontrez des problèmes lors du déploiement. Généralement, tout le trafic est basculé en même temps, mais vous pouvez également orienter des fractions de votre trafic vers chaque version pour accélérer l'adoption de la nouvelle version en utilisant les capacités de routage DNS pondéré d'Amazon Route 53. AWS CodeDeploy et AWS Elastic Beanstalk peuvent être configurés avec une configuration de déploiement autorisant un déploiement bleu/vert.

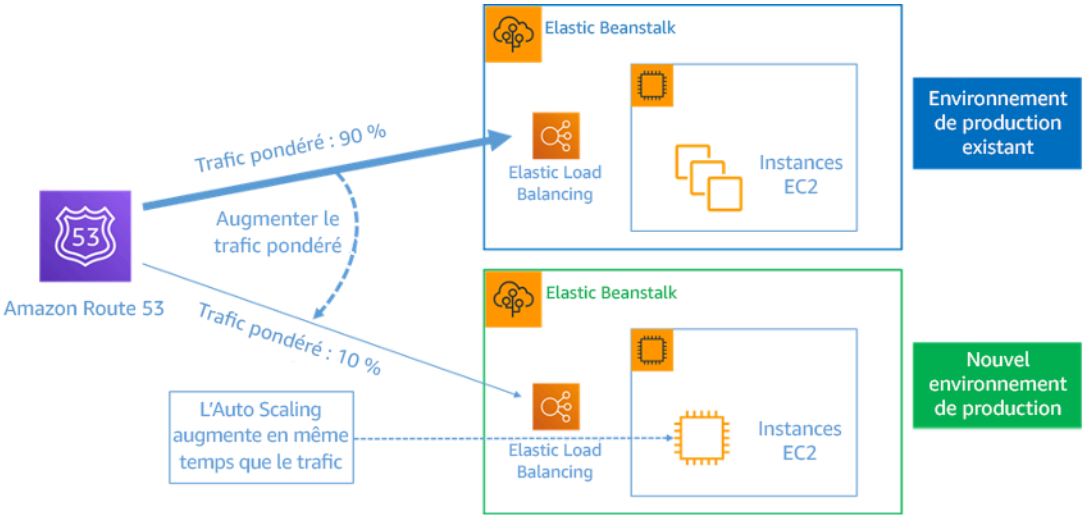


Figure 8 : Déploiement bleu/vert avec AWS Elastic Beanstalk et Amazon Route 53

Détection d'écart

Un écart est un changement qui entraîne un état ou une configuration d'une ressource d'infrastructure différent de celui attendu. Toute modification de configuration non gérée va à l'encontre de la notion d'infrastructure immuable et doit être détectée et corrigée afin de garantir la mise en œuvre d'une infrastructure immuable.

Étapes d'implémentation

- Interdisez la modification sur place des ressources d'infrastructure en cours d'exécution.
  - Vous pouvez utiliser AWS Identity and Access Management (IAM) pour spécifier les personnes ou les entités qui peuvent accéder aux services et aux ressources dans AWS, centraliser la gestion des autorisations précises et analyser les accès pour affiner les autorisations dans AWS
- Automatisez le déploiement des ressources d'infrastructure pour améliorer la reproductibilité et minimiser le risque d'erreur humaine.
  - Comme décrit dans le livre blanc Présentation de DevOps sur AWS, l'automatisation est la pierre angulaire des services AWS et elle est prise en charge en interne dans l'ensemble des services, fonctionnalités et offres.
  - La préparation de votre Amazon Machine Image (AMI) peut accélérer leur lancement. EC2 Image Builder est un service AWS entièrement géré qui vous aide à automatiser la création, la maintenance, la validation, le partage et le déploiement d'une AMI Linux ou Windows personnalisée, sécurisée et à jour.
  - Les services qui prennent en charge l'automatisation incluent :
    - AWS Elastic Beanstalk est un service permettant de déployer et de mettre à l'échelle rapidement des applications Web développées avec Java, .NET, PHP, Node.js, Python, Ruby, Go et Docker sur des serveurs familiers tels qu'Apache, NGINX, Passenger et IIS.
    - AWS Proton aide les équipes de plateforme à connecter et à coordonner tous les différents outils dont vos équipes de développement ont besoin pour le provisionnement de l'infrastructure, les déploiements de code, la surveillance et les mises à jour. AWS Proton permet d'automatiser le provisionnement de l'infrastructure en tant que code et le déploiement d'applications sans serveur et basées sur des conteneurs.
  - L'utilisation d'une infrastructure en tant que code facilite l'automatisation du déploiement de l'infrastructure et contribue à garantir l'immuabilité de l'infrastructure. AWS fournit des services de création, de déploiement et de maintenance programmatique, descriptive et déclarative de l'infrastructure.
    - AWS CloudFormation aide les développeurs à créer des ressources AWS de façon ordonnée et prévisible. Les ressources sont écrites dans des fichiers texte au format JSON ou YAML. Les modèles nécessitent une syntaxe et une structure spécifiques, qui dépendent des types de ressources créées et gérées. Vous créez vos ressources au format JSON ou YAML avec n'importe quel éditeur de code AWS Cloud9, vous les archivez dans un système de contrôle de version, puis CloudFormation crée les services spécifiés de manière sûre et reproductible.

    - AWS Serverless Application Model (AWS SAM) est un cadre open source que vous pouvez utiliser pour créer des applications sans serveur sur AWS. AWS SAM s'intègre à d'autres services AWS, et est une extension de AWS CloudFormation.
    - AWS Cloud Development Kit (AWS CDK) est un cadre de développement logiciel open source permettant de modéliser et de provisionner les ressources de vos applications cloud à l'aide de langages de programmation familiers. Vous pouvez utiliser AWS CDK pour modéliser l'infrastructure d'applications avec TypeScript, Python, Java et .NET. AWS CDK utilise AWS CloudFormation en arrière-plan pour provisionner les ressources de manière sécurisée et reproductible.
    - AWS Cloud Control API introduit un ensemble commun d'API CRUDL (Create, Read, Update, Delete, and List) pour aider les développeurs à gérer leur infrastructure cloud de façon simple et cohérente. Les API courantes de Cloud Control API permettent aux développeurs de gérer de manière uniforme le cycle de vie des services AWS et tiers.
- Mettez en œuvre des modèles de déploiement qui minimisent l'impact sur les utilisateurs.
    - Déploiements canary :
        - Configuration d'un déploiement de la version canary API Gateway
        - Create a pipeline with canary deployments for Amazon ECS using AWS App Mesh
    - Déploiements bleu/vert : le livre blanc Blue/Green Deployments on AWS décrit des exemples de techniques pour mettre en œuvre des stratégies de déploiement bleu/vert.
- Détectez les écarts de configuration ou d'état. Pour plus d'informations, consultez Détection de modifications non gérées de la configuration des piles et des ressources.

Ressources

Bonnes pratiques associées :

- REL08-BP05 Déployer les modifications avec l'automatisation

Documents connexes :

- Automatisation de déploiements sécurisés sans intervention
- Leveraging AWS CloudFormation to create an immutable infrastructure at Nubank
- Infrastructure en tant que code
- Implementing an alarm to automatically detect drift in AWS CloudFormation stacks

Vidéos connexes :

- AWS re:Invent 2020: Reliability, consistency, and confidence through immutability

REL08-BP05 Déployer les modifications avec l'automatisation

Les déploiements et l'application de correctifs sont automatisés pour éliminer l'impact négatif.

Les modifications apportées aux systèmes de production sont l'un des secteurs de risque les plus importants pour de nombreuses organisations. Nous considérons les déploiements comme un problème de premier ordre à résoudre, tout comme les problèmes opérationnels que le logiciel rencontre. Aujourd'hui, il convient d'appliquer l'automatisation dès que les opérations le permettent, y compris lors des tests et du déploiement de modifications, lors de l'ajout ou de la suppression de capacités et lors de la migration des données.

Résultat souhaité : vous intégrez la sécurité des déploiements automatisés au processus de publication grâce à des tests de pré-production approfondis, à des annulations automatiques et à des déploiements de production échelonnés. Cette automatisation minimise l'impact potentiel de l'échec des déploiements sur la production, et les développeurs n'ont plus besoin de surveiller activement les déploiements jusqu'à la production.

Anti-modèles courants :

- Vous effectuez des modifications manuelles.
- Vous sautez des étapes de l'automatisation grâce à des flux de travail d'urgence manuels.
- Vous ne suivez pas les plans et processus établis au profit de délais accélérés.
- Vous effectuez des déploiements de suivi rapides sans prévoir de durée d'intégration.

Avantages de la mise en place de cette bonne pratique : lorsque vous utilisez l'automatisation pour déployer toutes les modifications, vous supprimez le risque d'erreur humaine et vous offrez la possibilité de réaliser des tests avant de modifier la production. L'exécution de ce processus avant la phase de production permet de vérifier que vos plans sont complets. En outre, la restauration automatique de votre processus de publication peut identifier les problèmes de production et ramener votre charge de travail à son état de fonctionnement antérieur.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

Automatisez votre pipeline de déploiement. Le déploiement des pipelines vous permet d'une part d'invoquer des tests automatisés et la détection des anomalies et, d'autre part, d'arrêter le pipeline à une certaine étape avant le déploiement en production ou de restaurer automatiquement l'environnement d'avant la modification. L'adoption de la culture de l'intégration continue et de la livraison/déploiement continus (CI/CD) en fait partie intégrante. Dans ce cas, un commit ou une modification de code passe par différentes étapes automatisées, depuis les étapes de compilation et de test jusqu'au déploiement sur des environnements de production.

Bien que les principes traditionnels suggèrent d'impliquer l'intervention humaine pour les procédures opérationnelles les plus complexes, nous vous conseillons d'automatiser ces mêmes procédures pour cette même raison.

Étapes d'implémentation

Vous pouvez automatiser les déploiements pour supprimer les opérations manuelles en procédant comme suit :

- Configurez un référentiel de code pour stocker votre code en toute sécurité : utilisez AWS CodeCommit pour créer un référentiel sécurisé basé sur Git.
- Configurez un service d'intégration continue pour compiler votre code source, exécuter des tests et créer des artefacts de déploiement : pour configurer un projet de génération à cette fin, consultez Premiers pas avec AWS CodeBuild à l'aide de la console.
- Configurez un service de déploiement qui automatise les déploiements d'applications et gère la complexité des mises à jour des applications sans recourir aux déploiements manuels sujets aux erreurs : AWS CodeDeploy automatise les déploiements de logiciels vers divers services informatiques, comme Amazon EC2, AWS Fargate, AWS Lambda et vos serveurs sur site. Pour configurer ces étapes, consultez Premiers pas avec CodeDeploy.
- Configurez un service de livraison continue qui automatise vos pipelines de versions pour des mises à jour plus rapides et plus fiables des applications et de l'infrastructure : pensez à utiliser AWS CodePipeline pour vous aider à automatiser vos pipelines de publication. Pour plus d'informations, consultez les tutoriels CodePipeline.

Ressources

Bonnes pratiques associées :

- OPS05-BP04 Utiliser des systèmes de gestion du développement et du déploiement
- OPS05-BP10 Automatiser complètement l'intégration et le déploiement
- OPS06-BP02 Déploiements de tests
- OPS06-BP04 Automatiser les tests et les restaurations

Documents connexes :

- Continuous Delivery of Nested AWS CloudFormation Stacks Using AWS CodePipeline
- Complete CI/CD with AWS CodeCommit, AWS CodeBuild, AWS CodeDeploy, and AWS CodePipeline
- Partenaire APN : partenaires pouvant vous aider à créer des solutions de déploiement automatisées
- AWS Marketplace : produits pouvant être utilisés pour automatiser vos déploiements
- Automatisez les messages de chat avec les webhooks.
- L'Amazon Builders' Library : Garantir la sécurité des restaurations pendant les déploiements
- L'Amazon Builders' Library : Aller plus vite avec la distribution continue
- Qu'est-ce qu'AWS CodePipeline ?
- Qu'est-ce qu'CodeDeploy ?
- AWS Systems Manager Patch Manager
- Qu'est-ce qu'Amazon SES ?
- Qu'est-ce qu'Amazon Simple Notification Service ?

Vidéos connexes :

- AWS Summit 2019: CI/CD on AWS

Gestion des défaillances

> Des pannes finiront toujours par arriver : des routeurs aux disques durs, des systèmes d'exploitation aux unités de mémoire corrompant des paquets TCP, des erreurs transitoires aux pannes permanentes. C'est inéluctable, que vous utilisiez du matériel de la plus haute qualité ou les composants les moins chers - Werner Vogels, directeur technique - Amazon.com

Les pannes des composants matériels de bas niveau doivent être traitées au quotidien dans un centre de données sur site. En revanche, dans le cloud, vous devriez être à l'abri de la plupart de ces types de défaillances. Par exemple, les volumes Amazon EBS sont placés dans une zone de disponibilité spécifique où ils sont automatiquement répliqués pour vous protéger de la panne d'un seul composant. Tous les volumes EBS sont conçus pour fournir une disponibilité de 99,999 %. Les objets Amazon S3 sont stockés dans au moins trois zones de disponibilité offrant une durabilité des objets de 99,999999999 % sur une année donnée. Quel que soit votre fournisseur de cloud, des défaillances peuvent avoir un impact sur votre charge de travail. Vous devez donc prendre des mesures pour mettre en œuvre la résilience si voulez que votre charge de travail soit fiable.

Pour appliquer les bonnes pratiques présentées ici, vous devez vous assurer que les personnes qui conçoivent, implémentent et exécutent vos charges de travail connaissent les objectifs commerciaux et de fiabilité requis pour y parvenir. Elles doivent maîtriser ces exigences de fiabilité et être formées pour y répondre.

Les sections suivantes expliquent les bonnes pratiques de gestion des pannes afin de prévenir tout impact sur votre charge de travail.

Rubriques

- sauvegarder les données ;
- Utilisation de l'isolation des défaillances pour protéger votre charge de travail
- Conception d'une charge de travail qui résiste aux défaillances des composants
- Test de la fiabilité
- Planification de la reprise après sinistre

sauvegarder les données ;

Sauvegardez les données, les applications et la configuration pour répondre aux exigences relatives à la durée maximale d'interruption admissible (RTO) et aux objectifs de point de reprise (RPO).

Bonnes pratiques

- REL09-BP01 Identifier et sauvegarder toutes les données qui doivent être sauvegardées, ou reproduire les données à partir de sources
- REL09-BP02 Sécuriser et chiffrer les sauvegardes
- REL09-BP03 Effectuer automatiquement la sauvegarde des données
- REL09-BP04 Effectuer une récupération périodique des données pour vérifier l'intégrité et les processus de sauvegarde

REL09-BP01 Identifier et sauvegarder toutes les données qui doivent être sauvegardées, ou reproduire les données à partir de sources

Identifiez et utilisez les fonctionnalités de sauvegarde des services et ressources de données utilisés par votre charge de travail. La plupart des services offrent des fonctionnalités permettant de sauvegarder vos données de charge de travail.

Résultat souhaité : les sources de données ont été identifiées et classées en fonction de leur ordre d'importance. Définissez ensuite une stratégie de récupération des données basée sur le RPO. Cette stratégie implique soit de sauvegarder ces sources de données, soit d'avoir la capacité de reproduire des données provenant d'autres sources. En cas de perte de données, la stratégie mise en place permet la récupération ou la reproduction des données dans les RPO et RTO définis.

Phase de maturité du cloud : fondamentale

Anti-modèles courants :

- Ne pas connaître toutes les sources de données pour la charge de travail ni leur ordre d'importance.
- Ne pas effectuer de sauvegardes des sources de données critiques.
- Sauvegarder uniquement certaines sources de données sans utiliser leur ordre d'importance comme critère.
- Aucun RPO défini, ou la fréquence de sauvegarde ne parvient pas à atteindre le RPO.

- Ne pas évaluer si une sauvegarde est nécessaire ou si les données peuvent être reproduites à partir d'autres sources.

Avantages liés au respect de cette bonne pratique : identifier les emplacements où les sauvegardes sont nécessaires et mettre en place un mécanisme pour créer des sauvegardes, ou être capable de reproduire les données à partir d'une source externe améliore la capacité de restauration et de récupération des données lors d'une panne.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

Tous les magasins de données AWS offrent des fonctionnalités de sauvegarde. Des services comme Amazon RDS et Amazon DynamoDB prennent également en charge la sauvegarde automatisée qui permet la récupération ponctuelle (PITR). Vous pouvez ainsi restaurer une sauvegarde remontant jusqu'à cinq minutes ou moins avant l'heure actuelle. De nombreux services AWS offrent la possibilité de copier les sauvegardes vers une autre Région AWS. AWS Backup est un outil qui vous permet de centraliser et d'automatiser la protection des données entre les services AWS. AWS Elastic Disaster Recovery vous permet de copier des charges de travail complètes de serveurs et de maintenir une protection continue des données à partir d'un site, d'une zone géographique ou d'une région, avec un objectif de point de reprise (RPO) mesuré en secondes.

Amazon S3 peut être utilisé comme destination de sauvegarde pour les sources de données autogérées et gérées par AWS. Les services AWS tels qu'Amazon EBS, Amazon RDS et Amazon DynamoDB ont des fonctionnalités intégrées permettant de créer des sauvegardes. Vous pouvez aussi utiliser des logiciels de sauvegarde tiers.

Les données sur site peuvent être sauvegardées sur le AWS Cloud à l'aide de AWS Storage Gateway ou de AWS DataSync. Les compartiments Amazon S3 permettent de stocker ces données sur AWS. Amazon S3 offre plusieurs niveaux de stockage tels que Amazon S3 Glacier ou S3 Glacier Deep Archive pour réduire le coût du stockage des données.

Il se peut que vous puissiez répondre aux besoins de récupération de données en reproduisant les données à partir d'autres sources. Par exemple, les nœuds de réplica Amazon ElastiCache ou les réplicas en lecture Amazon RDS peuvent reproduire des données en cas de perte de la source principale. Dans les cas où des sources de ce type peuvent être utilisées pour atteindre votre objectif de point de reprise (RPO) et votre objectif de délai de reprise (RTO), il se peut que vous n'ayez pas besoin d'une sauvegarde. Autre exemple, si vous travaillez avec Amazon EMR, il n'est peut-être pas

nécessaire de sauvegarder votre magasin de données HDFS, tant que vous pouvez reproduire les données dans Amazon EMR à partir de Amazon S3.

Lors de la sélection d'une stratégie de sauvegarde, tenez compte du temps nécessaire pour récupérer les données. Le temps nécessaire pour récupérer les données dépend du type de sauvegarde (dans le cas d'une stratégie de sauvegarde) ou de la complexité du mécanisme de reproduction des données. Cette durée doit être conforme au RTO de la charge de travail.

Étapes d'implémentation

1. Identifiez toutes les sources de données pour la charge de travail. Les données peuvent être stockées sur un certain nombre de ressources telles que les bases de données, les volumes, les systèmes de fichiers, les systèmes de journalisation et le stockage d'objets. Reportez-vous à la section Ressources pour trouver des documents connexes sur les différents services AWS où les données sont stockées, et la capacité de sauvegarde que ces services fournissent.
2. Classez les sources de données en fonction de leur ordre d'importance. Différents jeux de données ont différents niveaux d'importance pour une charge de travail, et donc différentes exigences en matière de résilience. Par exemple, certaines données peuvent être critiques et nécessiter un RPO proche de zéro, tandis que d'autres données peuvent être moins critiques et peuvent tolérer un RPO plus élevé et la perte de certaines données. De même, différents jeu de données peuvent également avoir des exigences de RTO différentes.
3. Utilisez AWS ou des services tiers pour créer des sauvegardes des données. AWS Backup est un service géré qui permet de créer des sauvegardes de diverses sources de données sur AWS. AWS Elastic Disaster Recovery gère la réplication automatisée des données à la seconde près vers une Région AWS. La plupart des services AWS ont également des fonctionnalités natives permettant de créer des sauvegardes. AWS Marketplace inclut de nombreuses solutions qui offrent également ces fonctionnalités. Reportez-vous à la section Ressources ci-dessous pour découvrir comment créer des sauvegardes de données à partir de divers services AWS.
4. Pour les données non sauvegardées, définissez un mécanisme de reproduction des données. Vous pouvez choisir de ne pas sauvegarder les données qui peuvent être reproduites à partir d'autres sources pour diverses raisons. Il peut arriver qu'il soit moins coûteux de reproduire des données à partir de sources en cas de besoin plutôt que de créer une sauvegarde, car le stockage des sauvegardes peut impliquer un coût. Ou peut-être la restauration à partir d'une sauvegarde prend-elle plus de temps que la reproduction des données à partir des sources, ce qui entraîne une violation du RTO. Dans de telles situations, envisagez les avantages et inconvénients de chaque approche et définissez un processus clair sur la façon dont les données peuvent être reproduites à partir de ces sources lorsque la récupération des données est nécessaire. Si vous

avez chargé des données depuis Amazon S3 vers un entrepôt de données (comme Amazon Redshift) ou un cluster MapReduce (comme Amazon EMR) pour les analyser, vous disposez d'un exemple de données reproductibles à partir d'autres sources. Tant que les résultats de ces analyses sont stockés quelque part ou reproductibles, vous ne perdrez pas données en cas de défaillance de l'entrepôt de données ou du cluster MapReduce. Parmi les autres exemples reproductibles à partir de sources, figurent les caches (comme Amazon ElastiCache) ou les réplicas en lecture RDS.

5. Spécifiez un rythme de sauvegarde des données. La création de sauvegardes de sources de données est un processus périodique, et la fréquence doit dépendre du RPO.

Niveau d'effort du plan d'implémentation : modéré

Ressources

Bonnes pratiques associées :

REL13-BP01 Définir les objectifs de reprise pour les temps d'arrêt et les pertes de données

REL13-BP02 Utiliser des stratégies de reprise définies pour répondre aux objectifs de reprise

Documents connexes :

- Qu'est-ce que AWS Backup ?
- What is AWS DataSync? (Qu'est-ce qu'AWS DataSync ?)
- Qu'est-ce que la passerelle de volume ?
- Partenaire APN : partenaires pouvant faciliter la sauvegarde
- AWS Marketplace : produits pouvant être utilisés pour la sauvegarde
- Instantanés Amazon EBS
- Backing Up Amazon EFS (Sauvegarde Amazon EFS)
- Sauvegarde d'Amazon FSx for Windows File Server
- Sauvegarde et restauration d'ElastiCache pour Redis
- Création d'un instantané de cluster de base de données dans Neptune
- Création d'un instantané de base de données
- Création d'une règle EventBridge qui se déclenche selon un calendrier
- Réplication entre régions avec Amazon S3

- EFS-to-EFS AWS Backup
- Exportation de données de journal vers Amazon S3
- Gestion du cycle de vie des objets
- Sauvegarde et restauration à la demande pour DynamoDB
- Restauration à un instant dans le passé pour DynamoDB
- Utilisation des instantanés d'index Amazon OpenSearch Service
- Qu'est-ce que AWS Elastic Disaster Recovery ?

Vidéos connexes :

- AWS re:Invent 2021 - Backup, disaster recovery, and ransomware protection with AWS
- AWS Backup Demo: Cross-Account and Cross-Region Backup (Démonstration de la sauvegarde AWS : sauvegarde intercompte et inter-régions)
- AWS re:Invent 2019: Deep dive on AWS Backup, ft. Rackspace (STG341)

Exemples connexes :

- Atelier Well-Architected : implémentation de la réplication bidirectionnelle entre régions pour Amazon S3
- Atelier Well-Architected : test de la sauvegarde et de la restauration de données
- Atelier Well-Architected : sauvegarde et restauration avec basculement automatique pour la charge de travail d'analyse
- Atelier Well-Architected : reprise après sinistre - sauvegarde et restauration

REL09-BP02 Sécuriser et chiffrer les sauvegardes

Contrôlez et détectez l'accès aux sauvegardes à l'aide de l'authentification et de l'autorisation. Assurez la prévention et détectez si l'intégrité des données des sauvegardes est compromise à l'aide du chiffrement.

Anti-modèles courants :

- Avoir le même accès aux sauvegardes et à l'automatisation de la restauration que vous le faites pour les données.
- Absence de chiffrement de vos sauvegardes.

Avantages liés au respect de cette bonne pratique : la sécurisation de vos sauvegardes empêche la falsification des données. De même, le chiffrement des données empêche l'accès à ces données si elles sont accidentellement exposées.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

Contrôlez et détectez l'accès aux sauvegardes à l'aide de l'authentification et de l'autorisation, telles qu'AWS Identity and Access Management (IAM). Assurez la prévention et détectez si l'intégrité des données des sauvegardes est compromise à l'aide du chiffrement.

Amazon S3 prend en charge plusieurs méthodes de chiffrement de vos données inactives. Grâce au chiffrement côté serveur, Amazon S3 accepte vos objets sous forme de données non chiffrées, puis les chiffre lors de leur stockage. Avec le chiffrement côté client, votre application de charge de travail s'occupe du chiffrement des données avant leur transmission à Amazon S3. Ces deux méthodes vous permettent d'utiliser AWS Key Management Service (AWS KMS) pour créer et stocker la clé de données. Vous pouvez également fournir votre propre clé (vous en assumez alors la responsabilité). Avec AWS KMS, vous pouvez définir des stratégies via IAM pour déterminer qui peut ou non accéder à vos clés de données et à vos données déchiffrées.

Pour Amazon RDS, si vous avez choisi de chiffrer vos bases de données, vos sauvegardes sont également chiffrées. Les sauvegardes DynamoDB sont toujours chiffrées. En utilisant AWS Elastic Disaster Recovery, toutes les données en transit et au repos sont chiffrées. Avec Elastic Disaster Recovery, les données au repos peuvent être chiffrées à l'aide de la clé Amazon EBS de chiffrement de volume par défaut ou d'une clé personnalisée gérée par le client.

Étapes d'implémentation

1. Utilisez le chiffrement sur chacun de vos magasins de données. La sauvegarde est également chiffrée si vos données sources le sont.
   - Utilisez le chiffrement dans Amazon RDS. Vous pouvez configurer le chiffrement au repos à l'aide d'AWS Key Management Service lorsque vous créez une instance RDS.
   - Utilisez le chiffrement sur les volumes Amazon EBS. Vous pouvez configurer le chiffrement par défaut ou spécifier une clé unique lors de la création du volume.
   - Utilisez le chiffrement Amazon DynamoDB requis. DynamoDB chiffre toutes les données au repos. Vous pouvez utiliser une clé AWS KMS détenue par AWS ou une clé KMS gérée par AWS, en spécifiant une clé stockée dans votre compte.

- Chiffrez vos données stockées dans Amazon EFS. Configurez le chiffrement lorsque vous créez votre système de fichiers.
- Configurez le chiffrement dans les régions source et de destination. Vous pouvez configurer le chiffrement au repos dans Amazon S3 à l'aide de clés stockées dans KMS, mais les clés sont spécifiques à la région. Vous pouvez spécifier les clés de destination lorsque vous configurez la réplication.
- Choisissez d'utiliser le chiffrement par défaut ou le chiffrement Amazon EBS personnalisé pour Elastic Disaster Recovery. Cette option permet de chiffrer les données répliquées au repos sur les disques du sous-réseau de la zone de transit et sur les disques répliqués.

2. Mettez en œuvre les autorisations de moindre privilège pour accéder à vos sauvegardes. Suivez les bonnes pratiques pour limiter l'accès aux sauvegardes, instantanés et réplicas conformément aux bonnes pratiques de sécurité.

Ressources

Documents connexes :

- AWS Marketplace : produits pouvant être utilisés pour la sauvegarde
- Chiffrement Amazon EBS
- Protection des données Amazon S3 à l'aide du chiffrement
- Configuration supplémentaire de la réplication entre régions : réplication d'objets créés avec le chiffrement côté serveur (SSE) à l'aide de clés de chiffrement stockées dans AWS KMS
- Chiffrement DynamoDB au repos
- Chiffrement des ressources Amazon RDS
- Chiffrement des données et métadonnées dans Amazon EFS
- Chiffrement des sauvegardes dans AWS
- Gestion des tables chiffrées
- Pilier Sécurité - CadreAWS Well-Architected
- Qu'est-ce que AWS Elastic Disaster Recovery ?

Exemples connexes :

- Atelier Well-Architected : implémentation de la réplication bidirectionnelle entre régions pour Amazon S3

REL09-BP03 Effectuer automatiquement la sauvegarde des données

Configurez les sauvegardes à effectuer automatiquement en fonction d'un calendrier périodique informé par l'objectif de point de récupération (RPO) ou par les modifications du jeu de données. Les jeux de données critiques dont le seuil de tolérance pour la perte de données est faible doivent être sauvegardés automatiquement et fréquemment, tandis que les données moins critiques où certaines données peuvent être perdues peuvent être sauvegardées moins fréquemment.

Résultat souhaité : un processus automatisé qui crée des sauvegardes de sources de données à une cadence établie.

Anti-modèles courants :

- Exécution manuelle des sauvegardes
- Utilisation de ressources qui ont une capacité de sauvegarde, mais sans inclure la sauvegarde dans votre automatisation.

Avantages liés au respect de cette bonne pratique : l'automatisation des sauvegardes permet de vérifier qu'elles sont effectuées régulièrement en fonction de votre RPO, et vous alerte si elles ne sont pas effectuées.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

AWS Backup peut être utilisé pour créer des sauvegardes de données automatisées de diverses sources de données AWS. Les instances Amazon RDS peuvent être sauvegardées presque en continu toutes les cinq minutes, et les objets Amazon S3 peuvent être sauvegardés presque en continu toutes les quinze minutes, ce qui permet une récupération ponctuelle (PITR) dans l'historique de sauvegarde. Pour les autres sources de données AWS, telles que les volumes Amazon EBS, les tables Amazon DynamoDB ou les systèmes de fichiers Amazon FSx, AWS Backup peut exécuter une sauvegarde automatique qui peut avoir lieu toutes les heures. Ces services offrent également des capacités de sauvegarde natives. Les services AWS qui offrent une sauvegarde automatisée avec une récupération ponctuelle comprennent Amazon DynamoDB , Amazon RDS, et Amazon Keyspaces (pour Apache Cassandra) : ceux-ci peuvent être restaurés à un moment spécifique dans l'historique de sauvegarde. La plupart des autres services de stockage de données AWS offrent la possibilité de planifier des sauvegardes périodiques, à une fréquence de sauvegarde pouvant atteindre toutes les heures.

Amazon RDS et Amazon DynamoDB offrent une sauvegarde continue avec une récupération ponctuelle. La gestion des versions Amazon S3, une fois activée, est automatique.Amazon Data Lifecycle Manager peut automatiser la création, la copie et la suppression d'instantanés Amazon EBS. Il peut également automatiser la création, la copie, l'abandon et le désenregistrement des Amazon Machine Images (AMI) basées sur Amazon EBS et de leurs instantanés Amazon EBS sous-jacents.

AWS Elastic Disaster Recovery fournit une réplication continue au niveau des blocs depuis l'environnement source (sur site ou sur AWS) vers la région de reprise cible. Des instantanés Amazon EBS ponctuels sont automatiquement créés et gérés par le service.

AWS Backup fournit une solution de sauvegarde entièrement gérée basée sur des stratégies afin d'offrir une vue centralisée de l'automatisation et de l'historique de vos sauvegardes. Cette solution centralise et automatise la sauvegarde des données sur plusieurs services AWS dans le cloud et sur site à l'aide d'AWS Storage Gateway.

Outre la gestion des versions, Amazon S3 intègre la réplication. L'intégralité du compartiment S3 peut être automatiquement répliquée vers un autre compartiment d'une autre Région AWS.

Étapes d'implémentation

1. Identifiez les sources de données qui sont actuellement sauvegardées manuellement. Pour en savoir plus, consultez REL09-BP01 Identifier et sauvegarder toutes les données qui doivent être sauvegardées, ou reproduire les données à partir de sources. »
2. Déterminez le RPO de la charge de travail. Pour en savoir plus, consultez REL13-BP01 Définir les objectifs de reprise pour les temps d'arrêt et les pertes de données. »
3. Utilisez une solution de sauvegarde automatisée ou un service géré. AWS Backup est un service entièrement géré qui permet de centraliser et d'automatiser facilement la protection des données entre les services AWS, dans le cloud et sur site. En utilisant les plans de sauvegarde dans AWS Backup, créez des règles qui définissent les ressources à sauvegarder, et la fréquence à laquelle ces sauvegardes doivent être créées. Cette fréquence doit être informée par le RPO établi à l'étape 2. Pour obtenir des conseils pratiques sur la création de sauvegardes automatisées à l'aide de AWS Backup, consultez Test de la sauvegarde et de la restauration de données. Les fonctionnalités de sauvegarde natives sont offertes par la plupart des services AWS qui stockent des données. Par exemple, RDS peut être exploité pour les sauvegardes automatisées avec une récupération ponctuelle (PITR).
4. Pour les sources de données non prises en charge par une solution de sauvegarde automatisée ou un service géré tel que des sources de données sur site ou des files d'attente de messages,

envisagez d'utiliser une solution tierce de confiance pour créer des sauvegardes automatisées. Vous pouvez également créer une automatisation pour ce faire avec AWS CLI ou les kits SDK. Vous pouvez utiliser les fonctions AWS Lambda ou AWS Step Functions pour définir la logique impliquée dans la création d'une sauvegarde de données, et exploiter Amazon EventBridge pour l'exécuter à une fréquence basée sur votre RPO.

Niveau d'effort du plan d'implémentation : faible

Ressources

Documents connexes :

- Partenaire APN : partenaires pouvant faciliter la sauvegarde
- AWS Marketplace : produits pouvant être utilisés pour la sauvegarde
- Création d'une règle EventBridge qui se déclenche selon un calendrier
- Qu'est-ce que AWS Backup ?
- Qu'est-ce que AWS Step Functions ?
- Qu'est-ce que AWS Elastic Disaster Recovery ?

Vidéos connexes :

- AWS re:Invent 2019: Deep dive on AWS Backup, ft. Rackspace (STG341)

Exemples connexes :

- Atelier Well-Architected : test de la sauvegarde et de la restauration de données

REL09-BP04 Effectuer une récupération périodique des données pour vérifier l'intégrité et les processus de sauvegarde

Confirmez que l'implémentation de votre processus de sauvegarde répond à vos objectifs de délai de reprise (RTO) et à vos objectifs de point de reprise (RPO) en effectuant un test de reprise.

Résultat souhaité : les données des sauvegardes sont périodiquement récupérées à l'aide de mécanismes bien définis pour vérifier que la récupération est conforme à l'objectif de temps de récupération (RTO) établi pour la charge de travail. Vérifiez que la restauration à partir d'une

sauvegarde aboutit à une ressource qui contient les données d'origine sans qu'aucune d'entre elles ne soit corrompue ou inaccessible, et avec une perte de données conforme à l'objectif de point de récupération (RPO).

Anti-modèles courants :

- Restauration d'une sauvegarde, mais sans interroger ou récupérer des données pour vérifier l'utilisation de la restauration
- Supposer qu'une sauvegarde existe.
- Supposer que la sauvegarde d'un système est pleinement opérationnelle et que les données peuvent être récupérées à partir de celle-ci.
- Supposer que le temps de restauration ou de récupération des données à partir d'une sauvegarde est conforme au RTO de la charge de travail.
- Supposer que les données contenues dans la sauvegarde sont conformes au RPO de la charge de travail.
- Effectuez une restauration si nécessaire, sans utiliser de runbook ou en dehors d'une procédure automatisée établie.

Avantages liés au respect de cette bonne pratique : le test de la récupération des sauvegardes vérifie que les données peuvent être restaurées en cas de besoin sans craindre qu'elles soient manquantes ou corrompues, que la restauration et la récupération sont possibles conformément au RTO de la charge de travail et que toute perte de données est conforme au RPO de la charge de travail.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

Tester la fonctionnalité de sauvegarde et de restauration permet de garantir que ces actions peuvent être effectuées pendant une panne. Restaurez périodiquement les sauvegardes vers un nouvel emplacement et exécutez des tests pour vérifier l'intégrité des données. Certains tests courants doivent être effectués pour vérifier que toutes les données sont disponibles, qu'elles ne sont pas corrompues, qu'elles sont accessibles et que toute perte de données ne dépasse pas le RPO de la charge de travail. Ces tests peuvent également contribuer à déterminer si les mécanismes de récupération sont suffisamment rapides pour s'adapter au RTO de la charge de travail.

Avec AWS, vous pouvez mettre en place un environnement de test et restaurer vos sauvegardes afin d'évaluer le RTO et le RPO, et exécuter des tests sur le contenu et l'intégrité des données.

De plus, Amazon RDS et Amazon DynamoDB permettent une restauration à un instant donné dans le passé (PITR). Grâce à la sauvegarde continue, vous pouvez restaurer votre jeu de données à l'état dans lequel il était à une date et une heure spécifiées.

si toutes les données sont disponibles, si elles ne sont pas corrompues, si elles sont accessibles et si toute perte de données est conforme au RPO de la charge de travail. Ces tests peuvent également contribuer à déterminer si les mécanismes de récupération sont suffisamment rapides pour s'adapter au RTO de la charge de travail.

AWS Elastic Disaster Recovery offre des instantanés de récupération ponctuelle et continue des volumes Amazon EBS. Au fur et à mesure que les serveurs sources sont répliqués, les états ponctuels sont consignés dans le temps en fonction de la stratégie configurée. Elastic Disaster Recovery vous aide à vérifier l'intégrité de ces instantanés en lançant des instances à des fins de test et d'exercice sans rediriger le trafic.

Étapes d'implémentation

1. Identifiez les sources de données qui sont actuellement sauvegardées et où ces sauvegardes sont stockées. Pour obtenir des conseils de mise en œuvre, consultez REL09-BP01 Identifier et sauvegarder toutes les données qui doivent être sauvegardées, ou reproduire les données à partir de sources. »
2. Établissez des critères de validation des données pour chaque source de données. Différents types de données ont des propriétés différentes qui pourraient nécessiter des mécanismes de validation distincts. Réfléchissez à la manière dont ces données pourraient être validées avant de vous assurer que vous pouvez les utiliser en production. Certaines méthodes courantes de validation des données consistent à utiliser des propriétés de données et de sauvegarde telles que le type de données, le format, la somme de contrôle, la taille ou une combinaison de ces propriétés avec une logique de validation personnalisée. Par exemple, il peut s'agir d'une comparaison des valeurs de somme de contrôle entre la ressource restaurée et la source de données au moment de la création de la sauvegarde.
3. Établissez le RTO et le RPO pour la restauration des données en fonction de leur criticité. Pour obtenir des conseils de mise en œuvre, consultez REL13-BP01 Définir les objectifs de reprise pour les temps d'arrêt et les pertes de données. »
4. Évaluez votre capacité de récupération. Passez en revue votre stratégie de sauvegarde et de restauration pour déterminer si elle peut répondre au RTO et au RPO, et ajustez la stratégie si nécessaire. En utilisant AWS Resilience Hub, vous pouvez effectuer une évaluation de votre charge de travail. Celle-ci se concentre sur la configuration de votre application par rapport à la stratégie de résilience et signale si vos objectifs RTO et RPO peuvent être atteints.

5. Effectuez un test de restauration avec les processus établis utilisés en production pour la restauration des données. Ces processus dépendent de la façon dont la source de données d'origine a été sauvegardée, du format et de l'emplacement de stockage de la sauvegarde elle-même, ou ils varient selon que les données sont reproduites à partir d'autres sources. Par exemple, si vous utilisez un service géré tel que AWS Backup, il peut suffire de restaurer la sauvegarde en tant que nouvelle ressource. Si vous avez utilisé AWS Elastic Disaster Recovery, vous pouvez lancer une opération de récupération.
6. Validez la récupération des données à partir de la ressource restaurée en fonction des critères que vous avez précédemment établis pour la validation des données. Les données restaurées et récupérées contiennent-elles l'enregistrement/l'élément le plus récent au moment de la sauvegarde ? Ces données sont-elles conformes au RPO de la charge de travail ?
7. Mesurez le temps nécessaire à la restauration et à la récupération et comparez-le au RTO que vous avez défini. Ce processus est-il conforme au RTO de la charge de travail ? Par exemple, comparez les horodatages du début du processus de restauration et de la fin de la validation de la récupération pour calculer la durée de ce processus. Tous les appels d'API AWS sont horodatés, et ces informations sont disponibles dans AWS CloudTrail. Bien que ces informations puissent fournir des détails sur le début du processus de restauration, l'horodatage indiquant la fin de la validation doit être enregistré par votre logique de validation. Si vous utilisez un processus automatisé, des services tels que Amazon DynamoDB permettent de stocker ces informations. En outre, de nombreux services AWS fournissent un historique des événements qui contient des informations horodatées indiquant quand certaines actions se sont produites. Au sein de AWS Backup, les actions de sauvegarde et de restauration sont appelées tâches, et ces tâches contiennent des informations d'horodatage dans le cadre de leurs métadonnées qui peuvent mesurer le temps nécessaire à la restauration et à la récupération.
8. Informez les parties prenantes si la validation des données échoue ou si le temps requis pour la restauration et la récupération dépasse le RTO établi pour la charge de travail. Lors de la mise en œuvre de l'automatisation, comme dans cet atelier, des services tels que Amazon Simple Notification Service (Amazon SNS) peuvent envoyer des notifications push, par e-mail ou SMS, aux parties prenantes. Ces messages peuvent également être publiés dans des applications de messagerie telles que Amazon Chime, Slack ou Microsoft Teams ou utilisés pour créer des tâches en tant qu'OpsItems à l'aide d'AWS Systems Manager OpsCenter.
9. Automatisez ce processus pour qu'il s'exécute périodiquement. Par exemple, des services comme AWS Lambda ou une machine d'état dans AWS Step Functions peuvent être utilisés pour automatiser les processus de restauration et de récupération, et Amazon EventBridge peut être utilisé pour déclencher périodiquement ce flux de travail d'automatisation, comme indiqué dans le diagramme d'architecture ci-dessous. Apprenez à automatiser la validation de la récupération

des données avec AWS Backup. En outre, cet atelier Well-Architected permet d'acquérir une expérience pratique sur la façon d'automatiser plusieurs des étapes décrites ici.

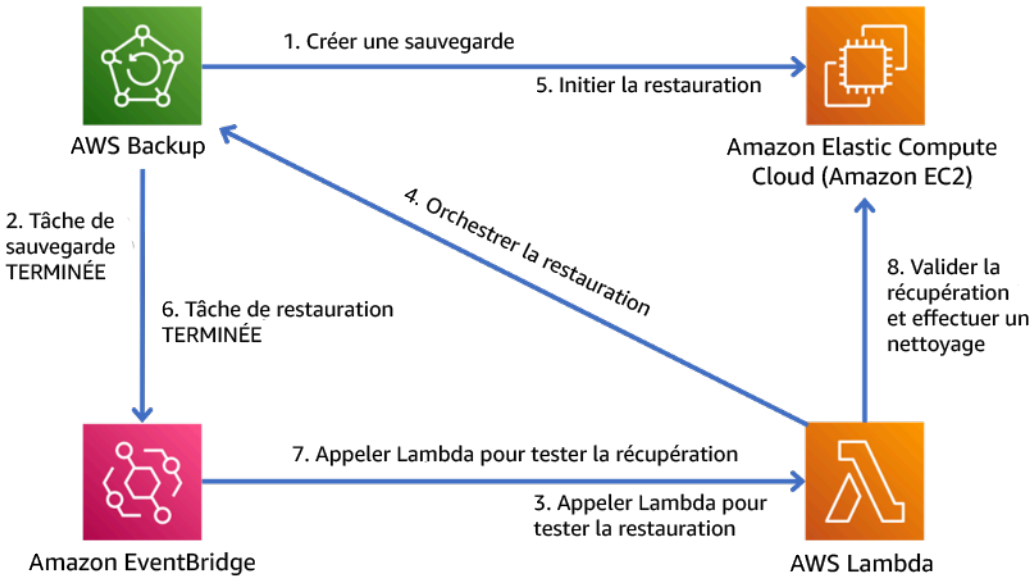


Figure 9 : Processus de sauvegarde et de restauration automatisé

Niveau d'effort du plan d'implémentation : modéré à élevé selon la complexité des critères de validation.

Ressources

Documents connexes :

- Automatiser la validation de la récupération des données avec AWS Backup
- Partenaire APN : partenaires pouvant faciliter la sauvegarde
- AWS Marketplace : produits pouvant être utilisés pour la sauvegarde
- Création d'une règle EventBridge qui se déclenche selon un calendrier
- Sauvegarde et restauration à la demande pour DynamoDB
- Qu'est-ce que AWS Backup ?
- Qu'est-ce que AWS Step Functions ?
- Qu'est-ce que AWS Elastic Disaster Recovery
- AWS Elastic Disaster Recovery

Exemples connexes :

- Atelier Well-Architected : test de la sauvegarde et de la restauration de données

Utilisation de l'isolation des défaillances pour protéger votre charge de travail

Les limites isolées pour les défaillances limitent l'effet d'une défaillance au sein d'une charge de travail à un nombre limité de composants. Les composants en dehors de la limite ne sont pas affectés par la défaillance. En utilisant plusieurs limites isolées par défaut, vous pouvez limiter l'impact sur votre charge de travail.

Bonnes pratiques

- REL10-BP01 Déployer la charge de travail sur plusieurs emplacements
- REL10-BP02 Sélectionner les emplacements appropriés pour votre déploiement multisite
- REL10-BP03 Automatiser la récupération pour les composants limités à un seul emplacement
- REL10-BP04 Utiliser des architectures cloisonnées pour limiter la portée de l'impact

REL10-BP01 Déployer la charge de travail sur plusieurs emplacements

Distribuez les données et les ressources de charge de travail sur plusieurs zones de disponibilité ou, si nécessaire, entre Régions AWS. Ces emplacements peuvent être aussi variés que nécessaire.

L'un des principes fondamentaux de la conception de services dans AWS est d'éviter les points de défaillance uniques dans l'infrastructure physique sous-jacente. Cela nous motive à développer des logiciels et des systèmes qui utilisent plusieurs zones de disponibilité et sont résilients à la défaillance d'une seule zone. De la même manière, les systèmes sont conçus pour être résilients à la défaillance d'un seul nœud de calcul, d'un seul volume de stockage ou d'une seule instance de base de données. Lors de la création d'un système qui s'appuie sur des composants redondants, il est important de s'assurer que les composants fonctionnent de manière indépendante et, dans le cas de Régions AWS, de façon autonome. Les avantages obtenus grâce aux calculs de disponibilité théoriques avec des composants redondants ne sont valides qu'à cette condition.

Zones de disponibilité (AZ)

Les Régions AWS sont composées de plusieurs zones de disponibilité, toutes conçues pour être indépendantes les unes des autres. Chaque zone de disponibilité est séparée par une distance physique logique des autres zones afin d'éviter les scénarios de défaillance corrélées liés à des

risques environnementaux comme les incendies, les inondations et les tornades. Chaque zone de disponibilité comporte également une infrastructure physique indépendante : connexions dédiées à l'alimentation, sources d'énergie d'appoint indépendantes, services mécaniques indépendants et connectivité réseau indépendante dans et au-delà de la zone de disponibilité. Ce modèle limite les défaillances susceptibles de se produire dans l'un de ces systèmes à la seule zone de disponibilité affectée. Bien que géographiquement séparées, les zones de disponibilité sont situées dans la même zone régionale, ce qui permet une mise en réseau à haut débit et à faible latence. L'intégralité de la Région AWS (dans toutes les zones de disponibilité, composées de plusieurs centres de données physiquement indépendants) peut être traitée comme une cible de déploiement logique unique pour votre charge de travail, y compris pour répliquer les données de manière synchrone (par exemple, entre les bases de données). Cela vous permet d'utiliser les zones de disponibilité dans une configuration active/active ou active/veille.

Les zones de disponibilité sont indépendantes et, par conséquent, la disponibilité de la charge de travail est augmentée lorsque la charge de travail est conçue pour utiliser plusieurs zones. Certains services AWS (y compris le plan de données d'instance Amazon EC2) sont déployés en tant que services strictement locaux où leur sort dépend de la zone de disponibilité dans laquelle ils se trouvent. Cependant, les instances Amazon EC2 dans les autres AZ ne sont pas affectées et continuent à fonctionner . De même, si une défaillance dans une zone de disponibilité entraîne l'échec d'une base de données Amazon Aurora, une instance de réplica en lecture Aurora dans une AZ non affectée peut être automatiquement promue comme instance principale. D'autre part, les services régionaux AWS, comme Amazon DynamoDB, utilisent en interne plusieurs zones de disponibilité dans une configuration active/active pour atteindre les objectifs de conception de disponibilité de ce service, sans que vous ayez besoin de configurer le placement AZ.

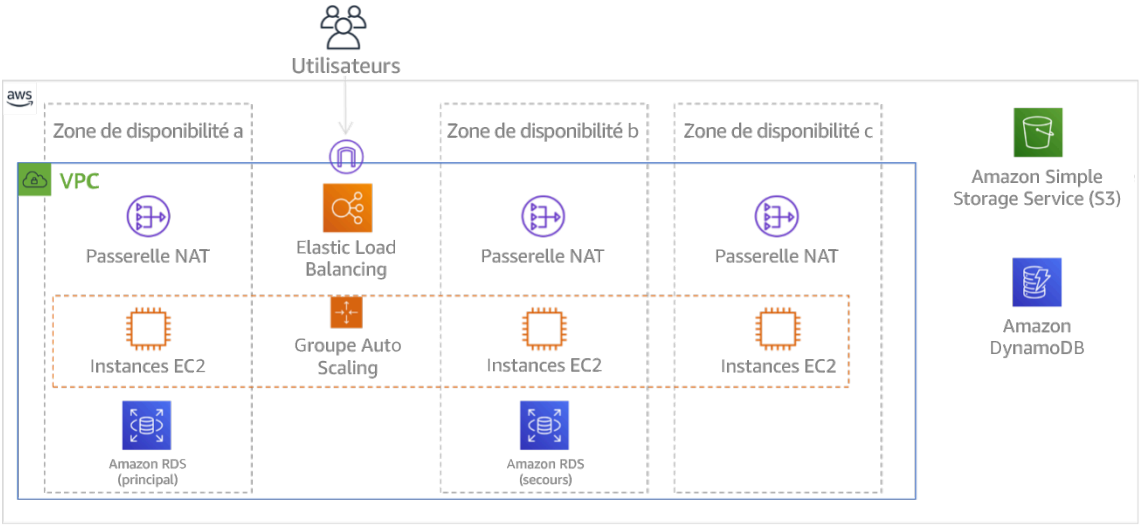


Figure 9 : Architecture multiniveau déployée sur trois zones de disponibilité. Notez qu'Amazon S3 et Amazon DynamoDB comportent toujours automatiquement plusieurs zones de disponibilité. L'ELB est également déployé dans les trois zones.

Bien que les plans de contrôle AWSoffrent généralement la possibilité de gérer les ressources sur l'ensemble de la région (plusieurs zones de disponibilité), certains plans de contrôle (y compris Amazon EC2 et Amazon EBS) ont la capacité de filtrer les résultats en une seule zone de disponibilité. Lorsque c'est le cas, la requête est traitée uniquement dans la zone de disponibilité spécifiée, ce qui réduit l'exposition aux perturbations dans les autres zones de disponibilité. Cet exemple AWS CLI illustre l'obtention d'informations sur l'instance Amazon EC2 uniquement à partir de la zone de disponibilité us-east-2c :

```
AWS ec2 describe-instances --filters Name=availability-zone,Values=us-east-2c
```

AWS Local Zones

Les AWS Local Zones agissent de la même manière que les zones de disponibilité au sein de leur Région AWS respective, car elles sont sélectionnables en tant qu'emplacement de placement des ressources AWS de la zone comme les sous-réseaux et les instances EC2. Elles sont spéciales, car elles ne sont pas situées dans la Région AWS associée, mais près de grands centres de population, industriels et informatiques où aucune Région AWS n'existe actuellement. Cependant, elles conservent une connexion sécurisée et à bande passante élevée entre les charges de travail locales dans la zone locale et celles exécutées dans la Région AWS. Utilise les AWS Local Zones pour déployer des charges de travail plus près de vos utilisateurs afin de répondre aux exigences de faible latence.

Réseau périphérique mondial d'Amazon

Le réseau périphérique mondial d'Amazon se compose d'emplacements périphériques dans des villes du monde entier. Amazon CloudFront utilise ce réseau pour fournir du contenu aux utilisateurs finaux avec une latence plus faible. AWS Global Accelerator vous permet de créer vos points de terminaison de charge de travail dans ces emplacements périphériques pour assurer l'intégration au réseau mondial AWS à proximité de vos utilisateurs. Amazon API Gateway active les points de terminaison d'API optimisés en périphérie à l'aide d'une distribution CloudFront pour faciliter l'accès client via l'emplacement périphérique le plus proche.

Régions AWS

Les Régions AWS sont conçues pour être autonomes. Par conséquent, pour utiliser une approche multirégion, vous devez déployer des copies dédiées des services dans chaque région.

Une approche multirégion est courante pour que les stratégies de reprise après sinistre atteignent les objectifs de récupération lorsque des événements ponctuels à grande échelle se produisent. Consulter Planification de la reprise après sinistre pour en savoir plus sur ces stratégies. Ici cependant, nous nous concentrons plutôt sur la disponibilité, qui vise à atteindre un objectif de disponibilité moyenne dans le temps. Pour des objectifs de haute disponibilité, une architecture multirégion est généralement conçue pour être active/active, où chaque copie de service (dans ses régions respectives) est active (en répondant aux requêtes).

> **Recommandations**
>
> Les objectifs de disponibilité pour la plupart des charges de travail peuvent être satisfaits à l'aide d'une stratégie multi-AZ dans une seule Région AWS. Envisagez les architectures multirégion uniquement lorsque les charges de travail ont des exigences de disponibilité extrêmes ou en cas d'autres objectifs commerciaux nécessitant une architecture multirégion.

AWS vous offre la possibilité de gérer des services entre régions. Par exemple, AWS fournit une réplication continue et asynchrone des données via la réplication Amazon Simple Storage Service (Amazon S3), les réplicas en lecture Amazon RDS (y compris les réplicas en lecture Aurora) et les tables globales Amazon DynamoDB. Grâce à la réplication continue, des versions de vos données sont disponibles pour une utilisation quasi immédiate dans chacune de vos régions actives.

Avec AWS CloudFormation, vous pouvez définir votre infrastructure et la déployer de manière cohérente sur les Comptes AWS et dans les Régions AWS. AWS CloudFormation StackSets étend cette fonctionnalité en vous permettant de créer, mettre à jour ou supprimer des piles AWS CloudFormation sur plusieurs comptes et régions en une seule opération. Pour les déploiements d'instance Amazon EC2, une AMI (Amazon Machine Image) est utilisée pour fournir des informations telles que la configuration matérielle et les logiciels installés. Vous pouvez implémenter un pipeline Amazon EC2 Image Builder qui crée les AMI dont vous avez besoin et les copie dans vos régions actives. Cela garantit que ces AMI approuvées disposent de tout ce dont vous avez besoin pour déployer et faire évoluer votre charge de travail dans chaque nouvelle région.

Pour acheminer le trafic, Amazon Route 53 et AWS Global Accelerator permettent la définition de politiques qui déterminent quels utilisateurs accèdent à quel point de terminaison régional actif. Avec Global Accelerator, vous définissez une option de trafic pour contrôler le pourcentage de trafic

dirigé vers chaque point de terminaison d'application. Route 53 prend en charge cette approche de pourcentage, ainsi que plusieurs autres politiques disponibles, notamment celles basées sur la géoproximité et la latence. Global Accelerator exploite automatiquement le vaste réseau de serveurs périphériques AWS pour intégrer le trafic à la dorsale du réseau AWS dès que possible, ce qui réduit la latence des demandes.

L'ensemble de ces fonctionnalités opèrent de manière à préserver l'autonomie de chaque région. Il existe quelques rares exceptions à cette approche, y compris nos services qui fonctionnent en périphérie au niveau mondial (comme Amazon CloudFront et Amazon Route 53), ainsi que le plan de contrôle pour le service AWS Identity and Access Management (IAM) La plupart des services fonctionnent entièrement au sein d'une seule région.

Centre de données sur site

Pour les charges de travail exécutées dans un centre de données sur site, concevez une expérience hybride dans la mesure du possible. AWS Direct Connect fournit une connexion réseau dédiée entre vos locaux et AWS, ce qui vous permet d'utiliser les deux.

Une autre option consiste à exécuter l'infrastructure et les services AWS sur site à l'aide d' AWS Outposts. AWS Outposts est un service entièrement géré qui étend l'infrastructure AWS, les services AWS, les API et les outils à votre centre de données. La même infrastructure matérielle utilisée dans le AWS Cloud est installée dans votre centre de données. Les services AWS Outposts sont alors connectés à la Région AWS la plus proche. Vous pouvez ensuite utiliser les services AWS Outposts pour prendre en charge vos charges de travail à faible latence ou vos exigences en matière de traitement des données locales.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

- Utilisez plusieurs zones de disponibilité et Régions AWS. Distribuez les données et les ressources de charge de travail sur plusieurs zones de disponibilité ou, si nécessaire, entre Régions AWS. Ces emplacements peuvent être aussi variés que nécessaire.
  - Les services régionaux sont, par nature, déployés entre les zones de disponibilité.
    - Cela inclut Amazon S3, Amazon DynamoDB et AWS Lambda (lorsqu'ils ne sont pas connectés à un VPC).
  - Déployez votre conteneur, votre instance et vos charges de travail basées sur des fonctions dans plusieurs zones de disponibilité. Utilisez les magasins de données multi-AZ, y compris les caches. Utilisez les fonctionnalités d'EC2 Auto Scaling, le placement de tâches ECS,

la configuration de fonctions AWS Lambda lors de l'exécution de votre VPC et les clusters ElastiCache.

- Utilisez les sous-réseaux situés dans des zones de disponibilité distinctes lorsque vous déployez des groupes Auto Scaling.
  - Exemple : Distribution d'instances dans des zones de disponibilité
  - Stratégies de placement des tâches Amazon ECS
  - Configuration d'une fonction AWS Lambda pour accéder aux ressources dans un Amazon VPC
  - Choix des régions et des zones de disponibilité
- Utilisez des sous-réseaux situés dans des zones de disponibilité distinctes lorsque vous déployez des groupes Auto Scaling.
  - Exemple : Distribution d'instances dans des zones de disponibilité
- Utilisez les paramètres de placement des tâches ECS, en spécifiant des groupes de sous-réseaux de base de données.
  - Stratégies de placement des tâches Amazon ECS
- Utilisez des sous-réseaux dans plusieurs zones de disponibilité lorsque vous configurez une fonction à exécuter dans votre VPC.
  - Configuration d'une fonction AWS Lambda pour accéder aux ressources dans un Amazon VPC
- Utilisez plusieurs zones de disponibilité avec des clusters ElastiCache.
  - Choix des régions et des zones de disponibilité

- Choisissez une stratégie sur plusieurs régions si votre charge de travail doit être déployée dans plusieurs régions. La plupart des besoins de fiabilité peuvent être satisfaits au sein d'une même Région AWS à l'aide d'une stratégie à plusieurs zones de disponibilité. Utilisez une stratégie sur plusieurs régions si nécessaire pour répondre aux besoins de votre entreprise.
  - AWS re:Invent 2018: Architecture Patterns for Multi-Region Active-Active Applications (ARC209-R2)
    - La sauvegarde dans une autre Région AWS peut donner des garanties supplémentaires que les données seront disponibles en cas de besoin.
    - Il existe, pour certaines charges de travail, des exigences réglementaires qui imposent l'utilisation d'une stratégie sur plusieurs régions.
- Évaluez AWS Outposts pour votre charge de travail. Si votre charge de travail nécessite une faible latence de connexion à votre centre de données sur site ou si elle a des exigences locales en

matière de traitement des données, Exécutez ensuite l'infrastructure et les services AWS sur site à l'aide d'AWS Outposts.

  - Qu'est-ce qu'AWS Outposts ?

- Déterminez si AWS Local Zones vous permet de fournir un service à vos utilisateurs. Si vous avez des exigences de faible latence, vérifiez si AWS Local Zones est situé près de vos utilisateurs. Si oui, utilisez-le pour déployer des charges de travail plus près de ces utilisateurs.
  - FAQ sur AWS Local Zones

Ressources

Documents connexes :

- Infrastructure mondiale AWS
- FAQ sur AWS Local Zones
- Stratégies de placement des tâches Amazon ECS
- Choix des régions et des zones de disponibilité
- Exemple : Distribution d'instances dans des zones de disponibilité
- Tables globales : réplication multirégions avec DynamoDB
- Utilisation des bases de données mondiales Amazon Aurora
- Série de blog sur la création d'une application multirégion avec les services AWS
- Qu'est-ce qu'AWS Outposts ?

Vidéos connexes :

- AWS re:Invent 2018: Architecture Patterns for Multi-Region Active-Active Applications (ARC209-R2)
- AWS re:Invent 2019: Innovation and operation of the AWS global network infrastructure (NET339)

REL10-BP02 Sélectionner les emplacements appropriés pour votre déploiement multisite

Résultat souhaité

Pour une haute disponibilité, déployez toujours (si possible) vos composants de charge de travail sur plusieurs zones de disponibilité (AZ), comme illustré à la figure 10. Pour les charges de travail avec des exigences de résilience extrêmes, évaluez soigneusement les options pour une architecture multirégion.

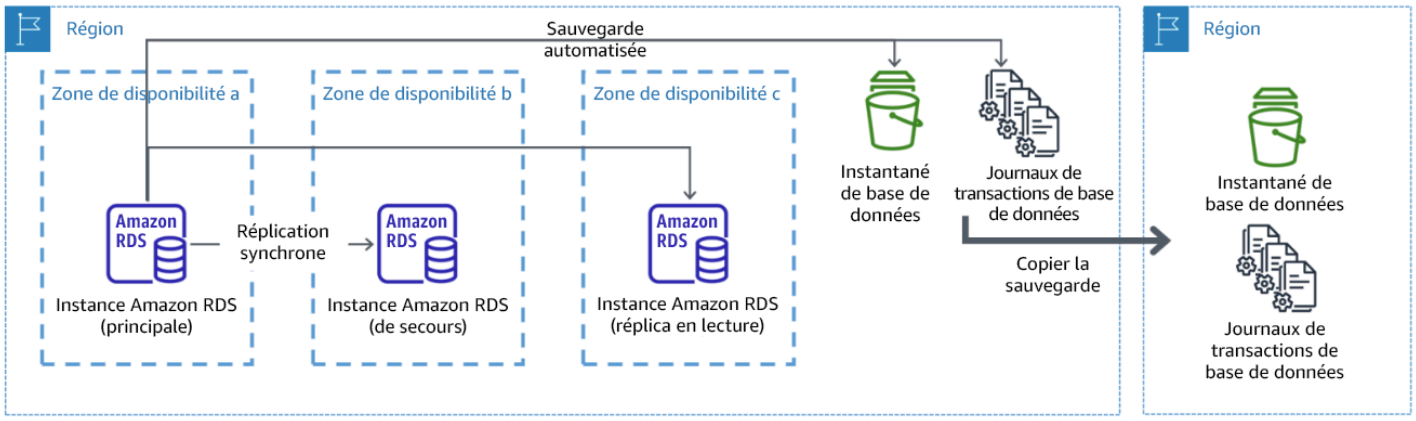


Figure 10 : déploiement d'une base de données multi-AZ résiliente avec sauvegarde dans une autre région AWS

Anti-modèles courants

- Choisir de concevoir une architecture multirégion alors qu'une architecture multi-AZ répond aux exigences.
- Ne pas tenir compte des dépendances entre les composants de l'application si les exigences en matière de résilience et d'emplacements multiples diffèrent entre ces composants.

Avantages liés au respect de cette bonne pratique :

Pour la résilience, vous devez adopter une approche qui repose sur des couches de défense. Une couche protège contre les perturbations de petite envergure et courantes en créant une architecture hautement disponible à l'aide de plusieurs AZ. Une autre couche de défense est destinée à protéger contre les événements rares tels que les catastrophes naturelles généralisées et les perturbations au niveau régional. Cette deuxième couche implique de concevoir l'architecture de votre application pour qu'elle s'étende sur plusieurs Régions AWS.

- La différence entre une disponibilité de 99,5 % et une disponibilité de 99,99 % est de plus de 3,5 heures par mois. La disponibilité attendue d'une charge de travail ne peut atteindre les « quatre neuf » que si elle se trouve dans plusieurs zones de disponibilité.
- En exécutant votre charge de travail dans plusieurs AZ, vous pouvez isoler les pannes d'alimentation, de refroidissement et de mise en réseau, ainsi que la plupart des catastrophes naturelles telles que les incendies et les inondations.
- La mise en œuvre d'une stratégie multirégion pour votre charge de travail permet de la protéger contre les catastrophes naturelles généralisées qui affectent une grande région géographique d'un pays, ou les défaillances techniques à l'échelle régionale. Sachez que la mise en œuvre d'une architecture multirégion peut être très complexe et n'est généralement pas requise pour la plupart des charges de travail.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

Dans le cas d'une catastrophe basée sur l'interruption ou la perte partielle d'une zone de disponibilité, la mise en œuvre d'une charge de travail hautement disponible dans plusieurs zones de disponibilité au sein d'une seule Région AWS permet d'atténuer les effets des catastrophes naturelles et techniques. Chaque Région AWS est composé de plusieurs zones de disponibilité, chacune isolée des défaillances des autres zones et séparée par une distance significative. Cependant, dans le cas d'une catastrophe incluant le risque de perdre plusieurs composants de la zone de disponibilité (composants qui sont à une distance importante les uns des autres), vous devez implémenter des options de reprise après sinistre pour vous prémunir contre les défaillances à l'échelle de la région. Pour les charges de travail nécessitant une résilience extrême (infrastructure critique, applications liées à la santé, infrastructure du système financier, etc.), une stratégie multirégion peut être nécessaire.

Étapes d'implémentation

1. Évaluez votre charge de travail et déterminez si les besoins de résilience peuvent être satisfaits par une approche multi-AZ (Région AWS unique) ou s'ils nécessitent une approche multirégion. La mise en œuvre d'une architecture multirégion pour répondre à ces exigences ajoute de la complexité. Par conséquent, examinez attentivement votre cas d'utilisation et ses exigences. Les exigences de résilience peuvent presque toujours être satisfaites avec une seule Région AWS. Tenez compte des exigences possibles suivantes lorsque vous déterminez si vous devez utiliser plusieurs régions :

a. Reprise après sinistre : dans le cas d'une catastrophe basée sur l'interruption ou la perte partielle d'une zone de disponibilité, la mise en œuvre d'une charge de travail hautement disponible dans plusieurs zones de disponibilité au sein d'une seule Région AWS permet d'atténuer les effets des catastrophes naturelles et techniques. Dans le cas d'une catastrophe incluant le risque de perdre plusieurs composants de la zone de disponibilité (composants qui sont à une distance importante les uns des autres), vous devez implémenter des options de reprise après sinistre entre plusieurs régions pour vous prémunir contre les catastrophes naturelles et les incidents techniques à l'échelle de la région.

b. Haute disponibilité : une architecture multirégion (utilisant plusieurs AZ dans chaque région) peut être utilisée pour atteindre une disponibilité supérieure à quatre 9 (> 99,99 %).

c. Localisation des piles : lors du déploiement d'une charge de travail auprès d'une audience mondiale, vous pouvez déployer des piles localisées dans différentes Régions AWS pour répondre aux besoins des audiences dans ces régions. La localisation peut inclure la langue, la devise et les types de données stockées.

d. Proximité avec les utilisateurs : lors du déploiement d'une charge de travail auprès d'une audience mondiale, vous pouvez réduire la latence en déployant des piles dans les Régions AWS à proximité de l'endroit où se trouvent les utilisateurs finaux.

e. Résidence des données : certaines charges de travail sont soumises à des exigences en matière de situation géographique des données, où les données de certains utilisateurs doivent rester à l'intérieur des frontières d'un pays spécifique. En fonction de la réglementation en question, vous pouvez choisir de déployer une pile entière, ou uniquement les données, dans la Région AWS au sein de ces frontières.

2. Voici quelques exemples de fonctionnalités multi-AZ fournies par les services AWS :

   a. Pour protéger les charges de travail à l'aide d'EC2 ou d'ECS, déployez un Elastic Load Balancer devant les ressources de calcul. Elastic Load Balancing fournira ensuite la solution pour détecter les instances dans les zones non saines et acheminer le trafic vers les zones qui le sont.

      i. Démarrer avec Application Load Balancers

      ii. Démarrer avec les Network Load Balancers

   b. Dans le cas d'instances EC2 exécutant des logiciels commerciaux prêts à l'emploi qui ne prennent pas en charge l'équilibrage de charge, vous pouvez bénéficier d'une forme de tolérance aux pannes via la mise en œuvre d'une méthodologie de reprise après sinistre multi-AZ.

   i. the section called “REL13-BP02 Utiliser des stratégies de reprise définies pour répondre aux objectifs de reprise”

c. Pour les tâches Amazon ECS, déployez votre service uniformément sur trois AZ pour atteindre un juste équilibre entre disponibilité et coût.

   i. Bonnes pratiques de disponibilité Amazon ECS | Conteneurs

d. Pour les tâches qui n'entrent pas dans le cadre d'Aurora Amazon RDS, vous pouvez choisir Multi-AZ comme option de configuration. En cas de défaillance de l'instance de base de données principale, Amazon RDS promeut automatiquement une base de données de secours pour recevoir le trafic dans une autre zone de disponibilité. Des réplicas en lecture multirégion peuvent également être créés pour améliorer la résilience.

   i. Déploiements multi-AZ Amazon RDS

   ii. Création d'un réplica en lecture dans une autre Région AWS

3. Voici quelques exemples de fonctionnalités multirégions fournies par les services AWS :

a. Pour les charges de travail Amazon S3, où la disponibilité multi-AZ est fournie automatiquement par le service, envisagez des points d'accès multirégions si un déploiement multirégion est nécessaire.

   i. Points d'accès multirégions dans Amazon S3

b. Pour les tables DynamoDB, où la disponibilité multi-AZ est fournie automatiquement par le service, vous pouvez facilement convertir les tables existantes en tables globales pour tirer parti de plusieurs régions.

   i. Convertir vos tables Amazon DynamoDB à région unique en tables globales

c. Si votre charge de travail repose sur des Application Load Balancers ou des Network Load Balancers, utilisez AWS Global Accelerator pour améliorer la disponibilité de votre application en dirigeant le trafic vers plusieurs régions qui contiennent des points de terminaison sains.

   i. Points de terminaison pour les accélérateurs standards dans AWS Global Accelerator - AWS Global Accelerator (amazon.com)

d. Pour les applications qui tirent parti d'AWS EventBridge, envisagez des bus interrégionaux pour transférer les événements vers d'autres régions que vous sélectionnez.

   i. Envoi et réception d'événements Amazon EventBridge entre les Régions AWS

e. Pour les bases de données Amazon Aurora, considérez les bases de données globales Aurora, qui couvrent plusieurs régions AWS. Les clusters existants peuvent également être modifiés pour ajouter de nouvelles régions.

   i. Premiers pas avec les avec les bases de données globales Amazon Aurora

f. Si votre charge de travail comprend des clés de chiffrement AWS Key Management Service (AWS KMS), déterminez si les clés multirégions sont appropriées pour votre application.
   i. Clés multirégions dans AWS KMS
g. Pour d'autres fonctionnalités de service AWS, consultez cette série de blog sur la création d'une application multirégion avec les services AWS

Niveau d'effort du plan d'implémentation : De Modéré à Élevé

Ressources

Documents connexes :

- Création d'une application multirégion avec les services AWS
- Architecture de reprise après sinistre sur AWS, partie 4 : multisite actif/actif
- Infrastructure mondiale AWS
- FAQ sur AWS Local Zones
- Architecture de reprise après sinistre (DR) sur AWS, première partie : stratégies de reprise dans le cloud
- La reprise après sinistre est différente dans le cloud
- Tables globales : réplication multirégions avec DynamoDB

Vidéos connexes :

- AWS re:Invent 2018: Architecture Patterns for Multi-Region Active-Active Applications (ARC209-R2)
- Auth0 : architecture à haute disponibilité sur plusieurs régions pouvant atteindre plus de 1,5 milliard de connexions par mois avec basculement automatique

Exemples connexes :

- Architecture de reprise après sinistre (DR) sur AWS, première partie : stratégies de reprise dans le cloud
- DTCC atteint une résilience bien supérieure à ce qu'elle peut obtenir sur site
- Expedia Group utilise une architecture reposant sur plusieurs zones de disponibilité et plusieurs régions avec un service DNS propriétaire pour renforcer la résilience des applications

- Uber : reprise après sinistre pour une plateforme Kafka sur plusieurs régions
- Netflix : stratégie active-active pour une résilience sur plusieurs régions
- Comment nous créons la résidence des données pour le cloud Atlassian
- Intuit TurboTax fonctionne sur deux régions

REL10-BP03 Automatiser la récupération pour les composants limités à un seul emplacement

Si les composants de la charge de travail ne peuvent s'exécuter que dans une seule zone de disponibilité ou un centre de données sur site, implémentez la capacité permettant d'effectuer une reconstruction complète de la charge de travail dans le cadre de vos objectifs de reprise définis.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen

Directives d'implémentation

Si la bonne pratique de déploiement de la charge de travail sur plusieurs emplacements n'est pas possible en raison de contraintes technologiques, vous devez implémenter une autre solution de résilience. Vous devez automatiser la possibilité de recréer l'infrastructure nécessaire, de redéployer les applications et de recréer les données nécessaires pour ces situations.

Par exemple, Amazon EMR lance tous les nœuds d'un cluster donné dans la même zone de disponibilité, car l'exécution d'un cluster dans la même zone améliore les performances des flux de travail en fournissant un taux d'accès aux données plus élevé. Si ce composant est requis pour la résilience de la charge de travail, vous devez pouvoir redéployer le cluster et ses données. De même, pour Amazon EMR, vous devez assurer la redondance autrement qu'en utilisant plusieurs zones de disponibilité. Vous pouvez provisionner plusieurs nœuds. Avec le système de fichiers EMR (EMRFS), les données EMR peuvent être conservées dans Amazon S3, et ainsi être répliquées sur plusieurs zones de disponibilité ou Régions AWS.

De même, Amazon Redshift met en service, par défaut, votre cluster dans une zone de disponibilité sélectionnée de façon aléatoire au sein de la Région AWS que vous sélectionnez. Tous les nœuds de cluster sont mis en service dans la même zone.

Pour les charges de travail basées sur des serveurs avec état déployés dans un centre de données sur site, vous pouvez utiliser AWS Elastic Disaster Recovery pour protéger vos charges de travail dans AWS. Si votre charge de travail est déjà hébergée dans AWS, vous pouvez utiliser Elastic Disaster Recovery pour la protéger dans une autre zone ou région de disponibilité. Elastic Disaster

Recovery utilise une réplication continue au niveau des blocs vers une zone de transit légère pour fournir une restauration rapide et fiable des applications sur site et dans le cloud.

Étapes d'implémentation

1. Implémentation de l'autorégénération Dans la mesure du possible, déployez vos instances ou vos conteneurs en utilisant la scalabilité automatique. Si vous ne pouvez pas utiliser la scalabilité automatique, utilisez la récupération automatique pour les instances EC2 ou mettez en place un mécanisme d'autoréparation basé sur Amazon EC2 ou des événements de cycle de vie de conteneur ECS.
   - Utilisez les groupes Amazon EC2 Auto Scaling pour les instances et les charges de travail de conteneur qui n'ont aucune exigence en matière d'adresse IP d'instance, d'adresse IP privée, d'adresse IP élastique et de métadonnées d'instance.
     - Les données utilisateur du modèle de lancement peuvent être utilisées pour mettre en place un mécanisme permettant la récupération automatique de la plupart des charges de travail.
   - Utilisez la récupération automatique des instances Amazon EC2 pour les charges de travail nécessitant une seule adresse d'ID d'instance, une seule adresse IP privée, une seule adresse IP élastique, et des métadonnées d'instance.
     - La récupération automatique envoie des alertes de statut de récupération à une rubrique SNS lorsque la défaillance de l'instance est détectée.
   - Utilisez les événements du cycle de vie de l'instance Amazon EC2 ou les événements Amazon ECS pour automatiser l'autoréparation lorsque la scalabilité automatique ou la récupération de votre instance EC2 ne peuvent pas être utilisées.
     - Utilisez les événements pour appeler le mécanisme vous permettant de réparer votre composant selon la logique de processus dont vous avez besoin.
   - Protégez les charges de travail avec état qui sont limitées à un seul emplacement en utilisant AWS Elastic Disaster Recovery.

Ressources

Documents connexes :

- Événements Amazon ECS
- Hooks de cycle de vie Amazon EC2 Auto Scaling
- Récupérer votre instance.
- Scalabilité automatique des services

- Qu'est-ce que Amazon EC2 Auto Scaling ?
- AWS Elastic Disaster Recovery

REL10-BP04 Utiliser des architectures cloisonnées pour limiter la portée de l'impact

Mettez en œuvre des architectures de cloisonnement (également connues sous le nom d'architectures cellulaires) pour restreindre l'effet d'une panne au sein d'une charge de travail à un nombre limité de composants.

Résultat souhaité : une architecture cellulaire utilise plusieurs instances isolées d'une charge de travail, où chaque instance est appelée cellule. Chaque cellule est indépendante, ne partage pas d'état avec les autres cellules et traite un sous-ensemble des demandes de la charge de travail globale. Cela réduit l'impact potentiel d'une panne, telle qu'une mauvaise mise à jour logicielle, sur une cellule individuelle et les demandes qu'elle traite. Si une charge de travail utilise 10 cellules pour traiter 100 demandes, lorsqu'une panne survient, 90 % des demandes globales ne sont pas affectées par la panne.

Anti-modèles courants :

- Permettre aux cellules de se développer sans limites.
- Appliquer des mises à jour ou des déploiements de code à toutes les cellules en même temps.
- Partage de l'état ou des composants entre les cellules (à l'exception de la couche routeur).
- Ajout d'une logique métier ou de routage complexe à la couche routeur.
- Ne pas minimiser les interactions entre les cellules.

Avantages liés au respect de cette bonne pratique : avec les architectures cellulaires, de nombreux types de pannes courants sont contenus dans la cellule elle-même, ce qui permet un isolement supplémentaire des pannes. Ces limites de défaillance peuvent fournir une résilience contre des types de panne qui seraient autrement difficiles à contenir, tels que des déploiements de code infructueux ou des requêtes corrompues ou déclenchant un mode de défaillance spécifique (également connues sous le nom de requêtes empoisonnées).

Directives d'implémentation

Sur un navire, les cloisons permettent de contenir une brèche dans la coque dans une seule section de la coque. Dans les systèmes complexes, ce modèle est souvent répliqué pour permettre d'isoler

des pannes. Les limites isolées pour les défaillances restreignent l'effet d'une panne au sein d'une charge de travail à un nombre limité de composants. Les composants en dehors de la limite ne sont pas affectés par la défaillance. En utilisant plusieurs limites isolées par défaut, vous pouvez limiter l'impact sur votre charge de travail. Sur AWS, les clients peuvent utiliser plusieurs zones de disponibilité et régions pour isoler des pannes, mais le concept d'isolement des pannes peut également être étendu à l'architecture de votre charge de travail.

La charge de travail globale est divisée en cellules par une clé de partition. Cette clé doit s'aligner sur la base de granularité du service, ou sur la manière naturelle dont la charge de travail d'un service peut être subdivisée avec un minimum d'interactions entre les cellules. Des exemples de clés de partition sont l'ID du client, l'ID de la ressource ou tout autre paramètre facilement accessible dans la plupart des appels d'API. Une couche de routage des cellules distribue les requêtes aux cellules individuelles en fonction de la clé de partition et présente un point de terminaison unique aux clients.

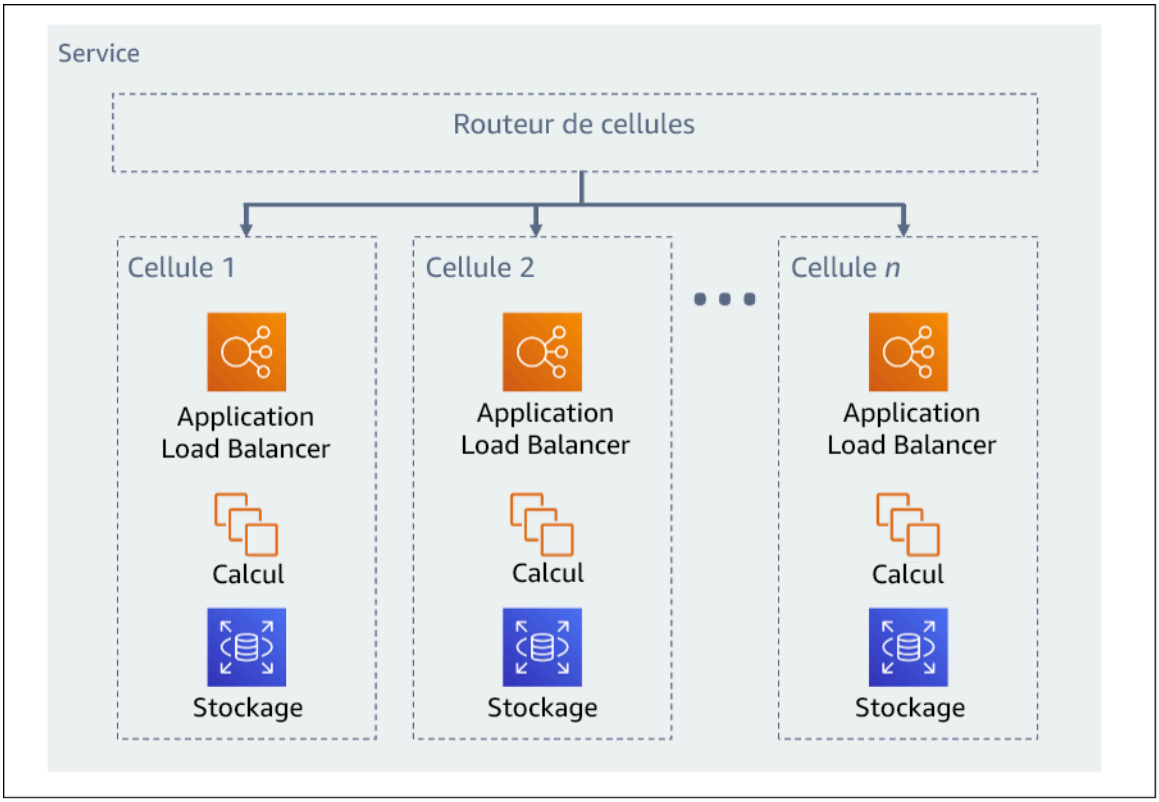


Figure 11 : Architecture cellulaire

Étapes d'implémentation

Lors de la conception d'une architecture cellulaire, vous devez tenir compte de plusieurs éléments :

1. Clé de partition : il convient d'accorder une attention particulière au choix de la clé de partition.

- Celle-ci doit s'aligner sur la base de granularité du service, ou sur la manière naturelle dont la charge de travail d'un service peut être subdivisée avec un minimum d'interactions entre les cellules. Voici quelques exemples : `ID du client` ou `ID de la ressource`. »
- La clé de partition doit être disponible dans toutes les requêtes, soit directement, soit d'une manière qui pourrait être facilement déduite de façon déterministe par d'autres paramètres.

2. Mappage persistant des cellules : les services en amont ne doivent interagir qu'avec une seule cellule pendant le cycle de vie de leurs ressources.
   - En fonction de la charge de travail, vous devrez peut-être concevoir une stratégie de migration de cellules pour faire migrer les données d'une cellule à l'autre. La migration d'une cellule peut s'avérer nécessaire si un utilisateur ou une ressource particulière de votre charge de travail devient trop importante et nécessite une cellule dédiée.
   - Les cellules ne doivent pas partager d'état ou de composants entre elles.
   - Par conséquent, les interactions entre cellules doivent être évitées ou réduites au minimum, car elles créent des dépendances entre les cellules et diminuent donc les bénéfices de l'isolement des défaillances.
3. Couche routeur : la couche routeur est un composant partagé entre les cellules, et ne peut donc pas suivre la stratégie de compartimentage des cellules.
   - Nous recommandons de paramétrer la couche routeur pour distribuer les requêtes à des cellules individuelles à l'aide d'un algorithme de mappage de partition d'une manière efficace sur le plan des calculs. Par exemple, en combinant des fonctions de hachage cryptographiques et de l'arithmétique modulaire pour mapper les clés de partition aux cellules.
   - Pour éviter les impacts sur plusieurs cellules, la couche de routage doit rester aussi simple et évolutive horizontalement que possible, ce qui nécessite d'éviter toute logique métier complexe au sein de cette couche. Cela présente l'avantage supplémentaire de faciliter la compréhension de son comportement attendu à tout moment, favorisant ainsi une testabilité approfondie. Comme l'explique Colm MacCárthaigh dans son article Reliability, constant work, and a good cup of coffee (Fiabilité, travail constant, et une bonne tasse de café), les conceptions simples et les modèles de travail constants produisent des systèmes fiables et réduisent le phénomène d'antifragilité.
4. Taille des cellules : les cellules doivent comporter une taille maximale définie et ne doivent pas être autorisées à se développer au-delà.
   - La taille maximale doit être identifiée en effectuant des tests approfondis, jusqu'à ce que les points de rupture soient atteints et que des marges de fonctionnement sûres soient établies.

Pour obtenir plus de détails sur la mise en œuvre des pratiques de test, consultez REL07-BP04 Effectuer un test de charge de votre charge de travail

- La charge de travail globale doit se développer en ajoutant des cellules supplémentaires, ce qui lui permet de s'adapter à l'augmentation de la demande.

5. Stratégies multi-AZ ou multirégion : il convient de tirer parti de plusieurs couches de résilience pour se protéger contre différents domaines de panne.
   - Pour la résilience, vous devez adopter une approche qui repose sur des couches de défense. Une couche protège contre les perturbations de petite envergure et courantes en créant une architecture hautement disponible à l'aide de plusieurs AZ. Une autre couche de défense est destinée à protéger contre les événements rares tels que les catastrophes naturelles généralisées et les perturbations au niveau régional. Cette deuxième couche implique de concevoir l'architecture de votre application pour qu'elle s'étende sur plusieurs Régions AWS. La mise en œuvre d'une stratégie multirégion pour votre charge de travail permet de la protéger contre les catastrophes naturelles généralisées qui affectent une grande région géographique d'un pays, ou les défaillances techniques à l'échelle régionale. Sachez que la mise en œuvre d'une architecture multirégion peut être très complexe et n'est généralement pas requise pour la plupart des charges de travail. Pour en savoir plus, consultez REL10-BP02 Sélectionner les emplacements appropriés pour votre déploiement multisite. »
6. Déploiement du code : il faut privilégier une stratégie de déploiement de code échelonnée plutôt que de déployer les modifications de code dans toutes les cellules en même temps.
   - Cela permettra de minimiser les risques de panne de plusieurs cellules en raison d'un mauvais déploiement ou d'une erreur humaine. Pour obtenir plus de détails, consultez Automatisation de déploiements sécurisés sans intervention.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Ressources

Bonnes pratiques associées :

- REL07-BP04 Effectuer un test de charge de votre charge de travail
- REL10-BP02 Sélectionner les emplacements appropriés pour votre déploiement multisite

Documents connexes :

- Reliability, constant work, and a good cup of coffee (Fiabilité, travail constant, et une bonne tasse de café)
- AWS and Compartmentalization (AWS et le cloisonnement)
- Isolation de la charge de travail à l'aide du partitionnement aléatoire
- Automatisation de déploiements sécurisés sans intervention

Vidéos connexes :

- AWS re:Invent 2018: Close Loops and Opening Minds: How to Take Control of Systems, Big and Small
- AWS re:Invent 2018: How AWS Minimizes the Blast Radius of Failures (ARC338)
- Shuffle-sharding: AWS re:Invent 2019: Introducing The Amazon Builders' Library (DOP328)
- AWS Summit ANZ 2021 - Everything fails, all the time: Designing for resilience (Sommet AWS ANZ 2021 - tout échoue, tout le temps : concevoir pour la résilience)

Exemples connexes :

- Atelier Well-Architected : isolement des pannes avec le partitionnement aléatoire

Conception d'une charge de travail qui résiste aux défaillances des composants

Les charges de travail exigeant une haute disponibilité et un faible temps moyen de récupération (MTTR) doivent être conçues pour être résilientes.

Bonnes pratiques

- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances
- REL11-BP02 Basculer vers des ressources saines
- REL11-BP03 Automatiser la réparation sur toutes les couches
- REL11-BP04 S'appuyer sur le plan de données et non sur le plan de contrôle pendant la récupération
- REL11-BP05 Utiliser la stabilité statique pour éviter les comportements bimodaux
- REL11-BP06 Envoyer des notifications lorsque des événements affectent la disponibilité

- REL11-BP07 Concevoir votre produit pour atteindre les objectifs de disponibilité et les accords de niveau de service (SLA)

REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances

Surveillez en continu l'état de votre charge de travail afin que vous et vos systèmes automatisés ayez connaissance des dégradations ou des défaillances dès qu'elles se produisent. Surveillez les indicateurs clés de performance (KPI) en fonction de la valeur commerciale.

Tous les mécanismes de récupération et de réparation doivent commencer par la capacité à détecter rapidement les problèmes. Les défaillances techniques doivent être détectées au préalable pour être résolues. Cependant, la disponibilité repose sur la capacité de votre charge de travail à fournir une valeur commerciale. Il doit donc s'agir d'indicateurs clés de performance (KPI) de votre stratégie de détection et de correction.

Résultat souhaité : Les composants essentiels d'une charge de travail sont surveillés de manière indépendante afin de détecter les défaillances et de les signaler au moment et à l'emplacement où elles se produisent.

Anti-modèles courants :

- Aucune alarme n'a été configurée. Il n'y a donc pas de notification lorsque des interruptions se produisent.
- Des alarmes existent, mais les seuils ne laissent pas assez de temps pour réagir.
- Les métriques ne sont pas collectées à une fréquence suffisante pour atteindre l'objectif de délai de reprise (RTO).
- Seules les interfaces de la charge de travail axées directement sur le client sont activement surveillées.
- Collecte uniquement des métriques techniques et non des métriques de fonction commerciale.
- Aucune métrique ne mesure l'expérience utilisateur de la charge de travail.
- Trop de contrôleurs sont créés.

Avantages liés au respect de cette bonne pratique : La surveillance appropriée à tous les niveaux vous permet de raccourcir le délai de reprise en réduisant le temps de détection.

Niveau d'exposition au risque si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Identifiez toutes les charges de travail qui seront examinées à des fins de surveillance. Une fois que vous avez identifié tous les composants de la charge de travail à surveiller, déterminez l'intervalle de surveillance. Cet intervalle a un impact direct sur la rapidité avec laquelle la restauration peut être initiée en fonction du temps nécessaire pour détecter une panne. Le délai moyen de détection (MTTD) est le délai entre le moment où une panne survient et le moment où les opérations de réparation commencent. La liste des services doit être longue et complète.

La surveillance doit couvrir toutes les couches de la pile d'applications, y compris l'application, la plate-forme, l'infrastructure et le réseau.

Votre stratégie de surveillance doit tenir compte de l'impact des défaillances grises. Pour plus de détails sur les défaillances grises, voir Défaillances grises dans le livre blanc sur les modèles de résilience multi-AZ avancés.

Étapes d'implémentation

- Votre intervalle de surveillance dépend de la vitesse à laquelle vous devez effectuer la récupération. Votre délai de reprise dépend du temps nécessaire à la récupération. Vous devez donc déterminer la fréquence de collecte en tenant compte de cette durée et de votre objectif de délai de reprise (RTO).
- Configurez la surveillance détaillée des composants et des services gérés.
  - Déterminez si la surveillance détaillée des instances EC2 et Auto Scaling est nécessaire. La surveillance détaillée fournit des métriques à intervalle d'une minute, et la surveillance par défaut fournit des métriques à intervalle de cinq minutes.
  - Déterminez si la surveillance améliorée pour RDS est nécessaire. La surveillance améliorée utilise un agent sur les instances RDS pour obtenir des informations utiles sur différents processus ou threads.
  - Déterminez les exigences de surveillance des composants sans serveur critiques pour Lambda, API Gateway, Amazon EKS, Amazon ECSet tous les types d'équilibreurs de charge.
  - Déterminez les exigences de surveillance des composants de stockage pour Amazon S3, Amazon FSx, Amazon EFSet Amazon EBS.
- Créez des métriques personnalisées pour mesurer les indicateurs clés de performance (KPI) de l'entreprise. Les charges de travail mettent en œuvre des fonctions commerciales stratégiques, qui

doivent être utilisées comme indicateurs clés de performance permettant d'identifier les problèmes indirects.

- Surveillez l'expérience utilisateur pour détecter les défaillances à l'aide de tests canary utilisateur. Les tests de transactions synthétiques (également appelés « tests canary », à ne pas confondre avec les déploiements canary) qui peuvent exécuter et simuler le comportement des clients font partie des processus de test les plus importants. Exécutez ces tests en permanence sur vos points de terminaison de charge de travail à partir de divers emplacements distants.
- Créez des métriques personnalisées qui suivent l'expérience de l'utilisateur. Si vous pouvez analyser l'expérience du client, vous pouvez savoir à quel moment l'expérience du consommateur se dégrade.
- Définissez des alarmes pour détecter quand une partie de votre charge de travail ne fonctionne pas correctement et pour indiquer quand mettre automatiquement à l'échelle les ressources. Le système peut afficher les alarmes sur des tableaux de bord, envoyer des alertes via Amazon SNS ou par e-mail et fonctionner avec Auto Scaling pour une mise à l'échelle à la hausse ou à la baisse des ressources de la charge de travail.
- Créez des tableaux de bord pour visualiser vos métriques. Les tableaux de bord peuvent être utilisés pour afficher visuellement des tendances, des valeurs aberrantes et d'autres indicateurs de problèmes potentiels, ou pour fournir une indication des problèmes que vous pourriez vouloir examiner.
- Créez une surveillance distribuée pour vos services. Avec la surveillance distribuée, vous pouvez analyser les performances de votre application et de ses services sous-jacents, afin d'identifier et de dépanner la cause première des problèmes et des erreurs de performances.
- Créez les systèmes de surveillance (en utilisant CloudWatch ou X-Ray), les tableaux de bord et la collecte de données dans une région et un compte distincts.
- Créez une intégration pour la surveillance Amazon Health Aware pour permettre d'identifier les ressources AWS susceptibles de subir des dégradations. Pour les charges de travail essentielles à l'entreprise, cette solution permet d'accéder à des alertes proactives et en temps réel pour les services AWS.

Ressources

Bonnes pratiques associées :

- Définition de la disponibilité
- REL11-BP06 Envoi de notifications lorsque des événements affectent la disponibilité

Documents connexes :

- Amazon CloudWatch Synthetics vous permet de créer des tests canary utilisateur
- Activer ou désactiver la surveillance détaillée pour votre instance
- Surveillance améliorée
- Surveillance de vos groupes et instances Auto Scaling à l'aide d'Amazon CloudWatch
- Publication des métriques personnalisées
- Utilisation des alarmes Amazon CloudWatch
- Fonctionnement des tableaux de bord CloudWatch
- Utilisation de tableaux de bord CloudWatch interrégionaux entre comptes
- Utilisation du suivi X-Ray interrégional entre comptes
- Compréhension de la disponibilité
- Implémentation d'Amazon Health Aware (AHA)

Vidéos connexes :

- Mitigating gray failures

Exemples connexes :

- Atelier Well-Architected : niveau 300 : implémentation de la surveillance de l'état et gestion des dépendances pour améliorer la fiabilité
- Un atelier sur l'observabilité : explorer X-Ray

Outils associés :

- CloudWatch
- CloudWatch X-Ray

REL11-BP02 Basculer vers des ressources saines

En cas de défaillance des ressources, les ressources saines doivent continuer à répondre aux requêtes. Pour les altérations liées à l'emplacement (par exemple, zone de disponibilité ou Région

AWS), vérifiez que des systèmes sont en place pour basculer vers des ressources saines dans des emplacements intacts.

Lorsque vous concevez un service, répartissez la charge entre les ressources, les zones de disponibilité ou les régions. Ainsi, la défaillance d'une ressource individuelle ou l'altération peut être atténuée en déplaçant le trafic vers les ressources saines restantes. Réfléchissez à la manière dont les services sont découverts et acheminés en cas de défaillance.

Concevez vos services en tenant compte de la restauration après panne. Chez AWS, nous concevons les services afin de minimiser le temps de restauration en cas de défaillance, ainsi que l'impact sur les données. Nos services utilisent principalement des magasins de données qui valident les requêtes uniquement lorsque les données sont stockées durablement sur plusieurs réplicas au sein d'une région. Ils sont élaborés de manière à utiliser l'isolation basée sur les cellules et à faire appel à l'isolement des pannes fourni par des zones de disponibilité. Nous utilisons largement l'automatisation dans nos procédures opérationnelles. Nous optimisons également notre fonction de remplacement et redémarrage afin de récupérer rapidement en cas d'interruptions.

Les modèles et les conceptions qui permettent le basculement varient pour chaque service de plateforme AWS. De nombreux services natifs gérés par AWS sont dotés de plusieurs zones de disponibilité (comme Lambda ou API Gateway). D'autres services AWS (comme EC2 et EKS) nécessitent des conceptions répondant à des bonnes pratiques spécifiques pour prendre en charge le basculement des ressources ou le stockage des données entre les zones de disponibilité.

La surveillance doit être configurée pour vérifier que la ressource de basculement est saine, suivre la progression du basculement des ressources et surveiller le rétablissement des processus métier.

Résultat souhaité : Les systèmes sont capables d'utiliser automatiquement ou manuellement de nouvelles ressources pour se remettre d'une dégradation.

Anti-modèles courants :

- La planification des défaillances ne fait pas partie de la phase de planification et de conception.
- Le RTO et le RPO ne sont pas établis.
- Surveillance insuffisante pour détecter les ressources défaillantes.
- Isolement approprié des domaines de défaillance.
- Le basculement multirégional n'est pas pris en compte.
- La détection des défaillances est trop sensible ou trop agressive lors de la décision de basculer.
- Pas de test ni de validation de la conception du basculement.

- Automatisation de la réparation automatique sans notification indiquant que la réparation était nécessaire.
- Pas de période d'attente pour éviter un failback trop précoce.

Avantages liés au respect de cette bonne pratique : Vous pouvez créer des systèmes plus résilients qui préservent leur fiabilité en cas de défaillance en se dégradant progressivement et en se rétablissant rapidement.

Niveau d'exposition au risque si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Les services AWS, comme Elastic Load Balancing et Amazon EC2 Auto Scalingcontribuent à répartir la charge entre les ressources et les zones de disponibilité. Par conséquent, la défaillance d'une ressource individuelle (telle qu'une instance EC2) ou l'altération d'une zone de disponibilité peut être atténuée en déplaçant le trafic vers les ressources saines restantes.

Pour les charges de travail multirégionales, les conceptions sont plus complexes. Par exemple, les réplicas en lecture entre régions vous permettent de déployer vos données sur plusieurs Régions AWS. Cependant, le basculement est toujours nécessaire pour faire passer le réplica en lecture au niveau principal, puis pour rediriger le trafic vers le nouveau point de terminaison. Amazon Route 53, Route 53 Route 53 ARC, CloudFront et AWS Global Accelerator aident à acheminer le trafic entre les Régions AWS.

Les services AWS, comme Amazon S3, Lambda, API Gateway, Amazon SQS, Amazon SNS, Amazon SES, Amazon Pinpoint, Amazon ECR, AWS Certificate Manager, EventBridge ou Amazon DynamoDB, sont automatiquement déployés dans plusieurs zones de disponibilité par AWS. En cas de défaillance, ces services AWS acheminent automatiquement le trafic vers des emplacements sains. Les données sont stockées de manière redondante dans plusieurs zones de disponibilité et restent disponibles.

Pour Amazon RDS, Amazon Aurora, Amazon Redshift, Amazon EKS ou Amazon ECS, Multi-AZ est une option de configuration. AWS peut diriger le trafic vers l'instance saine si le basculement est initié. Cette action de basculement peut être prise par AWS ou conformément aux exigences du client.

Pour les instances Amazon EC2, Amazon Redshift, les tâches Amazon ECS ou les pods Amazon EKS, vous choisissez les zones de disponibilité dans lesquelles vous souhaitez effectuer le déploiement. Pour certaines conceptions, Elastic Load Balancing fournit la solution permettant de

détecter les instances dans les zones défectueuses et d'acheminer le trafic vers les zones saines. Elastic Load Balancing peut également acheminer le trafic vers les composants de votre centre de données sur site.

Pour le basculement du trafic multirégional, le réacheminement peut tirer parti d'Amazon Route 53, Route 53 ARC, AWS Global Accelerator, Route 53 Private DNS for VPCs ou CloudFront pour fournir un moyen de définir des domaines Internet et d'attribuer des politiques de routage, y compris des surveillances de l'état, afin d'acheminer le trafic vers des régions saines. AWS Global Accelerator fournit des adresses IP statiques qui agissent comme point d'entrée fixe vers votre application, puis acheminent le trafic vers les points de terminaison des Régions AWS de votre choix, en utilisant le réseau mondial AWS plutôt qu'Internet pour de meilleures performances et une meilleure fiabilité.

Étapes d'implémentation

- Créez des modèles de basculement pour toutes les applications et tous les services appropriés. Isolez chaque composant de l'architecture et créez des conceptions de basculement respectant le RTO et le RPO pour chacun d'eux.
- Configurez les environnements de bas niveau (tels que le développement ou les tests) avec tous les services requis pour disposer d'un plan de basculement. Déployez les solutions en utilisant l'infrastructure en tant que code (IaC) pour garantir la reproductibilité.
- Configurez un site de reprise tel qu'une deuxième région pour implémenter et tester les modèles de basculement. Si nécessaire, les ressources pour les tests peuvent être configurées temporairement afin de limiter les coûts supplémentaires.
- Déterminez quels plans de basculement seront automatisés par AWS, ceux qui peuvent être automatisés par un processus DevOps et ceux qui peuvent être manuels. Documentez et mesurez le RTO et le RPO de chaque service.
- Créez un manuel de basculement et incluez toutes les étapes nécessaires au basculement de chaque ressource, application et service.
- Créez un manuel de failback et incluez toutes les étapes nécessaires (avec calendrier) pour chaque ressource, application et service
- Créez un plan pour lancer et répéter le manuel. Utilisez des simulations et des tests de chaos pour tester les étapes du manuel et l'automatisation.
- Pour toute altération liée à l'emplacement (par exemple, zone de disponibilité ou Région AWS), vérifiez que des systèmes sont en place pour basculer vers des ressources saines dans des emplacements intacts. Vérifiez le quota, les niveaux de mise à l'échelle automatique et les ressources en cours d'exécution avant le test de basculement.

Ressources

Bonnes pratiques Well-Architected connexes :

- REL13 – Plan de reprise après sinistre
- REL10 – Utilisation de l'isolation des défaillances pour protéger votre charge de travail

Documents connexes :

- Définition des objectifs RTO et RPO
- Configuration Route 53 ARC avec des Application Load Balancers
- Basculement à l'aide du routage pondéré Route 53
- Reprise après sinistre avec Route 53 ARC
- EC2 avec mise à l'échelle automatique
- Déploiements EC2 – Multi-AZ
- Déploiements ECS – Multi-AZ
- Changer de trafic avec Route 53 ARC
- Lambda avec un Application Load Balancer et un basculement
- Réplication et basculement ACM
- Réplication et basculement du magasin de paramètres
- Réplication entre régions ECR et basculement
- Configuration de la réplication entre régions du gestionnaire de secrets
- Activation de la réplication entre régions pour EFS et basculement
- Réplication entre régions EFS et basculement
- Basculement du réseau
- Basculement des points de terminaison S3 à l'aide du protocole MRAP
- Création d'une réplication entre régions pour S3
- API Gateway régional de basculement avec Route 53 ARC
- Basculement à l'aide d'un accélérateur mondial multirégional
- Basculement avec DRS
- Création de mécanismes de reprise après sinistre à l'aide d'Amazon Route 53

Exemples connexes :

- Reprise après sinistre sur AWS
- Elastic Disaster Recovery sur AWS

REL11-BP03 Automatiser la réparation sur toutes les couches

Utilisez des capacités automatisées pour effectuer des actions correctives en cas de détection d'une défaillance. Les dégradations peuvent être automatiquement corrigées par le biais de mécanismes de service internes ou peuvent nécessiter le redémarrage ou la suppression des ressources par le biais d'actions correctives.

Pour les applications autogérées et la réparation interrégionale, les modèles de restauration et les processus de réparation automatisés peuvent être extraits des bonnes pratiques existantes.

Pouvoir redémarrer ou supprimer une ressource est important pour remédier aux défaillances. Une bonne pratique consiste à rendre les services sans état dans la mesure du possible. Cela évite toute perte de données ou de disponibilité au redémarrage des ressources. Dans le cloud, vous pouvez (et devriez généralement) remplacer la totalité de la ressource (par exemple, une instance de calcul ou une fonction sans serveur) dans le cadre du redémarrage. Le redémarrage proprement dit est un moyen simple et fiable de récupération après une défaillance. De nombreux types de défaillances différents se produisent dans les charges de travail. Les défaillances peuvent se produire au niveau du matériel, des logiciels, des communications et des opérations.

Le redémarrage ou la nouvelle tentative s'appliquent également aux requêtes réseau. Appliquez la même approche de récupération à la fois pour un délai d'expiration réseau et une défaillance de la dépendance, si la dépendance renvoie une erreur. Comme ces deux événements ont un effet semblable sur le système, plutôt que d'essayer de traiter l'un ou l'autre comme un « cas particulier », appliquez une stratégie semblable de nouvelle tentative limitée avec un backoff exponentiel et une instabilité. La possibilité d'exécuter un redémarrage est un mécanisme de récupération présenté dans l'informatique orientée récupération et dans les architectures de cluster haute disponibilité.

Résultat souhaité : Des actions automatisées sont effectuées pour remédier à la détection d'une panne.

Anti-modèles courants :

- Allocation des ressources sans mise à l'échelle automatique.

- Déploiement d'applications une par une dans des instances ou des conteneurs.
- Déploiement d'applications qui ne peuvent pas être déployées dans plusieurs emplacements sans utiliser la récupération automatique.
- Réparation manuelle des applications impossible à réparer par la mise à l'échelle et la récupération automatiques.
- Aucune automatisation pour le basculement des bases de données.
- Absence de méthodes automatisées pour rediriger le trafic vers de nouveaux points de terminaison.
- Aucune réplication du stockage.

Avantages liés au respect de cette bonne pratique : La réparation automatisée contribue à réduire le temps moyen de récupération et à améliorer votre disponibilité.

Niveau d'exposition au risque si cette bonne pratique n'est pas respectée : Élevé

Directives d'implémentation

Les conceptions pour Amazon EKS ou les autres services Kubernetes doivent inclure à la fois un minimum et un maximum de réplicas ou d'ensembles dynamiques, ainsi que le dimensionnement minimal des clusters et des groupes de nœuds. Ces mécanismes mettent à disposition un minimum de ressources de traitement en permanence tout en corrigeant automatiquement les défaillances à l'aide du plan de contrôle Kubernetes.

Les modèles de conception accessibles via un équilibreur de charge utilisant des clusters de calcul doivent tirer parti des groupes Auto Scaling. Elastic Load Balancing (ELB) distribue automatiquement le trafic applicatif entrant sur plusieurs cibles et appareils virtuels dans une ou plusieurs zones de disponibilité (AZ).

La taille des conceptions basées sur le calcul en cluster qui n'utilisent pas l'équilibrage de charge doit être conçue pour la perte d'au moins un nœud. Cela permet au service de continuer à fonctionner avec une capacité potentiellement réduite pendant la restauration d'un nouveau nœud. Exemples de services : Mongo, DynamoDB Accelerator, Amazon Redshift, Amazon EMR, Cassandra, Kafka, MSK-EC2, Couchbase, ELK et Amazon OpenSearch Service. Bon nombre de ces services peuvent être conçus avec des fonctionnalités supplémentaires de réparation automatique. Certaines technologies de cluster doivent générer une alerte en cas de perte d'un nœud, déclenchant un flux de travail automatique ou manuel pour recréer un nœud. Ce flux de travail peut être automatisé avec AWS Systems Manager pour résoudre rapidement les problèmes.

Amazon EventBridge peut être utilisé pour surveiller et filtrer les événements tels que les alarmes CloudWatch ou les changements d'état dans d'autres services AWS. Sur la base des informations relatives aux événements, il peut ensuite appeler AWS Lambda, Systems Manager Automation ou d'autres cibles pour exécuter une logique de correction personnalisée sur votre charge de travail. Amazon EC2 Auto Scaling peut être configuré pour vérifier l'état de l'instance EC2. Si l'instance est dans un état autre que celui en cours d'exécution, ou si le statut du système est dégradé, Amazon EC2 Auto Scaling considère l'instance comme défectueuse et lance une instance de remplacement. Pour les remplacements à grande échelle (comme la perte d'une zone de disponibilité complète), il est préférable d'opter pour la stabilité statique pour une haute disponibilité.

Étapes d'implémentation

- Utilisez des groupes Auto Scaling pour déployer des niveaux dans une charge de travail. Auto Scaling peut effectuer une autoréparation sur les applications sans état et ajouter ou supprimer de la capacité.
- Pour les instances de calcul mentionnées précédemment, utilisez l'équilibrage de charge et choisissez le type d'équilibreur de charge approprié.
- Envisagez la réparation pour Amazon RDS. Avec les instances de secours, configurez le basculement automatique vers l'instance de secours. Pour les réplicas en lecture Amazon RDS, un flux de travail automatisé est nécessaire pour rendre un réplica en lecture principal.
- Implémentez la restauration automatique sur les instances EC2 dont les applications déployées ne peuvent pas être déployées sur plusieurs sites et qui peuvent tolérer un redémarrage en cas de panne. La récupération automatique peut être utilisée pour remplacer du matériel défaillant et redémarrer l'instance lorsque l'application ne peut pas être déployée sur plusieurs emplacements. Les métadonnées de l'instance et les adresses IP associées sont conservées, ainsi que les volumes EBS et les points de montage vers Amazon Elastic File System ou le systèmes de fichiers pour Lustre et Windows. Avec AWS OpsWorks, vous pouvez configurer la réparation automatique des instances EC2 au niveau de la couche.
- Mettez en œuvre une restauration automatique avec AWS Step Functions et AWS Lambda lorsque vous ne pouvez pas utiliser la mise à l'échelle automatique ou la récupération automatique, ou lorsque la récupération automatique échoue. Lorsque vous ne pouvez pas utiliser la scalabilité automatique, que vous ne pouvez pas utiliser la récupération automatique ou que la récupération automatique échoue, vous pouvez automatiser la réparation à l'aide d'AWS Step Functions et d'AWS Lambda.
- Amazon EventBridge peut être utilisé pour surveiller et filtrer des événements tels que des alarmes CloudWatch ou des changements d'état dans d'autres services AWS. En fonction des informations

d'événement, il peut ensuite déclencher AWS Lambda (ou d'autres cibles) pour exécuter une logique de correction personnalisée sur votre charge de travail.

Ressources

Bonnes pratiques associées :

- Définition de la disponibilité
- REL11-BP01 Surveillance de tous les composants de la charge de travail pour détecter les défaillances

Documents connexes :

- Fonctionnement d'AWS Auto Scaling
- Récupération automatique Amazon EC2
- Amazon Elastic Block Store (Amazon EBS)
- Amazon Elastic File System (Amazon EFS)
- Qu'est-ce que Amazon FSx for Lustre ?
- Qu'est-ce que Amazon FSx for Windows File Server ?
- AWS OpsWorks : utilisation de la réparation automatique pour remplacer des instances défectueuses
- Qu'est-ce qu'AWS Step Functions ?
- Qu'est-ce qu'AWS Lambda ?
- Qu'est-ce qu'Amazon EventBridge ?
- Utilisation des alarmes Amazon CloudWatch
- Basculement Amazon RDS
- SSM – Systems Manager Automation
- Bonnes pratiques en matière d'architecture résiliente

Vidéos connexes :

- Automatically Provision and Scale OpenSearch Service
- Amazon RDS Failover Automatically

Exemples connexes :

- Atelier sur Auto Scaling
- Atelier sur le basculement Amazon RDS

Outils associés :

- CloudWatch
- CloudWatch X-Ray

REL11-BP04 S'appuyer sur le plan de données et non sur le plan de contrôle pendant la récupération

Les plans de contrôle fournissent les API administratives utilisées pour créer, lire et décrire, mettre à jour, supprimer et répertorier (CRUDL) les ressources, tandis que les plans de données gèrent le trafic quotidien des services. Lorsque vous mettez en œuvre des réponses de restauration ou d'atténuation en cas d'événements susceptibles d'avoir un impact sur la résilience, concentrez-vous sur l'utilisation d'un nombre minimal d'opérations du plan de contrôle pour récupérer, redimensionner, restaurer, réparer ou basculer le service. L'action du plan de données doit remplacer toute activité lors de ces événements de dégradation.

Par exemple, les actions suivantes font toutes partie du plan de contrôle : lancement d'une nouvelle instance de calcul, création d'un stockage par blocs et description des services de file d'attente. Lorsque vous lancez des instances de calcul, le plan de contrôle doit effectuer plusieurs tâches, telles que la recherche d'un hôte physique avec la capacité suffisante, l'allocation d'interfaces réseau, la préparation de volumes locaux de stockage par blocs, la génération d'informations d'identification et l'ajout de règles de sécurité. Les plans de contrôle relèvent souvent d'une orchestration complexe.

Résultat souhaité : Lorsqu'une ressource passe à un état altéré, le système peut être rétabli automatiquement ou manuellement en transférant le trafic des ressources altérées vers des ressources saines.

Anti-modèles courants :

- Nécessité de modifier les enregistrements DNS pour rediriger le trafic.
- Nécessité de réaliser des opérations de dimensionnement du plan de contrôle pour remplacer les composants endommagés en raison de ressources sous-provisionnées.

- Utilisation d'actions de plan de contrôle étendues, multiservices et multi-API pour remédier à toute catégorie d'altération.

Avantages liés au respect de cette bonne pratique : L'augmentation du taux de réussite de la correction automatisée contribue à réduire le temps moyen de récupération et à améliorer la disponibilité de la charge de travail.

Niveau d'exposition au risque si cette bonne pratique n'est pas respectée : Moyen : pour certains types de dégradations de service, les plans de contrôle sont concernés. La nécessité d'utiliser de manière intensive le plan de contrôle pour la correction peut augmenter le délai de reprise (RTO) et le temps moyen de récupération (MTTR).

Directives d'implémentation

Pour limiter les actions du plan de données, évaluez chaque service pour déterminer les actions nécessaires afin de restaurer le service.

Tirez parti d'Amazon Route 53 Application Recovery Controller pour déplacer le trafic DNS. Ces fonctionnalités surveillent en permanence la capacité de votre application à se rétablir suite à des défaillances et vous permettent de contrôler la reprise de votre application dans plusieurs Régions AWS, plusieurs zones de disponibilité et sur site.

Les politiques de routage Route 53 utilisent le plan de contrôle. Ne vous fiez donc pas à celui-ci pour la récupération. Les plans de données Route 53 répondent aux requêtes DNS et effectuent et évaluent des surveillances de l'état. Ils sont distribués dans le monde entier et conçus pour un contrat de niveau de service (SLA) de 100 % de disponibilité..

Les API et consoles de gestion Route 53 dans lesquelles vous créez, mettez à jour et supprimez des ressources Route 53 s'exécutent sur des plans de contrôle conçus pour donner la priorité à la cohérence forte et à la durabilité dont vous avez besoin lors de la gestion du DNS. Pour ce faire, les plans de contrôle sont situés dans une seule région : USA Est (Virginie du Nord). Bien que les deux systèmes soient conçus pour être très fiables, les plans de contrôle ne sont pas inclus dans le SLA. Dans de rares cas, la conception résiliente du plan de données permet de maintenir la disponibilité alors que les plans de contrôle ne le font pas. Pour les mécanismes de reprise après sinistre et de basculement, utilisez les fonctions du plan de données pour assurer la meilleure fiabilité possible.

Pour Amazon EC2, utilisez des modèles de stabilité statique pour limiter les actions du plan de contrôle. Les actions du plan de contrôle incluent l'augmentation des ressources individuellement ou à l'aide de groupes Auto Scaling (ASG). Pour obtenir les niveaux de résilience les plus élevés,

allouez une capacité suffisante dans le cluster utilisé pour le basculement. Si ce seuil de capacité doit être limité, définissez des limitations sur l'ensemble du système de bout en bout afin de restreindre en toute sécurité le trafic total atteignant l'ensemble limité de ressources.

Pour des services comme Amazon DynamoDB, Amazon API Gateway, les équilibreurs de charge et AWS Lambda sans serveur, leur utilisation permet de tirer parti du plan de données. Cependant, la création de fonctions, d'équilibreurs de charge, de passerelles d'API ou de tables DynamoDB est une action du plan de contrôle qui doit être terminée avant la dégradation afin de préparer un événement et de répéter les actions de basculement. Pour Amazon RDS, les actions du plan de données permettent d'accéder aux données.

Pour plus d'informations sur les plans de données, les plans de contrôle et la manière dont AWS crée des services pour atteindre les objectifs de haute disponibilité, consultez le livre blanc sur la stabilité statique avec les zones de disponibilité.

Comprendre quelles opérations relèvent du plan de données et quelles opérations relèvent du plan de contrôle

Étapes d'implémentation

Pour chaque charge de travail qui doit être restaurée après un événement de dégradation, évaluez le runbook de basculement, la conception de la haute disponibilité, la conception de la réparation automatique ou le plan de restauration des ressources haute disponibilité. Identifiez chaque action qui pourrait être considérée comme une action du plan de contrôle.

Envisagez de remplacer l'action du plan de contrôle par une action de plan de données :

- Auto Scaling (plan de contrôle) par rapport aux ressources Amazon EC2 pré-dimensionnées (plan de données)
- Migration vers Lambda et ses méthodes de mise à l'échelle (plan de données) ou Amazon EC2 et ASG (plan de contrôle)
- Évaluez toutes les conceptions utilisant Kubernetes, ainsi que la nature des actions du plan de contrôle. L'ajout de pods est une action du plan de données dans Kubernetes. Les actions doivent se limiter à l'ajout de pods et non à l'ajout de nœuds. L'utilisation de nœuds surprovisionnés est la méthode préférée pour limiter les actions du plan de contrôle.

Envisagez d'autres approches qui permettent aux actions du plan de données d'affecter les mêmes mesures correctives.

- Modification d'enregistrement Route 53 (plan de contrôle) ou Route 53 ARC (plan de données)
- Surveillance de l'état Route 53 pour des mises à jour plus automatisées

Envisagez certains services dans une région secondaire, s'ils sont critiques, afin de permettre davantage d'actions du plan de contrôle et du plan de données dans une région non affectée.

- Amazon EC2 Auto Scaling ou Amazon EKS dans une région principale par rapport à Amazon EC2 Auto Scaling ou Amazon EKS dans une région secondaire et acheminement du trafic vers la région secondaire (action du plan de contrôle)
- Réalisez un réplica en lecture dans la région secondaire ou tentez la même action dans la région principale (action du plan de contrôle).

Ressources

Bonnes pratiques associées :

- Définition de la disponibilité
- REL11-BP01 Surveillance de tous les composants de la charge de travail pour détecter les défaillances

Documents connexes :

- Partenaire APN : partenaires pouvant vous aider à automatiser votre tolérance aux pannes
- AWS Marketplace : produits pouvant être utilisés pour la tolérance aux pannes
- L'Amazon Builders' Library : éviter la surcharge des systèmes distribués en plaçant sous contrôle le plus petit service
- API Amazon DynamoDB (plan de contrôle et plan de données)
- Exécutions AWS Lambda (réparties entre le plan de contrôle et le plan de données)
- Plan de données AWS Elemental MediaStore
- Création d'applications hautement résilientes à l'aide d'Amazon Route 53 Application Recovery Controller, partie 1 : pile dans une seule région
- Création d'applications hautement résilientes à l'aide d'Amazon Route 53 Application Recovery Controller, partie 2 : pile multirégion
- Création de mécanismes de reprise après sinistre à l'aide d'Amazon Route 53

- Qu'est-ce que Route 53 Application Recovery Controller ?
- Plan de contrôle et plan de données Kubernetes

Vidéos connexes :

- Back to Basics - Using Static Stability
- Building resilient multi-site workloads using AWS global services

Exemples connexes :

- Qu'est-ce qu'Amazon Route 53 Application Recovery Controller ?
- L'Amazon Builders' Library : éviter la surcharge des systèmes distribués en plaçant sous contrôle le plus petit service
- Création d'applications hautement résilientes à l'aide d'Amazon Route 53 Application Recovery Controller, partie 1 : pile dans une seule région
- Création d'applications hautement résilientes à l'aide d'Amazon Route 53 Application Recovery Controller, partie 2 : pile multirégion
- stabilité statique avec les zones de disponibilité

Outils associés :

- Amazon CloudWatch
- AWS X-Ray

REL11-BP05 Utiliser la stabilité statique pour éviter les comportements bimodaux

Les charges de travail doivent être statiquement stables et ne fonctionner que dans un seul mode normal. On parle de comportement bimodal lorsque la charge de travail présente un comportement différent en mode normal et en mode d'échec.

Par exemple, vous pouvez essayer de récupérer une défaillance de la zone de disponibilité en lançant de nouvelles instances dans une zone de disponibilité différente. Il peut en résulter une réponse bimodale lors d'un mode de défaillance. Pour éviter ce type de comportement, vous devez créer des charges de travail stables statiquement et qui fonctionnent dans un seul mode. Dans cet

exemple, ces instances auraient dû être provisionnées dans la deuxième zone de disponibilité avant la panne. Ce modèle de stabilité statique permet de vérifier que la charge de travail ne fonctionne que dans un seul mode.

Résultat souhaité : Les charges de travail ne présentent pas de comportement bimodal en mode normal et en mode d'échec.

Anti-modèles courants :

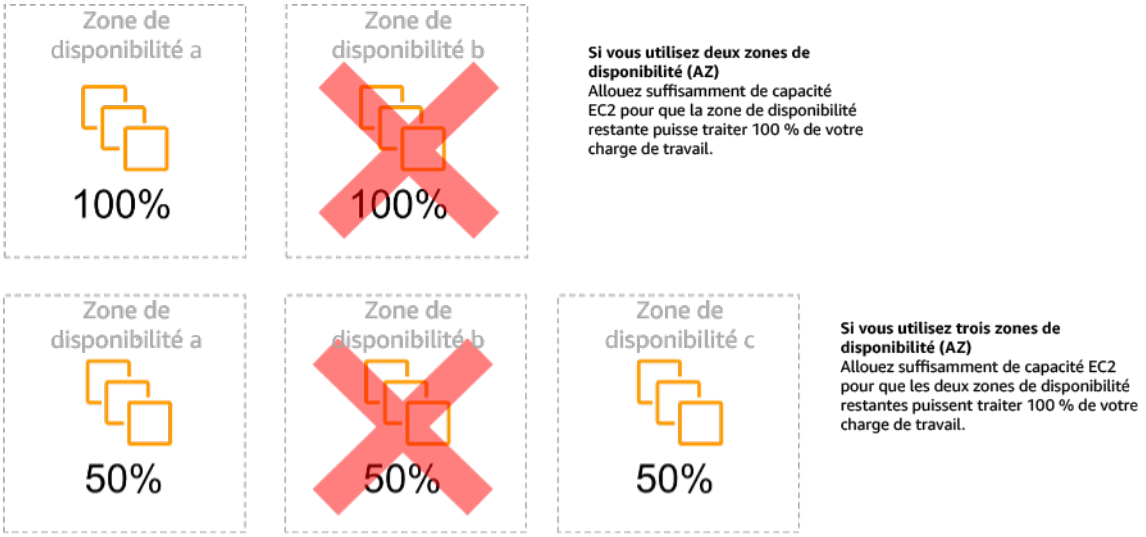
- Supposer que les ressources peuvent toujours être provisionnées quelle que soit l'étendue de la défaillance.
- Essayer d'acquérir dynamiquement des ressources lors d'une panne.
- Ne pas provisionner les ressources adéquates dans les zones ou les régions jusqu'à ce qu'une panne se produise.
- Envisager des modèles statiques et stables pour les ressources informatiques uniquement.

Avantages liés au respect de cette bonne pratique : Les charges de travail exécutées avec des modèles statiquement stables sont capables d'avoir des résultats prévisibles lors d'événements normaux et de défaillances.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyen

Directives d'implémentation

Un comportement bimodal survient lorsque votre charge de travail adopte un comportement différent en mode normal et en mode de défaillance (par exemple, en s'appuyant sur le lancement de nouvelles instances en cas de défaillance d'une zone de disponibilité). Exemple de comportement bimodal : les modèles Amazon EC2 stables provisionnent suffisamment d'instances dans chaque zone de disponibilité pour gérer la charge de travail si une zone de disponibilité était supprimée. L'état de Elastic Load Balancing ou de Amazon Route 53 effectuerait une vérification pour déplacer une charge des instances déficientes. Une fois le trafic déplacé, utilisez AWS Auto Scaling pour remplacer de manière asynchrone les instances de la zone défaillante et les lancer dans les zones saines. La stabilité statique du déploiement de calcul (par exemple, des conteneurs ou des instances EC2) garantit une fiabilité optimale.



Stabilité statique des instances EC2 dans les zones de disponibilité

Cela doit être comparé au coût de ce modèle et à la valeur commerciale du maintien de la charge de travail dans tous les cas de résilience. Il est moins coûteux de provisionner moins de capacité de calcul et de compter sur le lancement de nouvelles instances en cas de panne. Cependant, pour les pannes à grande échelle (comme une zone de disponibilité ou une panne régionale), cette approche se révèle moins efficace, car elle repose à la fois sur un plan opérationnel et sur la disponibilité de ressources suffisantes dans les zones ou les régions non affectées.

Votre solution doit également tenir compte de la fiabilité par rapport aux coûts nécessaires pour votre charge de travail. Les architectures de stabilité statique s'appliquent à différentes architectures, notamment aux instances de calcul réparties dans les zones de disponibilité, les modèles de réplicas en lecture de bases de données, les modèles de clusters Kubernetes (Amazon EKS) et les architectures de basculement multi-régions.

Il est également possible de mettre en œuvre un modèle plus stable sur le plan statique en utilisant davantage de ressources dans chaque zone. En ajoutant davantage de zones, vous réduisez la quantité de calcul supplémentaire nécessaire à la stabilité statique.

Autre exemple de comportement bimodal : un délai d'expiration du réseau peut amener un système à tenter d'actualiser l'état de configuration de l'ensemble du système. Cela ajouterait une charge inattendue à un autre composant et pourrait provoquer sa défaillance, entraînant d'autres conséquences inattendues. Cette boucle de rétroaction négative a un impact sur la disponibilité de votre charge de travail. Vous pourriez donc créer des systèmes stables statiquement et fonctionnant dans un seul mode. Un modèle statiquement stable consisterait à effectuer un travail constant et

à toujours actualiser l'état de la configuration selon une cadence fixe. Lorsqu'un appel échoue, la charge de travail utilise la valeur précédemment mise en cache et déclenche une alarme.

Un autre exemple de comportement bimodal consiste à autoriser les clients à contourner votre cache de charge de travail lorsque des défaillances se produisent. Cette solution peut sembler répondre aux besoins des clients, mais elle peut modifier considérablement les exigences de votre charge de travail et risque d'entraîner des échecs.

Évaluez les charges de travail critiques afin de déterminer celles qui nécessitent ce type de modèle de résilience. Pour celles qui sont jugées critiques, chaque composant de l'application doit être examiné. Voici quelques exemples de services nécessitant une évaluation de la stabilité statique :

- Calcul: Amazon EC2, EKS-EC2, ECS-EC2, EMR-EC2
- Bases de données: Amazon Redshift, Amazon RDS, Amazon Aurora
- Stockage: Amazon S3 (zone unique), Amazon EFS (supports), Amazon FSx (supports)
- Équilibreurs de charge : Selon certains modèles

Étapes d'implémentation

- Créez des systèmes stables statiquement et qui fonctionnent dans un seul mode. Dans ce cas, provisionnez suffisamment d'instances dans chaque zone de disponibilité ou région pour gérer la capacité de la charge de travail si une zone de disponibilité ou une région était supprimée. Plusieurs services peuvent être utilisés pour l'acheminement vers des ressources saines, par exemple :
  - Routage DNS entre régions
  - Routage multi-régions MRAP Amazon S3
  - AWS Global Accelerator
  - Amazon Route 53 Application Recovery Controller
- Configurez les réplicas en lecture de la base de données pour tenir compte de la perte d'une instance primaire unique ou d'un réplica en lecture. Si le trafic est desservi par des réplicas en lecture, la quantité dans chaque zone de disponibilité et chaque région doit correspondre au besoin global en cas de défaillance de la zone ou de la région.
- Configurez les données critiques dans un stockage Amazon S3 conçu pour être statiquement stable pour les données stockées en cas de défaillance d'une zone de disponibilité. Si la classe de stockage Amazon S3 One Zone-IA est utilisée, elle ne doit pas être considérée comme statiquement stable, car la perte de cette zone minimise l'accès aux données stockées.

- Les équilibreurs de charge sont parfois configurés de manière incorrecte ou sciemment pour desservir une zone de disponibilité spécifique. Dans ce cas, le modèle statiquement stable peut consister à répartir une charge de travail sur plusieurs zones de disponibilité dans le cadre d'un modèle plus complexe. Le modèle original peut être utilisé pour réduire le trafic interzone pour des raisons de sécurité, de latence ou de coût.

Ressources

Bonnes pratiques Well-Architected connexes :

- Définition de la disponibilité
- REL11-BP01 Surveillance de tous les composants de la charge de travail pour détecter les défaillances
- REL11-BP04 S'appuyer sur le plan de données et non sur le plan de contrôle pendant la récupération

Documents connexes :

- Minimiser les dépendances dans un plan de reprise après sinistre
- L'Amazon Builders' Library : stabilité statique avec les zones de disponibilité
- Fault Isolation Boundaries
- stabilité statique avec les zones de disponibilité
- RDS multi-zones
- Minimiser les dépendances dans un plan de reprise après sinistre
- Routage DNS entre régions
- Routage multi-régions MRAP Amazon S3
- AWS Global Accelerator
- Route 53 ARC
- Zone unique Amazon S3
- Équilibrage de charge entre zones

Vidéos connexes :

- Stabilité statique dans AWS : AWS re:Invent 2019 : présentation de la bibliothèque Amazon Builders' Library (DOP328)

Exemples connexes :

- L'Amazon Builders' Library : stabilité statique avec les zones de disponibilité

REL11-BP06 Envoyer des notifications lorsque des événements affectent la disponibilité

Des notifications sont envoyées en cas de détection de dépassement de seuils, même si l'événement à l'origine du problème a été automatiquement résolu.

La réparation automatisée permet à votre charge de travail d'être fiable. Cependant, elle peut également masquer les problèmes sous-jacents à résoudre. Implémentez une surveillance et des événements appropriés afin de pouvoir détecter les schémas de problèmes, y compris ceux résolus par la réparation automatique, afin de pouvoir résoudre les problèmes de cause racine.

Les systèmes résilients sont conçus de manière à ce que les événements de dégradation soient immédiatement communiqués aux équipes concernées. Ces notifications doivent être envoyées par un ou plusieurs canaux de communication.

Résultat souhaité : Des alertes sont immédiatement envoyées aux équipes chargées des opérations lorsque des seuils sont dépassés, tels que les taux d'erreur, la latence ou d'autres métriques d'indicateurs clés de performance (KPI) critiques, afin que ces problèmes soient résolus dès que possible et que l'impact sur les utilisateurs soit évité ou minimisé.

Anti-modèles courants :

- Envoyer un trop grand nombre d'alarmes.
- Envoyer des alarmes non exploitables.
- Régler les seuils d'alarme à un niveau trop élevé (sensibilité excessive) ou trop faible (sensibilité insuffisante).
- Ne pas envoyer d'alarmes pour les dépendances externes.
- Ne pas tenir compte des défaillances grises lors de la conception de la surveillance et des alarmes.
- Effectuer des réparations automatisées, mais ne pas notifier l'équipe appropriée que des réparations étaient nécessaires.

Avantages liés au respect de cette bonne pratique : Les notifications de reprise permettent aux équipes commerciales et chargées des opérations d'être informées des dégradations de service, de sorte qu'elles peuvent réagir immédiatement pour minimiser à la fois le temps moyen de détection (MTTD) et le temps moyen de réparation (MTTR). Les notifications d'événements de reprise vous permettent également de ne pas ignorer les problèmes qui se produisent peu fréquemment.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : moyen. L'absence de mise en œuvre de mécanismes appropriés de surveillance et de notification des événements peut entraîner l'incapacité à détecter des schémas de problèmes, y compris ceux traités par la réparation automatisée. Une équipe ne sera informée de la dégradation du système que lorsque les utilisateurs contacteront le service clientèle ou par hasard.

Directives d'implémentation

Lors de la définition d'une stratégie de surveillance, le déclenchement d'une alarme est un événement courant. Cet événement contiendra probablement un identifiant pour l'alarme, l'état de l'alarme (comme `IN ALARM` ou `OK`) et des détails sur ce qui l'a déclenchée. Dans de nombreux cas, un événement d'alarme doit être détecté et une notification par e-mail doit être envoyée. Voici un exemple d'action sur une alarme. La notification d'alarme est essentielle pour l'observabilité, car elle permet d'informer les bonnes personnes de l'existence d'un problème. Cependant, lorsque l'action sur les événements arrive à maturité dans votre solution d'observabilité, elle peut automatiquement remédier au problème sans nécessiter d'intervention humaine.

Une fois que les alarmes de suivi des KPI ont été établies, des alertes doivent être envoyées aux équipes concernées lorsque les seuils sont dépassés. Ces alertes peuvent également être utilisées pour déclencher des processus automatisés qui tenteront de remédier à la dégradation.

Pour une surveillance plus complexe des seuils, des alarmes composites doivent être envisagées. Les alarmes composites utilisent un certain nombre d'alarmes de surveillance des KPI pour créer une alerte basée sur la logique métier opérationnelle. Les alarmes CloudWatch peuvent être configurées pour envoyer des e-mails afin de consigner des incidents dans des systèmes tiers de suivi des incidents à l'aide de l'intégration d'Amazon SNS ou de Amazon EventBridge.

Étapes d'implémentation

Créez différents types d'alarmes en fonction des charges de travail surveillées, par exemple :

- Les alarmes d'application permettent de détecter si une partie de votre charge de travail ne fonctionne pas correctement.

- Les alarmes d'infrastructure indiquent à quel moment il convient de mettre les ressources à l'échelle. Le système peut afficher les alarmes sur des tableaux de bord, envoyer des alertes via Amazon SNS ou par e-mail et fonctionner avec Auto Scaling pour une mise à l'échelle verticale ou horizontale des ressources de la charge de travail.
- Simple Des alarmes statiques simples peuvent être créées pour surveiller le dépassement d'un seuil statique par une métrique pendant un nombre spécifié de périodes d'évaluation.
- Les alarmes composites peuvent prendre en compte des alarmes complexes provenant de sources multiples.
- Une fois l'alarme créée, créez les événements de notification appropriés. Vous pouvez directement appeler une API Amazon SNS pour envoyer des notifications et lier toute automatisation pour la remédiation ou la communication.
- Intégrez Amazon Health Aware pour permettre d'identifier les ressources AWS susceptibles de subir des dégradations. Pour les charges de travail essentielles à l'entreprise, cette solution permet d'accéder à des alertes proactives et en temps réel pour les services AWS.

Ressources

Bonnes pratiques Well-Architected connexes :

- Définition de la disponibilité

Documents connexes :

- Création d’une alarme CloudWatch basée sur un seuil statique
- Qu'est-ce qu'Amazon EventBridge ?
- Qu'est-ce qu'Amazon Simple Notification Service ?
- Publication des métriques personnalisées
- Utilisation des alarmes Amazon CloudWatch
- Amazon Health Aware (AHA)
- Configuration d'alarmes composites CloudWatch
- Les nouveautés en matière d'AWSobservabilité à re:Invent 2022

Outils associés :

- CloudWatch

- CloudWatch X-Ray

REL11-BP07 Concevoir votre produit pour atteindre les objectifs de disponibilité et les accords de niveau de service (SLA)

Concevez votre produit de manière à atteindre les objectifs de disponibilité et les accords de niveau de service (SLA). Si vous publiez ou convenez en privé d'objectifs de disponibilité ou d'accords de niveau de service, vérifiez que votre architecture et vos processus opérationnels sont conçus pour les prendre en charge.

Résultat souhaité : chaque application a un objectif défini pour la disponibilité et un accord de niveau de service pour les métriques de performance. Ces éléments peuvent être surveillés et maintenus afin d'atteindre les résultats opérationnels.

Anti-modèles courants :

- Concevoir et déployer des charges de travail sans fixer d'accords de niveau de service.
- Les métriques des SLA sont fixées à un niveau élevé sans justification ni exigences commerciales.
- Fixer des accords de niveau de service sans tenir compte des dépendances et des accords de niveau de service sous-jacents.
- Les conceptions d'applications sont créées sans tenir compte du modèle de responsabilité partagée pour la résilience.

Avantages liés au respect de cette bonne pratique : la conception d'applications basées sur des objectifs clés de résilience vous aide à atteindre les objectifs commerciaux et à répondre aux attentes des clients. Ces objectifs contribuent à orienter le processus de conception de l'application qui évalue les différentes technologies et envisage divers compromis.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

La conception des applications doit tenir compte d'un ensemble diversifié d'exigences découlant d'objectifs commerciaux, opérationnels et financiers. Dans le cadre des exigences opérationnelles, les charges de travail doivent avoir des objectifs spécifiques en matière de métriques de résilience afin qu'elles puissent être correctement surveillées et prises en charge. Les métriques de résilience ne doivent pas être définies ou déduites après le déploiement de la charge de travail. Elles doivent être définies pendant la phase de conception et aider à guider les diverses décisions et compromis.

- Chaque charge de travail doit disposer de son propre ensemble de métriques de résilience. Ces métriques peuvent être différentes de celles d'autres applications commerciales.
- La réduction des dépendances peut avoir un impact positif sur la disponibilité. Chaque charge de travail doit tenir compte de ses dépendances et de leurs accords de niveau de service. En général, sélectionnez les dépendances dont les objectifs de disponibilité sont égaux ou supérieurs à ceux de votre charge de travail.
- Envisagez des conceptions faiblement couplées afin que votre charge de travail puisse fonctionner correctement malgré l'altération des dépendances, lorsque cela est possible.
- Réduisez les dépendances du plan de contrôle, notamment lors de la reprise ou d'une dégradation. Évaluez les conceptions statiques stables pour les charges de travail critiques. Utilisez le partage des ressources pour augmenter la disponibilité de ces dépendances dans une charge de travail.
- L'observabilité et l'instrumentation sont essentielles pour respecter les accords de niveau de service en réduisant le temps moyen de détection (MTTD) et le temps moyen de réparation (MTTR).
- Des défaillances moins fréquentes (MTBF plus long), des temps de détection des défaillances plus courts (MTTD plus court) et des temps de réparation plus courts (MTTR plus court) sont les trois facteurs utilisés pour améliorer la disponibilité des systèmes distribués.
- L'établissement et le respect des métriques de résilience pour une charge de travail sont à la base de toute conception efficace. Ces conceptions doivent tenir compte des compromis entre la complexité de la conception, les dépendances des services, les performances, la mise à l'échelle et les coûts.

Étapes d'implémentation

- Examinez et documentez la conception de la charge de travail en tenant compte des questions suivantes :
  - Où les plans de contrôle sont-ils utilisés dans la charge de travail ?
  - Comment la charge de travail met-elle en œuvre la tolérance aux pannes ?
  - Quels sont les modèles de conception pour la mise à l'échelle, la scalabilité automatique, la redondance et les composants hautement disponibles ?
  - Quelles sont les exigences en matière de cohérence et de disponibilité des données ?
  - Y a-t-il des considérations relatives à l'économie des ressources ou à la stabilité statique des ressources ?
  - Quelles sont les dépendances des services ?

- Définissez les métriques SLA en fonction de l'architecture de la charge de travail tout en travaillant avec les parties prenantes. Tenez compte des SLA de toutes les dépendances utilisées par la charge de travail.
- Une fois l'objectif du SLA fixé, optimisez l'architecture pour le respecter.
- Une fois que la conception a été définie de manière à respecter l'accord de niveau de service, il faut mettre en œuvre les changements opérationnels, l'automatisation des processus et les runbooks qui visent également à réduire les délais d'attente et les temps de réponse.
- Une fois le déploiement effectué, surveillez et rendez compte de l'accord de niveau de service.

Ressources

Bonnes pratiques associées :

- REL03-BP01 Choisir comment segmenter votre charge de travail
- REL10-BP01 Déployer la charge de travail sur plusieurs emplacements
- REL11-BP01 Surveiller tous les composants de la charge de travail pour détecter les défaillances
- REL11-BP03 Automatiser la réparation sur toutes les couches
- REL12-BP05 Tester la résilience à l'aide de l'ingénierie du chaos
- REL13-BP01 Définir les objectifs de reprise pour les temps d'arrêt et les pertes de données
- Comprendre l'état de la charge de travail

Documents connexes :

- Availability with redundancy (Disponibilité avec redondance)
- Pilier Fiabilité : disponibilité
- Measuring availability (Mesurer la disponibilité)
- AWS Fault Isolation Boundaries (Limites d'isolement des pannes AWS)
- Modèle de responsabilité partagée pour la résilience
- stabilité statique avec les zones de disponibilité
- Accords de niveau de service (SLA) AWS
- Guidance for Cell-based Architecture on AWS (Guide de l'architecture cellulaire sur AWS)
- Infrastructure AWS

- Advanced Multi-AZ Resiliance Patterns whitepaper (Livre blanc sur les modèles de résilience multi-AZ avancés)

Services associés :

- Amazon CloudWatch
- AWS Config
- AWS Trusted Advisor

Test de la fiabilité

Une fois que vous avez conçu votre charge de travail pour qu'elle soit résiliente aux sollicitations de la production, les tests sont le seul moyen de s'assurer qu'elle fonctionne comme prévu et d'obtenir la résilience voulue.

Exécutez un test pour vérifier que votre charge de travail répond aux exigences fonctionnelles et non fonctionnelles, car les bogues ou les goulots d'étranglement des performances peuvent avoir un impact sur sa fiabilité. Testez la résilience de votre charge de travail pour vous aider à détecter les bogues latents présents uniquement dans la production. Pratiquez régulièrement ces tests.

Bonnes pratiques

- REL12-BP01 Utiliser des playbooks pour enquêter sur les causes des défaillances
- REL12-BP02 Effectuer une analyse post-incident
- REL12-BP03 Tester les exigences fonctionnelles
- REL12-BP04 Tester les exigences de mise à l'échelle et de performance
- REL12-BP05 Tester la résilience à l'aide de l'ingénierie du chaos
- REL12-BP06 Organiser régulièrement des tests de simulation de panne

REL12-BP01 Utiliser des playbooks pour enquêter sur les causes des défaillances

Consignez le processus d'enquête dans des playbooks afin de faciliter l'application de réponses cohérentes et rapides face aux scénarios de défaillance qui ne sont pas bien compris. Les playbooks sont les étapes prédéfinies suivies pour identifier les facteurs adjuvants à un scénario défaillance. Les

résultats des étapes du processus sont utilisés pour déterminer les prochaines mesures à prendre jusqu'à ce que la question soit identifiée ou remontée.

Le playbook est une planification proactive que vous devez appliquer afin de pouvoir prendre efficacement des mesures réactives. Lorsque des scénarios de défaillance ne figurant pas dans le playbook sont rencontrés en production, commencez par résoudre le problème (éteindre l'incendie). Procédez ensuite à une rétrospective en examinant les étapes suivies pour résoudre le problème et utilisez-les pour ajouter une nouvelle entrée dans le playbook.

Notez que les playbooks sont utilisés en réponse à des incidents spécifiques, tandis que les runbooks le sont pour obtenir des résultats spécifiques. En règle générale, les runbooks sont employés pour les activités de routine et les playbooks pour répondre à des événements non réguliers.

Anti-modèles courants :

- Planification du déploiement d'une charge de travail sans connaître les processus permettant de diagnostiquer les problèmes ou de répondre aux incidents.
- Décisions imprévues sur les systèmes à partir desquels peut se faire la collecte des journaux et métriques lors de l'examen d'un événement.
- Non-conservation des métriques et événements pendant suffisamment longtemps pour pouvoir récupérer les données.

Avantages liés au respect de cette bonne pratique : La capture des playbooks garantit que les processus peuvent être suivis de manière cohérente. La codification de vos playbooks limite l'introduction d'erreurs à partir de l'activité manuelle. L'automatisation des playbooks accélère le temps de réponse à un événement en évitant aux membres de l'équipe d'intervenir ou en leur fournissant des informations supplémentaires lorsque leur intervention commence.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

- Utilisez des playbooks pour identifier les problèmes. Les playbooks sont des processus documentés pour enquêter sur les problèmes. Mettez en œuvre des réponses cohérentes et rapides aux échecs en documentant les processus dans des playbooks. Les playbooks doivent contenir les informations et les instructions nécessaires pour permettre à une personne compétente de recueillir les informations pertinentes, identifier les causes potentielles de défaillance, isoler les pannes et déterminer les facteurs adjuvants (c'est-à-dire effectuer une analyse post-incident).

- Mettez en œuvre les playbooks en tant que code Effectuez vos opérations en tant que code scriptant vos playbooks afin d'en assurer la cohérence et de limiter les erreurs causées par les processus manuels. Les playbooks peuvent être composés de plusieurs scripts représentant les différentes étapes qui pourraient être nécessaires pour identifier les facteurs contribuant à un problème. Les activités Runbook peuvent être déclenchées ou effectuées dans le cadre d'activités playbook, ou peuvent demander l'exécution d'un playbook en réponse à des événements identifiés.
  - Automatisez vos playbooks opérationnels avec AWS Systems Manager
  - AWS Systems Manager Run Command
  - AWS Systems Manager Automation
  - Qu'est-ce qu'AWS Lambda ?
  - Qu'est-ce qu'Amazon EventBridge ?
  - Utilisation des alarmes Amazon CloudWatch

Ressources

Documents connexes :

- AWS Systems Manager Automation
- AWS Systems Manager Run Command
- Automatisez vos playbooks opérationnels avec AWS Systems Manager
- Utilisation des alarmes Amazon CloudWatch
- Utilisation de scripts Canary (Amazon CloudWatch Synthetics)
- Qu'est-ce qu'Amazon EventBridge ?
- Qu'est-ce qu'AWS Lambda ?

Exemples connexes :

- Automatisation des opérations avec les playbooks et les runbooks

REL12-BP02 Effectuer une analyse post-incident

Passez en revue les événements ayant un impact sur le client et identifiez les facteurs adjuvants et les mesures préventives. Utilisez ces informations pour prendre des mesures d'atténuation afin de

limiter ou de remédier aux problèmes. Développez des procédures pour fournir des réponses rapides et efficaces. Publiez, le cas échéant, les facteurs adjuvants et les mesures correctives adaptées au public ciblé. Vous devez disposer d'une méthode pour communiquer ces causes à d'autres si nécessaire.

Évaluez pourquoi les tests existants n'ont pas permis de résoudre le problème. Ajoutez des tests pour ce cas si aucun test correspondant n'existe.

Résultat souhaité : vos équipes ont adopté une approche cohérente et concertée pour gérer l'analyse post-incident. L'un des mécanismes est le processus de correction des erreurs (COE). Celui-ci aide vos équipes à identifier, comprendre et traiter les causes profondes des incidents, tout en mettant en place des mécanismes et des barrières de protection pour limiter la probabilité qu'un incident se reproduise.

Anti-modèles courants :

- Trouver des facteurs adjuvants sans pour autant continuer à chercher plus en profondeur d'autres problèmes et approches potentiels pour atténuer les risques.
- Se contenter d'identifier les causes des erreurs humaines et ne pas proposer de formation ou d'automatisation susceptibles d'empêcher les erreurs humaines.
- Se concentrer sur les reproches plutôt que sur la compréhension des causes profondes, ce qui crée une culture de la peur et empêche de communiquer ouvertement
- Absence de partage d'informations, qui entrave la circulation des résultats de l'analyse de l'incident et empêche les autres de bénéficier des enseignements tirés
- Absence de mécanisme permettant de capturer les connaissances institutionnelles, ce qui engendre une perte d'informations précieuses en ne conservant pas les enseignements tirés sous la forme de bonnes pratiques actualisées, et entraîne la répétition d'incidents ayant des causes profondes identiques ou similaires

Avantages liés au respect de cette bonne pratique : la réalisation d'une analyse post-incident et le partage des résultats permettent à d'autres charges de travail d'atténuer les risques si elles ont les mêmes facteurs contributifs. Cela permet aussi de mettre en œuvre la mesure d'atténuation ou de récupération automatisée avant qu'un incident ne se produise.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

Une bonne analyse post-incident permet de proposer des solutions courantes pour les problèmes avec des modèles d'architecture utilisés dans d'autres compartiments de vos systèmes.

La documentation et la résolution des problèmes sont l'une des pierres angulaires du processus COE. Il est recommandé de définir une méthode normalisée pour documenter les causes profondes et de veiller à ce qu'elles soient examinées et traitées. Attribuez clairement la responsabilité du processus d'analyse post-incident. Désignez une équipe ou une personne chargée de superviser les enquêtes et le suivi de l'incident.

Encouragez une culture axée sur l'apprentissage et l'amélioration plutôt que sur les reproches. Insistez sur le fait que l'objectif est de prévenir de futurs incidents, et non de pénaliser des individus.

Élaborez des procédures bien définies pour mener les analyses post-incident. Ces procédures doivent décrire les étapes à suivre, les informations à collecter et les principales questions à aborder lors de l'analyse. Enquêtez en profondeur sur les incidents, en allant au-delà des causes immédiates afin d'identifier les causes profondes et les facteurs contributifs. Utilisez des techniques telles que les cinq pourquoi pour approfondir les problèmes sous-jacents.

Tenez un répertoire des enseignements tirés des analyses des incidents. Ces connaissances institutionnelles peuvent servir de référence pour les incidents futurs et les efforts de prévention. Partagez les conclusions et les réflexions tirées des analyses post-incident, et envisagez d'organiser des réunions de synthèse post-incident ouvertes à tous pour discuter des enseignements tirés.

Étapes d'implémentation

- Veillez à ce que l'analyse post-incident soit exempte de tout reproche. Cela permet aux personnes impliquées dans l'incident de faire preuve d'objectivité quant aux actions correctives proposées, et de promouvoir une auto-évaluation et une collaboration honnêtes entre les équipes.
- Définissez une méthode standardisée pour documenter les problèmes critiques. Voici un exemple de structure :
  - Que s'est-il passé ?
  - Quel a été l'impact sur vos clients et votre activité ?
  - Quelle était la cause profonde ?
  - Quelles sont les données à votre disposition pour étayer votre raisonnement ?
    - Par exemple, des métriques et des graphiques.

    - Quelles ont été les principales répercussions, notamment en termes de sécurité ?
        - Lors de la conception des charges de travail, vous faites un compromis entre les piliers en fonction de votre activité. Ces décisions métier peuvent vous aider à gérer vos priorités techniques. Vous pouvez opter pour l'optimisation afin de réduire les coûts au détriment de la fiabilité dans les environnements de développement ou, pour les solutions stratégiques, vous pouvez optimiser la fiabilité pour des coûts plus importants. La sécurité est toujours une priorité, car vous devez protéger vos clients.
    - Quelles leçons avez-vous apprises ?
    - Quelles mesures correctives allez-vous prendre ?
        - Éléments d'action
        - Articles connexes
- Élaborez des procédures bien définies pour mener les analyses post-incident.
- Mettez en place un processus standardisé de signalement des incidents. Documentez tous les incidents de manière exhaustive, y compris le rapport d'incident initial, les journaux, les communications et les mesures prises pendant l'incident.
- N'oubliez pas qu'un incident n'est pas forcément une panne. Il peut s'agir d'un accident évité de justesse ou d'un système qui fonctionne de manière inattendue tout en remplissant sa fonction.
- Améliorez sans cesse votre processus d'analyse post-incident en fonction des retours et des enseignements tirés.
- Capturez les principales conclusions dans un système de gestion des connaissances et examinez les modèles qui devraient être ajoutés aux guides du développeur ou aux listes de contrôle de prédéploiement.

Ressources

Documents connexes :

- Pourquoi mettre en place la correction des erreurs (COE)

Vidéos connexes :

- Amazon’s approach to failing successfully
- AWS re:Invent 2021 - Amazon Builders’ Library: Operational Excellence at Amazon

REL12-BP03 Tester les exigences fonctionnelles

Utilisez des techniques telles que les tests unitaires et les tests d'intégration qui valident les fonctionnalités requises.

Vous obtenez les meilleurs résultats lorsque ces tests sont exécutés automatiquement dans le cadre d'actions de génération et de déploiement. Par exemple, grâce à AWS CodePipeline, les développeurs valident les modifications apportées à un référentiel source dans lequel CodePipeline détecte automatiquement les modifications. Ces modifications sont générées et des tests sont exécutés. Une fois les tests terminés, le code généré est déployé sur des serveurs intermédiaires à des fins de test. Depuis le serveur intermédiaire, CodePipeline exécute d'autres tests, tels que des tests d'intégration ou de chargement. Une fois ces tests terminés avec succès, CodePipeline déploie le code testé et approuvé sur les instances de production.

De plus, l'expérience montre que les tests de transactions synthétiques (également appelés tests canaryà ne pas confondre avec les déploiements canary) qui peuvent exécuter et simuler le comportement des clients font partie des processus de test les plus importants. Exécutez ces tests en permanence sur vos points de terminaison de charge de travail à partir de divers emplacements distants. Amazon CloudWatch Synthetics vous permet de créer des scripts Canari pour surveiller vos points de terminaison et vos API.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

- Testez les exigences fonctionnelles. Il s'agit, entre autres, des tests unitaires et des tests d'intégration qui valident les fonctionnalités requises.
  - Utilisez CodePipeline avec AWS CodeBuild pour tester le code et exécuter des générations
  - AWS CodePipeline ajoute la prise en charge des tests unitaires et des tests d'intégration personnalisés avec AWS CodeBuild
  - Livraison et intégration continues (CI/CD)
  - Utilisation de scripts Canary (Amazon CloudWatch Synthetics)
  - Automatisation des tests logiciels

Ressources

Documents connexes :

- Partenaire APN : partenaires pouvant faciliter l'implémentation d'un pipeline d'intégration continue
- AWS CodePipeline ajoute la prise en charge des tests unitaires et des tests d'intégration personnalisés avec AWS CodeBuild
- AWS Marketplace : produits pouvant être utilisés pour une intégration continue
- Livraison et intégration continues (CI/CD)
- Automatisation des tests logiciels
- Utilisez CodePipeline avec AWS CodeBuild pour tester le code et exécuter des générations
- Utilisation de scripts Canary (Amazon CloudWatch Synthetics)

REL12-BP04 Tester les exigences de mise à l'échelle et de performance

Utilisez des techniques telles que les tests de charge pour valider que la charge de travail répond aux exigences de mise à l'échelle et de performances.

Dans le cloud, vous pouvez créer un environnement de test à la demande à l'échelle de la production pour votre charge de travail. Si vous exécutez ces tests sur une infrastructure réduite, vous devez mettre vos résultats observés à l'échelle en fonction de ce que vous pensez qu'il se produira en production. Les tests de charge et de performances peuvent également être réalisés en production si vous veillez à ne pas affecter les utilisateurs réels et à baliser vos données de test afin qu'elles ne correspondent pas aux données utilisateur réelles et ne corrompent pas les statistiques d'utilisation ni les rapports de production.

Utilisez les tests pour vous assurer que vos ressources de base, vos paramètres de mise à l'échelle, vos quotas de service et votre conception de la résilience fonctionnent comme prévu sous charge.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

- Testez les exigences de mise à l'échelle et de performance. Effectuez un test de charge pour vérifier que la charge de travail répond aux exigences de mise à l'échelle et de performance.
  - Test de charge distribuée sur AWS : simulation de milliers d'utilisateurs connectés
  - Apache JMeter
    - Déployez votre application dans un environnement identique à votre environnement de production et effectuez un test de charge.

- Utilisez les concepts d'Infrastructure as Code pour créer un environnement aussi similaire que possible à votre environnement de production.

Ressources

Documents connexes :

- Test de charge distribuée sur AWS : simulation de milliers d'utilisateurs connectés
- Apache JMeter

REL12-BP05 Tester la résilience à l'aide de l'ingénierie du chaos

Exécutez des expériences de chaos dans des environnements dont les conditions se rapprochent autant que possible de la production pour comprendre comment nos systèmes réagissent à des conditions défavorables.

Résultat souhaité :

La résilience de la charge de travail est régulièrement vérifiée en appliquant l'ingénierie du chaos sous la forme d'expériences d'injection de défaillances ou de charge inattendue, en plus des tests de résilience qui confirment le comportement attendu connu de votre charge de travail lors d'un événement. Associez l'ingénierie du chaos aux tests de résilience pour avoir l'assurance que votre charge de travail peut résister en cas de défaillance des composants et récupérer suite à des perturbations inattendues avec peu ou pas d'impact.

Anti-modèles courants :

- Conception à des fins de résilience, mais pas de vérification du fonctionnement de la charge de travail dans son ensemble en cas de défaillances.
- Pas d'expériences dans des conditions concrètes et pour la charge prévue.
- Pas de traitement de vos expériences en tant que code ou de maintenance de vos expériences tout au long du cycle de développement.
- Pas d'exécution d'expériences de chaos dans le cadre de votre pipeline CI/CD, ainsi qu'en dehors des déploiements.
- Pas d'utilisation des analyses passées post-incident pour déterminer les défaillances à tester.

Avantages liés au respect de cette bonne pratique : L'injection de défaillances pour vérifier la résilience de votre charge de travail vous permet d'avoir l'assurance que les procédures de récupération de votre conception résiliente fonctionneront en cas de défaillances réelles.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyen

Directives d'implémentation

L'ingénierie du chaos offre la possibilité à vos équipes d'injecter en continu des perturbations concrètes (simulations) de manière contrôlée au niveau du fournisseur de services, de l'infrastructure, de la charge de travail et des composants, avec peu ou pas d'impact pour vos clients. Ainsi, vos équipes tirent les leçons de ces défaillances et observent, mesurent et améliorent la résilience de vos charges de travail, tout en confirmant que les alertes se déclenchent et que les équipes sont informées en cas d'événement.

Une pratique de l'ingénierie du chaos en continu peut mettre en évidence des défaillances dans vos charges de travail qui, si elles ne sont pas résolues, peuvent impacter de manière négative la disponibilité et le fonctionnement.

> **Note**
>
> L'ingénierie du chaos est la discipline d'expérimentation d'un système. Elle permet de s'assurer de la capacité du système à résister à des conditions de production difficiles. – Principes de l'ingénierie du chaos

Si un système est capable de résister à ces perturbations, l'expérience de chaos doit être maintenue en tant que test de régression automatisé. De cette façon, les expériences de chaos doivent être réalisées dans le cadre de votre cycle de développement des systèmes et de votre pipeline CI/CD.

Pour veiller à ce que votre charge de travail résiste en cas de défaillance des composants, injectez des événements concrets dans le cadre de vos expériences. Par exemple, expérimentez une perte des instances Amazon EC2 ou un basculement de l'instance de base de données Amazon RDS principale, puis vérifiez que votre charge de travail n'est pas impactée (ou très peu). Utilisez plusieurs défaillances des composants pour simuler des événements capables de causer une perturbation dans une zone de disponibilité.

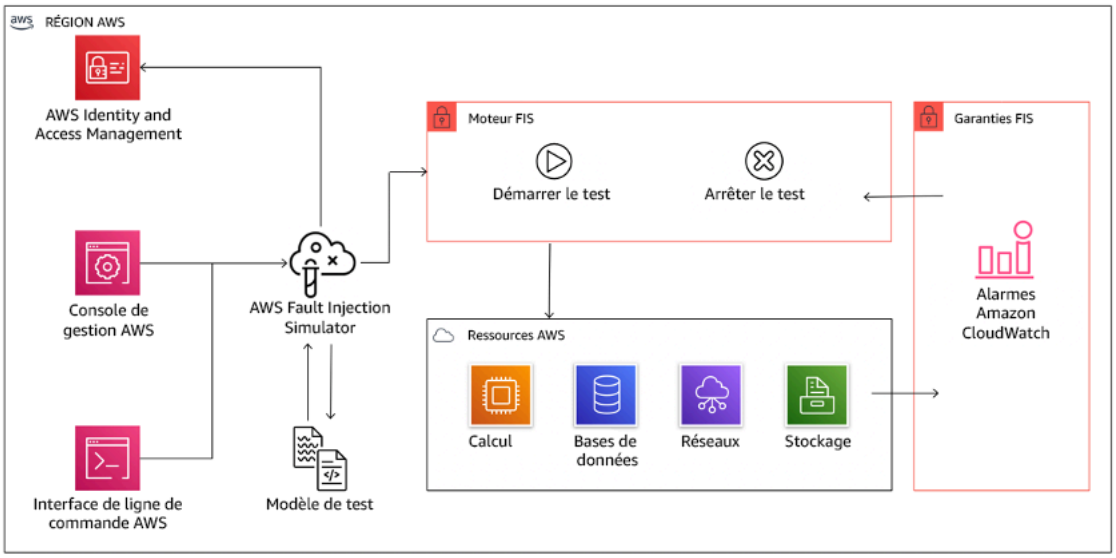
Pour les défaillances de niveau application (telles que les plantages), commencez par des tests de stress comme l'épuisement de la mémoire et du processeur.

Afin de valider les solutions de secours ou les mécanismes de basculement pour les dépendances externes dues aux pannes réseau intermittentes, vos composants doivent simuler un tel événement en bloquant l'accès aux fournisseurs tiers pendant une durée spécifiée pouvant aller de quelques secondes à plusieurs heures.

D'autres modes de dégradation peuvent entraîner des fonctionnalités limitées et ralentir les réponses, ce qui se traduit par une perturbation de vos services. Généralement, cette dégradation résulte d'une latence accrue sur les services critiques et d'une communication réseau peu fiable (perte de paquets). Les expériences avec ces défaillances, dont les effets de mise en réseau tels que la latence, les messages supprimés et les défaillances DNS, peuvent inclure l'incapacité de résoudre un nom, d'atteindre un service DNS ou de se connecter aux services dépendants.

Outils de l'ingénierie du chaos :

AWS Fault Injection Service (AWS FIS) est un service entièrement géré permettant l'exécution d'expériences d'injection de défaillances qui peuvent être utilisées dans le cadre de votre pipeline CD, ou en dehors du pipeline. AWS FIS s'impose donc comme un choix judicieux lors des tests de simulation de pannes. Il prend en charge de manière simultanée l'injection de défaillances sur différents types de ressources dont Amazon EC2, Amazon Elastic Container Service (Amazon ECS), Amazon Elastic Kubernetes Service (Amazon EKS) et Amazon RDS. Ces défaillances incluent l'arrêt des ressources, les basculements forcés, le stress du processeur ou de la mémoire, la limitation, la latence et la perte de paquets. Comme il est intégré aux alarmes Amazon CloudWatch, vous pouvez définir des conditions d'arrêt comme barrières de protection pour annuler une expérience si elle provoque un impact inattendu.



AWS Fault Injection Service s'intègre aux ressources AWS pour vous permettre d'exécuter des expériences d'injection de défaillances pour vos charges de travail.

Il existe également plusieurs options tierces pour les expériences d'injection de défaillances. Il existe notamment des outils open source tels que Chaos Toolkit, Chaos Meshet Litmus Chaos, ainsi que des options commerciales comme Gremlin. Pour élargir le champ des défaillances pouvant être injectées sur AWS, AWS FIS prend désormais en charge Chaos Mesh et Litmus Chaos, ce qui vous permet de coordonner les flux de travail d'injection des défaillances entre plusieurs outils. Par exemple, vous pouvez exécuter un test de stress sur un processeur de pod à l'aide des défaillances Chaos Mesh ou Litmus, tout en arrêtant un pourcentage de nœuds de cluster sélectionné de façon aléatoire grâce aux actions des défaillances AWS FIS.

Étapes d'implémentation

- Déterminer les défaillances à utiliser pour les expériences.

  Évaluez la conception de votre charge de travail à des fins de résilience. De telles conceptions (créées à l'aide des bonnes pratiques de Le cadre AWS Well-Architected) tiennent compte des risques en se basant sur des dépendances critiques, des événements passés, des erreurs connues et des exigences de conformité. Répertoriez chaque élément de la conception destiné à maintenir la résilience et les défaillances qu'il entend réduire. Pour plus d'informations sur la création de telles listes, consultez le livre blanc Examens de disponibilité opérationnelle qui vous guidera dans la création d'un processus capable de prévenir la répétition des incidents précédents. Le processus de Failure Modes and Effects Analysis (FMEA) ou d'analyse des modes de défaillance et de leurs effets vous propose un framework pour réaliser une analyse de niveau composant des défaillances et de leur impact sur votre charge de travail. Le processus FMEA est décrit plus en détail par Adrian Cockcroft dans l'article Failure Modes and Continuous Resilience (modes de défaillance et résilience continue).

- Attribuer une priorité à chaque défaillance.

  Commencez par définir une classification grossière telle que élevée, moyenne et basse. Pour évaluer les priorités, tenez compte de la fréquence de la défaillance et de son impact sur la charge de travail dans son ensemble.

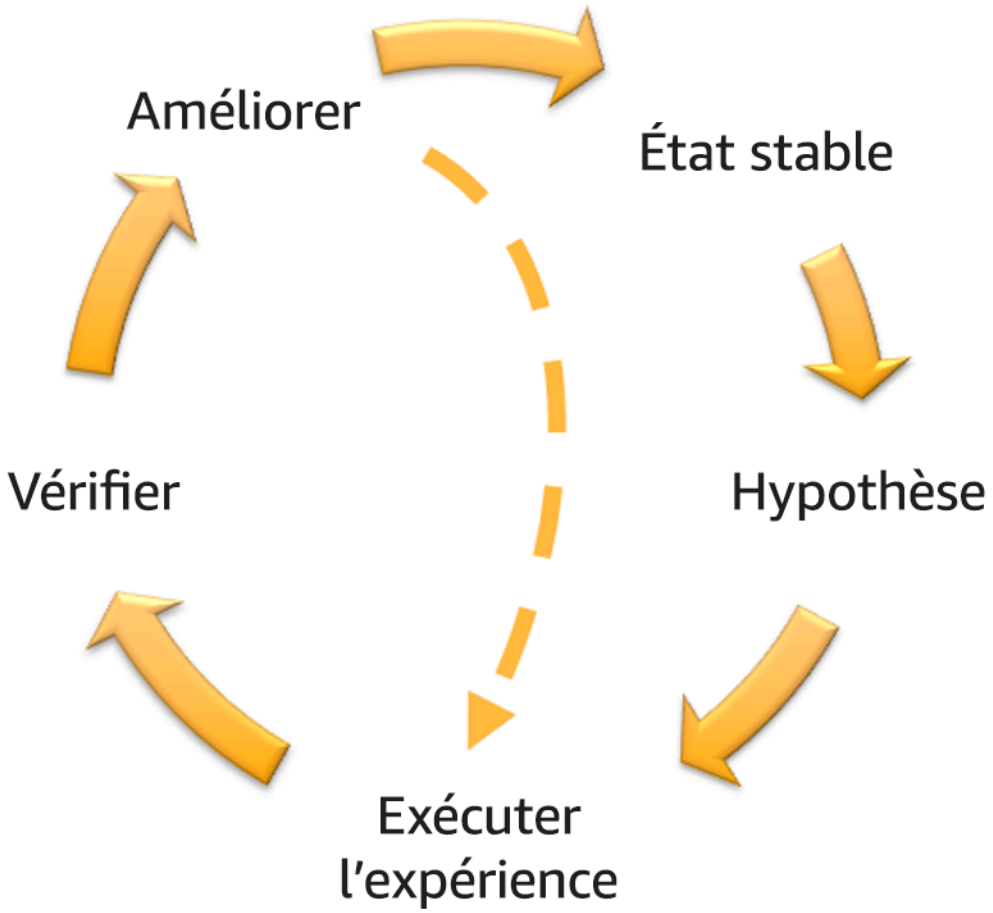
  Lors de la prise en compte de la fréquence d'une défaillance donnée, analysez les données passées de cette charge de travail, le cas échéant. Si aucune donnée passée n'est disponible, utilisez les données des autres charges de travail s'exécutant dans un environnement semblable.

Lors de la prise en compte de l'impact d'une défaillance donnée, souvenez-vous qu'en général plus le champ de la défaillance est large, plus grand est l'impact. Tenez compte également de la conception de la charge de travail et de son objectif. Par exemple, la capacité à accéder aux magasins de données sources est essentielle pour une charge de travail effectuant des transformations et de l'analyse de données. Dans ce cas, vous donnerez la priorité aux expériences liées aux défaillances d'accès ainsi qu'aux accès limités et à l'insertion de la latence.

Les analyses post-incident constituent une excellente source de données pour comprendre à la fois la fréquence et l'impact des modes de défaillance.

Utilisez la priorité attribuée pour déterminer les défaillances à expérimenter en premier lieu, puis l'ordre dans lequel développer de nouvelles expériences d'injection de défaillances.

- Suivre le volant d'inertie de l'ingénierie du chaos et de la résilience continue pour chaque expérience réalisée.



Volant d'inertie de l'ingénierie du chaos et de la résilience continue réalisé grâce à la méthode scientifique d'Adrian Hornsby.

- Définir l'état stable comme le résultat mesurable d'une charge de travail qui indique un comportement normal.

  Votre charge de travail présente un état stable si elle fonctionne de manière fiable et comme prévu. Par conséquent, confirmez que charge de travail est saine avant de définir un état stable. L'état stable ne signifie pas forcément sans impact pour la charge de travail en cas de défaillance, car un certain pourcentage des défaillances n'excède pas des limites supportables. L'état stable constitue le repère que vous observerez pendant l'expérience, qui mettra en évidence des anomalies si votre hypothèse formulée dans l'étape suivante ne donne pas les résultats escomptés.

  Par exemple, un état stable d'un système de paiements peut être défini comme le traitement de 300 TPS avec un taux de réussite de 99 % et un temps de transmission aller-retour de 500 ms.

- Formuler une hypothèse sur la façon dont la charge de travail réagira à la défaillance.

  Une bonne hypothèse repose sur la façon dont la charge de travail est destinée à réduire la défaillance pour maintenir l'état stable. L'hypothèse indique que vu qu'il s'agit d'une défaillance d'un type particulier, le système ou la charge de travail maintiendra un état stable, car la charge de travail a été conçue avec une atténuation des risques spécifique. Le type particulier de défaillance et d'atténuation des risques doit être spécifié dans l'hypothèse.

  Le modèle suivant peut être utilisé pour l'hypothèse (mais une autre formulation est aussi acceptable) :

  > **Note**
  >
  > Si *`défaillance spécifique`* se produit, la charge de travail *`nom de la charge de travail`* va *`décrire les contrôles pour atténuer les risques`* afin de limiter *`impact métier ou technique`*.

  Par exemple :

  - Si 20 % des nœuds du node-group Amazon EKS sont supprimés, l'API Transaction Create API continue de répondre au 99e centile des demandes en moins de 100 ms (état stable). Les nœuds Amazon EKS seront opérationnels dans les cinq minutes, et les pods seront planifiés

et traiteront le trafic huit minutes après le début de l'expérience. Les alertes se déclencheront sous trois minutes.

- En cas de défaillance d'une seule instance Amazon EC2, la surveillance de l'état Elastic Load Balancing du système de commandes permet à Elastic Load Balancing d'envoyer uniquement des demandes aux instances saines restantes, tandis qu'Amazon EC2 Auto Scaling remplace l'instance en échec, tout en maintenant une augmentation des erreurs (5xx) côté serveur (état stable) inférieure à 0,01 %.
- Si l'instance de base de données Amazon RDS principale échoue, la charge de travail de collecte des données Chaîne d'approvisionnement basculera et se connectera à l'instance de base de données Amazon RDS de secours pour maintenir les erreurs de lecture ou d'écriture de base de données (état stable) inférieures à 1 minute.

- Exécuter l'expérience en injectant la défaillance.

  Une expérience doit par défaut être sécurisée et tolérée par la charge de travail. Si vous savez que la charge de travail va échouer, n'exécutez pas l'expérience. L'ingénierie du chaos doit être utilisée pour rechercher les risques connus ou inconnus. Les risques connus sont les choses dont vous êtes conscient mais que vous ne comprenez pas bien, et les risques inconnus sont les choses dont vous n'êtes pas conscient ou que vous ne comprenez pas bien. Exécuter une expérience sur une charge de travail que vous savez défaillante ne vous apportera rien de plus. Votre expérience doit être soigneusement préparée, disposer d'un champ d'impact défini et fournir un mécanisme de protection pouvant être appliqué en cas de perturbations inattendues. Si votre vérification préalable indique que votre charge de travail doit résister à l'expérience, exécutez cette dernière. Il existe plusieurs moyens d'injecter les défaillances. Pour les charges de travail sur AWS, AWS FIS propose de nombreuses simulations de défaillances prédéfinies appelées des mesures. Vous pouvez également définir des actions personnalisées qui s'exécutent dans AWS FIS à l'aide des documents AWS Systems Manager.

  Nous déconseillons l'utilisation de scripts personnalisés pour les expériences de chaos, sauf si ces derniers sont capables de comprendre l'état actuel de la charge de travail, d'émettre des journaux, de fournir des mécanismes de protection pour annuler une expérience et des conditions d'arrêt dans la mesure du possible.

  Un framework ou des outils efficaces capables de prendre en charge l'ingénierie du chaos doivent suivre l'état actuel d'une expérience, émettre des journaux et fournir des mécanismes de protection pour prendre en charge l'exécution contrôlée d'une expérience. Commencez par un service établi comme AWS FIS qui vous permet d'exécuter des expériences avec un champ clairement défini et des mécanismes de sécurité capables de protéger l'expérience en cas de

perturbations inattendues. Pour découvrir plusieurs expériences utilisant AWS FIS, consultez également l' atelier Applications résilientes et Well-Architected avec l'ingénierie du chaos. Par ailleurs, AWS Resilience Hub analysera votre charge de travail et créera des expériences que vous pourrez choisir d'implémenter et d'exécuter dans AWS FIS.

> **Note**
>
> Pour chaque expérience, vous devez bien comprendre le champ et son impact. Nous recommandons que les défaillances soient d'abord simulées sur un environnement hors production avant d'être exécutées en production.

Les expériences doivent s'exécuter en production sous une charge concrète à l'aide des déploiements canary qui mettent en place des déploiements de système de contrôles et d'expériences, sous réserve de faisabilité. L'exécution d'expériences pendant les heures creuses est une bonne pratique pour réduire l'impact potentiel de la première expérience en production. De plus, si l'utilisation du trafic client réel s'avère trop risquée, vous pouvez exécuter des expériences à l'aide du trafic synthétique sur l'infrastructure de production pour des déploiements de système de contrôles et d'expériences. Lorsqu'une exécution en production n'est pas possible, exécutez les expériences dans des environnements de pré-production aussi proches que possible de la production.

Vous devez définir des barrières de protection pour veiller à ce que l'expérience n'impacte pas le trafic de la production ou d'autres systèmes au-delà des limites acceptables. Définissez des conditions d'arrêt pour stopper une expérience si elle atteint le seuil d'une métrique de barrière protection défini par vos soins. Ces conditions doivent inclure les métriques de l'état stable de la charge de travail, ainsi que celles sur les composants dans lesquels vous injectez la défaillance. A surveillance synthétique (également appelée un utilisateur canary) est une métrique que vous devez généralement inclure en tant que proxy utilisateur. Les conditions d'arrêt pour AWS FIS sont prises en charge dans le cadre d'un modèle de test, autorisant jusqu'à cinq conditions d'arrêt par modèle.

L'un des principes de l'ingénierie du chaos est de minimiser le champ de l'expérience et son impact :

Bien qu'un impact négatif à court terme soit autorisé, l'ingénieur du chaos a la responsabilité et l'obligation de minimiser et de maîtriser les conséquences des expériences.

Pour vérifier le champ et l'impact potentiel, vous pouvez dans un premier temps exécuter l'expérience dans un environnement hors production, en vérifiant que les seuils des conditions d'arrêt s'activent comme prévu pendant l'expérience et que l'observabilité est en place pour détecter une exception, plutôt que d'exécuter l'expérience directement en production.

Lorsque vous exécutez des expériences d'injection de défaillances, vérifiez que toutes les parties responsables sont bien informées. Communiquez avec les équipes appropriées, telles que les équipes en charge des opérations, les équipes chargées de la fiabilité du service et le service client pour leur indiquer quand les expériences seront exécutées et à quoi ils doivent s'attendre. Donnez à ces équipes les outils de communication nécessaires pour informer les personnes en charge de l'exécution de l'expérience si elles constatent des effets négatifs.

Vous devez restaurer la charge de travail et ses systèmes sous-jacents dans leur état fonctionnel et connu d'origine. En général, la conception résiliente de la charge de travail lui permet de s'auto-réparer. Cependant, certaines conceptions défaillantes ou échecs d'expériences peuvent laisser votre charge de travail dans un état d'échec inattendu. À la fin de l'expérience, vous devez en être conscient et restaurer la charge de travail et les systèmes. Avec AWS FIS, vous pouvez définir une configuration de barrière de protection (également appelée post action) dans les paramètres d'action. Une post action restaure la cible dans l'état dans lequel elle se trouvait avant l'exécution de l'action. Qu'elles soient automatisées (comme lorsque vous utilisez AWS FIS) ou manuelles, ces post actions doivent faire partie d'un playbook décrivant la façon de détecter et de gérer les échecs.

- Vérifier l'hypothèse.

  Principes de l'ingénierie du chaos donne des conseils sur la façon de vérifier l'état stable de votre charge de travail :

  Concentrez-vous sur le résultat mesurable d'un système, plutôt que sur les attributs internes du système. Les mesures de ce résultat sur une courte période de temps constituent un proxy pour l'état stable du système. Le débit général du système, les taux d'erreur et les centiles de latence peuvent tous être des métriques d'intérêt représentant un comportement d'état stable. En se focalisant sur les modèles de comportement systémique pendant les expériences, l'ingénierie du chaos vérifie que le système fonctionne, au lieu d'essayer de confirmer qu'il fonctionne.

  Dans nos deux exemples précédents, nous incluons la métrique de l'état stable inférieure à 0,01 % d'augmentation des erreurs (5xx) côté serveur et la métrique inférieure à 1 minute d'erreurs de lecture ou d'écriture de base de données.

Les erreurs 5xx constituent une bonne métrique, car elles sont une conséquence du mode de défaillance dont le client de la charge de travail fera l'expérience directement. La mesure des erreurs de base de données est correcte en tant que conséquence directe de la défaillance, mais doit être également complétée par une mesure d'impact, telle que les échecs de demandes client ou les erreurs remontées. Par ailleurs, incluez une surveillance synthétique (également appelée utilisateur canary) sur n'importe quelle API ou URI directement accessible par le client de votre charge de travail.

- Améliorer la conception de la charge de travail à des fins de résilience.

  Si l'état stable n'a pas été maintenu, enquêtez sur les moyens d'améliorer la conception de la charge de travail afin de réduire la défaillance, tout en appliquant les bonnes pratiques du pilier Fiabilité du cadre AWS Well-Architected. Des conseils et ressources supplémentaires sont disponibles dans la bibliothèque des créateurs AWS, qui héberge des articles sur la façon d' améliorer vos surveillances de l'état ou utiliser de nouvelles tentatives avec interruption dans votre code d'application, entre autres.

  Une fois ces changements implémentés, exécutez de nouveau l'expérience (illustrée par la ligne pointillée dans le volant d'inertie de l'ingénierie du chaos) pour déterminer son efficacité. Si l'étape de vérification indique que l'hypothèse est vraie, alors la charge de travail sera en état stable et le cycle continuera.

- Exécuter régulièrement des expériences.

  Une expérience de chaos est un cycle, et les expériences doivent être exécutées régulièrement dans le cadre de l'ingénierie du chaos. Lorsqu'une charge de travail correspond à l'hypothèse d'une expérience, cette dernière doit être automatisée pour s'exécuter en continu en tant que test de régression de votre pipeline CI/CD. Pour savoir comment faire, consultez ce blog sur l'exécution d'expériences AWS FIS en utilisant AWS CodePipeline. Cet atelier sur les expériences AWS FIS récurrentes dans un pipeline CI/CD vous permet de mettre la main à la pâte.

  Les expériences d'injection de défaillance font également partie des tests de simulation de pannes (consultez REL12-BP06 Organiser régulièrement des tests de simulation de panne). Les tests de simulation de pannes simulent une défaillance ou un événement pour vérifier les systèmes, les processus et la réponse de l'équipe. L'objectif est d'effectuer les actions que l'équipe effectuerait si un événement exceptionnel se produisait.

- Enregistrer et stocker les résultats des expériences.

Les résultats des expériences d'injection de défaillance doivent être enregistrés et conservés. Incluez toutes les données nécessaires (telles que l'heure, la charge de travail et les conditions) afin de pouvoir analyser ultérieurement les résultats de l'expérience et les tendances. Les exemples de résultats peuvent inclure des captures d'écran des tableaux de bord, des fichiers CSV de la base de données de votre/vos métriques ou un registre manuscrit des événements et observations pendant l'expérience. La journalisation des expériences avec AWS FIS peut faire partie de la collecte des données.

Ressources

Bonnes pratiques associées :

- REL08-BP03 Intégrer les tests de résilience dans le cadre de votre déploiement
- REL13-BP03 Effectuer un test de validation de la mise en œuvre de la reprise après sinistre

Documents connexes :

- Qu'est-ce qu'AWS Fault Injection Service ?
- Qu'est-ce qu'AWS Resilience Hub ?
- Principes de l'ingénierie du chaos
- Ingénierie du chaos : préparation de votre première expérience
- Ingénierie de résilience : apprendre à intégrer les pannes
- Témoignages d'utilisation de l'ingénierie du chaos
- Éviter les solutions de secours dans les systèmes distribués
- Déploiement canary pour des expériences de chaos

Vidéos connexes :

- AWS re:Invent 2020: Testing resiliency using chaos engineering (ARC316)
- AWS re:Invent 2019: Improving resiliency with chaos engineering (DOP309-R1)
- AWS re:Invent 2019: Performing chaos engineering in a serverless world (CMY301)

Exemples connexes :

- Atelier Well-Architected : niveau 300 : test de la résilience d'Amazon EC2, Amazon RDS et Amazon S3
- Atelier L'ingénierie du chaos dans AWS
- atelier Applications résilientes et Well-Architected avec l'ingénierie du chaos
- Atelier Chaos sans serveur
- Atelier Mesurer et améliorer la résilience de votre application avec AWS Resilience Hub

Outils associés :

- AWS Fault Injection Service
- AWS Marketplace : Gremlin Chaos Engineering Platform
- Chaos Toolkit
- Chaos Mesh
- Litmus

REL12-BP06 Organiser régulièrement des tests de simulation de panne

Utilisez des tests de simulation de panne pour exercer régulièrement vos procédures de réponse aux événements et aux défaillances aussi près que possible de la production (y compris dans les environnements de production) avec les personnes qui seront impliquées dans les scénarios de défaillance réels. Les tests de simulation de panne appliquent des mesures pour s'assurer que les événements de production n'affectent pas les utilisateurs.

Ils simulent une défaillance ou un événement pour tester les systèmes, les processus et la réponse de l'équipe. L'objectif est d'effectuer les actions que l'équipe effectuerait si un événement exceptionnel se produisait. Cela vous aidera à comprendre où apporter des améliorations et à développer une expérience de gestion des événements au sein de votre organisation. Des tests de simulation de panne doivent être effectués régulièrement afin que votre équipe se forge une « mémoire musculaire » quant à la façon de réagir.

Une fois votre conception de résilience en place et testée dans des environnements non liés à la production, un test de simulation de panne permet de s'assurer que tout fonctionne comme prévu en production. Un test de simulation de pannes, particulièrement le premier, est une activité « exploitant toutes les ressources ». L'intégralité des ingénieurs et des opérations est informée de ce qui se passera et quand. Les playbooks sont en place. Des événements simulés sont exécutés,

y compris des événements de défaillance possibles, dans les systèmes de production de la manière prescrite, et l'impact est évalué. Si tous les systèmes fonctionnent comme prévu, la détection et l'auto-réparation se produiront avec peu, voire aucun impact. En revanche, si un impact négatif est observé, le test est annulé et les problèmes de charge de travail sont résolus, manuellement au besoin (à l'aide du runbook). Étant donné que les tests de simulation de pannes se déroulent souvent en production, toutes les précautions doivent être prises pour s'assurer de l'absence d'impact sur la disponibilité pour vos clients.

Anti-modèles courants :

- Documenter vos procédures sans jamais les exercer.
- Non-inclusion des décideurs commerciaux dans les exercices de test.

Avantages liés au respect de cette bonne pratique : L'organisation régulière de tests de simulation de panne a un double avantage. D'une part, elle permet de s'assurer que tout le personnel suit les stratégies et les procédures lorsqu'un incident réel se produit. D'autre part, elle facilite la validation de l'adéquation de ces stratégies et procédures.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

- Planifiez des tests de simulation de panne pour tester régulièrement vos runbooks et vos playbooks. Les tests de simulation de panne doivent impliquer tous ceux qui seraient affectés par une interruption de production : le propriétaire de l'entreprise, les développeurs, le personnel d'exploitation et les équipes d'interventions.
  - Effectuez vos tests de charge ou de performances et mettez en œuvre l'injection de défaillances.
  - Recherchez des anomalies dans vos runbooks et des possibilités de test de vos playbooks.
    - Si vous vous écartez de vos runbooks, affinez-les ou corrigez le comportement. Lors des tests de votre playbook,, identifiez les runbooks qui auraient dû être utilisés ou créez-en de nouveaux.

Ressources

Documents connexes :

- Qu'est-ce qu'AWS GameDay ?

Vidéos connexes :

- AWS re:Invent 2019: Improving resiliency with chaos engineering (DOP309-R1)

Exemples connexes :

- AWS Well-Architected Labs - Testing Resiliency

Planification de la reprise après sinistre

La mise en place de sauvegardes et de composants de charge de travail redondants constitue le début de votre stratégie de DR. RTO et RPO sont vos objectifs pour la restauration de votre charge de travail. Définissez-les en fonction des besoins de l'entreprise. Mettez en œuvre une stratégie pour atteindre ces objectifs, en particulier en tenant compte de l'emplacement et de la fonction des données et des ressources de charge de travail. La probabilité d'une perturbation et le coût de la reprise sont également des facteurs clés qui permettent de déterminer la valeur opérationnelle de la reprise après sinistre d'une charge de travail.

La disponibilité et la reprise après sinistre s'appuient sur les mêmes bonnes pratiques, telles que la surveillance des pannes, le déploiement sur plusieurs sites et le basculement automatique. Cependant, la disponibilité se concentre sur les composants de la charge de travail, tandis que la reprise après sinistre se concentre sur des copies distinctes de l'ensemble de la charge de travail. La reprise après sinistre a des objectifs différents de la disponibilité, car elle se concentre sur le temps de récupération après un sinistre.

Bonnes pratiques

- REL13-BP01 Définir les objectifs de reprise pour les temps d'arrêt et les pertes de données
- REL13-BP02 Utiliser des stratégies de reprise définies pour répondre aux objectifs de reprise
- REL13-BP03 Effectuer un test de validation de la mise en œuvre de la reprise après sinistre
- REL13-BP04 Gérer l'écart de configuration au niveau du site ou de la région de reprise après sinistre
- REL13-BP05 Automatiser la reprise

REL13-BP01 Définir les objectifs de reprise pour les temps d'arrêt et les pertes de données

La charge de travail est associée à un objectif de délai de reprise (RTO) et à un objectif de point de reprise (RPO).

La durée maximale d'interruption admissible (RTO) correspond au délai maximum acceptable entre l'interruption du service et la restauration du service. Elle détermine ce qui est considéré comme étant un créneau de temps acceptable d'indisponibilité du service.

L'objectif de point de reprise (RPO) correspond au temps maximal acceptable depuis le dernier point de reprise des données. Il détermine ce qui est considéré comme étant une perte de données acceptable entre le dernier point de reprise et l'interruption du service.

Les valeurs RTO et RPO sont des considérations importantes lors de la sélection d'une stratégie de reprise après sinistre adaptée à votre charge de travail. Ces objectifs sont déterminés par l'entreprise, puis utilisés par les équipes techniques pour sélectionner et mettre en œuvre une stratégie de reprise après sinistre.

Résultat souhaité :

Un RTO et un RPO, définis en fonction de l'impact sur l'entreprise, sont attribués à chaque charge de travail. Un niveau prédéfini, définissant la disponibilité du service et une perte de données acceptable, avec un RTO et un RPO associés est assigné à la charge de travail. Si cette hiérarchisation n'est pas possible, elle peut être attribuée sur mesure pour chaque charge de travail, dans l'intention de créer des niveaux ultérieurement. Le RTO et le RPO font partie des principaux éléments pris en compte pour la sélection de la mise en œuvre d'une stratégie de reprise après sinistre pour la charge de travail. D'autres considérations dans le choix d'une stratégie de reprise après sinistre sont les contraintes de coût, les dépendances de la charge de travail et les exigences opérationnelles.

Pour le RTO, identifiez l'impact en fonction de la durée d'une panne. Est-il linéaire ou non (par exemple, après quatre heures, vous arrêtez une ligne de fabrication jusqu'au début du prochain quart de travail) ?

Une matrice de reprise après sinistre, comme la suivante, peut vous aider à comprendre dans quelle mesure l'ordre d'importance de la charge de travail est lié aux objectifs de reprise. Notez que les valeurs réelles des axes X et Y doivent être personnalisées en fonction des besoins de votre organisation.

Matrice de reprise après sinistre						
		Objectif de point de reprise				
		Moins de 1 minute	Moins de 1 heure	Moins de 6 heures	Moins de 1 jour	+ de 1 jour
Durée maximale d'interruption	Moins de 10 minutes	Critique	Critique	Débit	Moyenne entreprise	Moyenne entreprise
	Moins de 2 heures	Critique	Débit	Moyenne entreprise	Moyenne entreprise	Faible
	Moins de 8 heures	Débit	Moyenne entreprise	Moyenne entreprise	Faible	Faible
	Moins de 24 heures	Moyenne entreprise	Moyenne entreprise	Faible	Faible	Faible
	+ de 24 heures	Moyenne entreprise	Faible	Faible	Faible	Faible

Figure 16 : matrice de reprise après sinistre

Anti-modèles courants :

- Aucun objectif de reprise défini.
- Sélection d'objectifs arbitraires de reprise.
- Sélection d'objectifs de reprise trop lents et qui ne répondent pas aux objectifs de l'entreprise.
- Ne pas comprendre l'impact des temps d'arrêt et de la perte de données.
- Sélection d'objectifs de reprise irréalistes, tels que zéro temps de reprise et zéro perte de données, qui peuvent ne pas être réalisables pour la configuration de votre charge de travail.
- Sélection d'objectifs de reprise plus rigoureux que les objectifs commerciaux réels. Cela impose des implémentations de reprise après sinistre qui sont plus coûteuses et plus compliquées que ce dont a besoin la charge de travail.
- Sélection d'objectifs de reprise incompatibles avec ceux d'une charge de travail dépendante.
- Vos objectifs de reprise ne tiennent pas compte des exigences de conformité réglementaire.
- Définition de RTO et RPO jamais testés pour une charge de travail.

Avantages liés au respect de cette bonne pratique : Vos objectifs de reprise en cas de perte de temps et de données sont nécessaires pour guider votre implémentation de DR.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Débit

Directives d'implémentation

Pour la charge de travail donnée, vous devez comprendre l'impact des temps d'arrêt et de la perte de données sur votre entreprise. L'impact augmente généralement avec les temps d'arrêt ou les pertes de données plus importants, mais son ampleur peut varier en fonction du type de charge de travail. Par exemple, vous pouvez tolérer des temps d'arrêt pouvant atteindre une heure avec peu d'impact, mais au-delà de ce délai, l'impact augmente rapidement. L'impact sur l'entreprise se manifeste sous de nombreuses formes, notamment le coût (tel que la perte de revenus), la confiance des clients (et l'impact sur la réputation), les problèmes opérationnels (tels que l'absence d'employés ou la baisse de productivité) et le risque réglementaire. Utilisez les étapes suivantes pour comprendre ces impacts et définir le RTO et le RPO pour votre charge de travail.

Étapes d'implémentation

1. Identifiez les parties prenantes spécifiques à cette charge de travail et collaborez avec elles pour mettre en œuvre ces étapes. Les objectifs de reprise d'une charge de travail relèvent d'une décision de l'entreprise. Les équipes techniques travaillent ensuite avec les parties prenantes de l'entreprise pour utiliser ces objectifs afin de sélectionner une stratégie de reprise après sinistre.

   > **Note**
   >
   > Pour les étapes 2 et 3, vous pouvez utiliser the section called “Fiche d'implémentation”.

2. Répondez aux questions ci-dessous pour rassembler les informations nécessaires pour prendre une décision.
3. Utilisez-vous des catégories ou des niveaux de criticité pour déterminer l'impact de la charge de travail dans votre organisation ?
   a. Si oui, affectez cette charge de travail à une catégorie.
   b. Dans le cas contraire, définissez ces catégories. Créez cinq catégories ou moins et affinez la plage de vos objectifs de délai et de point de reprise. Exemples de catégories : critique, élevé, moyen, faible. Pour comprendre comment les charges de travail correspondent aux catégories, déterminez si la charge de travail est stratégique, importante pour l'entreprise ou non commerciale.
   c. Définissez le RTO et le RPO de la charge de travail en fonction de sa catégorie. Choisissez toujours une catégorie plus stricte (RTO et RPO inférieurs) que les valeurs brutes calculées au début de cette étape. Si cela entraîne une variation de valeur trop importante, envisagez de créer une autre catégorie.

4. En fonction de ces réponses, attribuez des valeurs de RTO et RPO à la charge de travail. Cela peut se faire directement ou en affectant la charge de travail à un niveau de service prédéfini.
5. Documentez le plan de reprise après sinistre (DRP) pour cette charge de travail, qui fait partie du plan de continuité d'activité (BCP), à un emplacement accessible à l'équipe responsable de la charge de travail et aux parties prenantes.
   a. Enregistrez le RTO et le RPO, ainsi que les informations utilisées pour déterminer ces valeurs. Spécifiez la stratégie utilisée pour évaluer l'impact de la charge de travail sur l'entreprise.
   b. Enregistrez d'autres métriques que le RTO et le RPO que vous suivez ou prévoyez de suivre pour les objectifs de reprise après sinistre.
   c. Vous ajouterez les détails de votre stratégie de reprise après sinistre et de votre runbook à ce plan lorsque vous les créerez.
6. En recherchant la criticité de la charge de travail dans une matrice telle que celle de la figure 15, vous pouvez commencer à établir des niveaux de service prédéfinis pour votre organisation.
7. Après avoir mis en œuvre une stratégie de reprise après sinistre (ou une preuve de concept pour une stratégie de reprise après sinistre) conformément à the section called “REL13-BP02 Utiliser des stratégies de reprise définies pour répondre aux objectifs de reprise”, testez cette stratégie pour déterminer les valeurs RTC (temps de reprise possible) et RPC (point de reprise possible) réelles de la charge de travail. Si ceux-ci n'atteignent pas les objectifs de reprise cibles, vous pouvez soit collaborer avec les parties prenantes de votre entreprise pour les ajuster, soit apporter des modifications à la stratégie de reprise après sinistre, le cas échéant, pour atteindre ces objectifs.

Questions principales

1. Quelle est la durée maximale pendant laquelle la charge de travail peut être interrompue avant qu'un impact grave n'affecte l'entreprise ?
   a. Déterminez le coût (impact financier direct) pour l'entreprise par minute où la charge de travail est interrompue.
   b. Considérez que l'impact n'est pas toujours linéaire. L'impact peut être limité au début, puis augmenter rapidement au-delà d'un point critique dans le temps.
2. Quelle est la quantité maximale de données pouvant être perdues avant qu'un impact grave n'affecte l'entreprise ?
   a. Déterminez cette valeur en fonction de votre magasin de données le plus critique. Identifiez la criticité respective pour les autres magasins de données.

   b. Les données de la charge de travail peuvent-elles être recréées en cas de perte ? Si cette approche est plus facile sur le plan opérationnel que la sauvegarde et la restauration, choisissez le RPO en fonction de la criticité des données sources utilisées pour recréer les données de la charge de travail.

3. Quels sont les objectifs de reprise et les attentes de disponibilité des charges de travail dont celle-ci dépend (en aval) ou des charges de travail qui dépendent de celle-ci (en amont) ?
   a. Choisissez des objectifs de reprise qui permettent à cette charge de travail de répondre aux exigences des dépendances en amont.
   b. Choisissez des objectifs de reprise réalisables compte tenu des capacités de reprise des dépendances en aval. Les dépendances en aval non critiques (celles que vous pouvez « contourner ») peuvent être exclues. Ou, si nécessaire, traitez les dépendances critiques en aval pour améliorer leurs capacités de reprise.

Questions supplémentaires

Envisagez ces questions et dans quelle mesure elles s'appliquent à cette charge de travail :

4. Avez-vous des RTO et des RPO différents selon le type de panne (région ou AZ, etc.) ?
5. Existe-t-il un moment précis (saisonnalité, événements commerciaux, lancements de produits) où votre RTO/RPO peut changer ? Si oui, en quoi diffèrent-ils et quelle est leur limite de temps ?
6. Combien de clients seront touchés si la charge de travail est interrompue ?
7. Quel sera l'impact sur la réputation si la charge de travail est interrompue ?
8. Quels autres impacts opérationnels peuvent entrer en jeu si la charge de travail est interrompue ? Par exemple, la productivité des employés sera affectée si les systèmes de messagerie ne sont pas disponibles ou si les systèmes de paie ne sont pas en mesure de soumettre des transactions.
9. Comment le RTO et le RPO de la charge de travail s'alignent-ils sur la stratégie de reprise après sinistre de la succursale et de l'organisation ?
10. Existe-t-il des obligations contractuelles internes régissant la prestation d'un service ? Des sanctions sont-elles appliquées en cas de non-respect ?
11. Quelles sont les contraintes réglementaires ou de conformité liées aux données ?

Fiche d'implémentation

Vous pouvez utiliser cette feuille de calcul pour les étapes d'implémentation 2 et 3. Vous pouvez l'ajuster en fonction de vos besoins spécifiques, par exemple en ajoutant des questions supplémentaires.

Étape 2 : Questions principales	S'applique à la charge de travail ?	RPO de la charge de travail	RPO de la charge de travail	Ajustement de RTO.	Ajustement de RPO.	Instructions
[1] durée maximale pendant laquelle la charge de travail peut être inactive						mesuré en temps depuis le début de la panne jusqu'à la récupération
[2] quantité maximale de données pouvant être perdues						mesuré dans le temps depuis le dernier jeu de données restaurable
[3a] dépendances en amont						saisissez les objectifs de récupération en amont les plus stricts
[3b] dépendances en aval						saisissez les objectifs de récupération en aval les moins stricts
[3a] dépendances en amont rapprochées						Si la valeur en amont est inférieure aux valeurs actuelles et la valeur en aval supérieure,
[3b] dépendances en aval rapprochées						manipulez les dépendances pour les rapprocher et entrez les valeurs rapprochées ici
[3] dépendances						réduisez les valeurs pour répondre aux dépendances en amont ou augmentez-les selon les fonctionnalités de dépendance en aval
Étape 2 : Questions supplémentaires						Indiquez si la question s'applique. Dans le cas contraire, ignorez-la
RTO/RPO de base						Transférez les valeurs RTO et RPO d'en haut jusqu'ici
[4] type de panne	[ ]O / [ ]N					Indiquez les objectifs de récupération pour les événements avec les exigences les plus strictes
[5] objectifs temporels spécifiques	[ ]O / [ ]N					Indiquez les objectifs de récupération pour les durées avec les exigences les plus strictes
[6] clients perturbés	[ ]O / [ ]N					Représentez graphiquement les clients impactés en fonction du temps d'arrêt ou de la perte de données. Utilisez ces informations pour saisir le RTO et le RPO maximum autorisés en fonction de l'impact sur le client
[7] impact sur la réputation	[ ]O / [ ]N					Déterminez avec l'entreprise le RTO et le RPO maximum en fonction de l'impact sur la réputation
[8] impact opérationnel	[ ]O / [ ]N					Indiquez un RTO et un RPO maximum en fonction de l'impact opérationnel
[9] alignement organisationnel	[ ]O / [ ]N					Indiquez le RTO et le RPO maximum pour les charges de travail de ce type selon les exigences LOB et organisationnelles
[10] obligations contractuelles	[ ]O / [ ]N					Indiquez un RTO et un RPO maximum en fonction des obligations contractuelles
[11] conformité réglementaire	[ ]O / [ ]N					Indiquez le RTO et le RPO maximum en fonction de la conformité réglementaire applicable
cible basée sur des questions supplémentaires						Prenez la valeur minimale (valeur plus stricte) des Q 11-4 et entrez-la ici
cible ajustée						Si les objectifs de la ligne ci-dessus ne peuvent pas être atteints, collaborez avec les parties prenantes pour assouplir les contraintes et entrez un nouveau minimum ici
RTO/RPO ajusté						Indiquez les valeurs RPO/RTO de base, ou la cible ajustée, selon la valeur la plus basse
Étape 3						
Mapper vers une catégorie ou un niveau prédéfini						Ajustez les deux valeurs vers le bas (méthode plus stricte) pour vous aligner sur le niveau défini le plus proche

Fiche

Niveau d'effort du plan d'implémentation : Faible

Ressources

Bonnes pratiques associées :

- the section called “REL09-BP04 Effectuer une récupération périodique des données pour vérifier l'intégrité et les processus de sauvegarde”
- the section called “REL13-BP02 Utiliser des stratégies de reprise définies pour répondre aux objectifs de reprise”
- the section called “REL13-BP03 Effectuer un test de validation de la mise en œuvre de la reprise après sinistre”

Documents connexes :

- Blog d'architecture AWS : série sur la reprise après sinistre
- Reprise après sinistre des charges de travail sur AWS : reprise dans le cloud (livre blanc AWS)
- Gestion des politiques de résilience avec AWS Resilience Hub
- Partenaire APN : partenaires pouvant faciliter la reprise après sinistre
- AWS Marketplace : produits pouvant être utilisés pour la reprise après sinistre

Vidéos connexes :

- AWS re:Invent 2018: Architecture Patterns for Multi-Region Active-Active Applications (ARC209-R2)
- Disaster Recovery of Workloads on AWS

REL13-BP02 Utiliser des stratégies de reprise définies pour répondre aux objectifs de reprise

Définissez une stratégie de reprise après sinistre qui répond aux objectifs de reprise de votre charge de travail. Choisissez une stratégie telle que : sauvegarde et restauration, mode secours (actif/passif) ou actif/actif.

Résultat souhaité : pour chaque charge de travail, il existe une stratégie de reprise après sinistre définie et implémentée qui permet à cette charge de travail d'atteindre les objectifs de reprise. Les stratégies de reprise après sinistre entre les charges de travail utilisent des modèles réutilisables (comme les stratégies décrites précédemment).

Anti-modèles courants :

- Mettre en œuvre des procédures de récupération incohérentes pour les charges de travail avec des objectifs de reprise après sinistre similaires.
- Conserver l'implémentation ad hoc de la stratégie de reprise après sinistre lorsqu'un sinistre se produit.
- Ne pas avoir de plan de reprise après sinistre.
- Être dépendant des opérations du plan de contrôle pendant la récupération.

Avantages liés au respect de cette bonne pratique :

- L'utilisation de stratégies de reprise définies vous permet d'utiliser des outils et des procédures de test courantes.
- L'utilisation de stratégies de reprise définies améliore le partage des connaissances entre les équipes et la mise en œuvre de la reprise après sinistre sur les charges de travail qu'elles possèdent.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé. Sans une stratégie de reprise après sinistre planifiée, mise en œuvre et testée, il est peu probable que vous atteigniez vos objectifs de reprise en cas de sinistre.

Directives d'implémentation

Une stratégie de reprise après sinistre repose sur la capacité à rétablir votre charge de travail sur un site de reprise si votre emplacement principal ne parvient plus à exécuter cette charge de travail. Les objectifs de récupération les plus courants sont le RTO et le RPO, comme indiqué dans REL13-BP01 Définir les objectifs de reprise pour les temps d'arrêt et les pertes de données.

Une stratégie de reprise après sinistre sur plusieurs zones de disponibilité (AZ) au sein d'une seule Région AWS peut vous prémunir contre les événements catastrophiques tels que les incendies, les inondations et les pannes de courant majeures. S'il est nécessaire de mettre en œuvre une protection contre un événement improbable qui empêcherait votre charge de travail de s'exécuter dans une Région AWS donnée, optez pour une stratégie de reprise après sinistre qui utilise plusieurs régions.

Lors de la conception d'une stratégie de reprise après sinistre dans plusieurs régions, vous devez choisir l'une des approches suivantes. Elles sont répertoriées par ordre croissant de coûts et de complexité et par ordre décroissant de RTO et RPO. La région de reprise fait référence à une Région AWS autre que la région principale utilisée pour votre charge de travail.

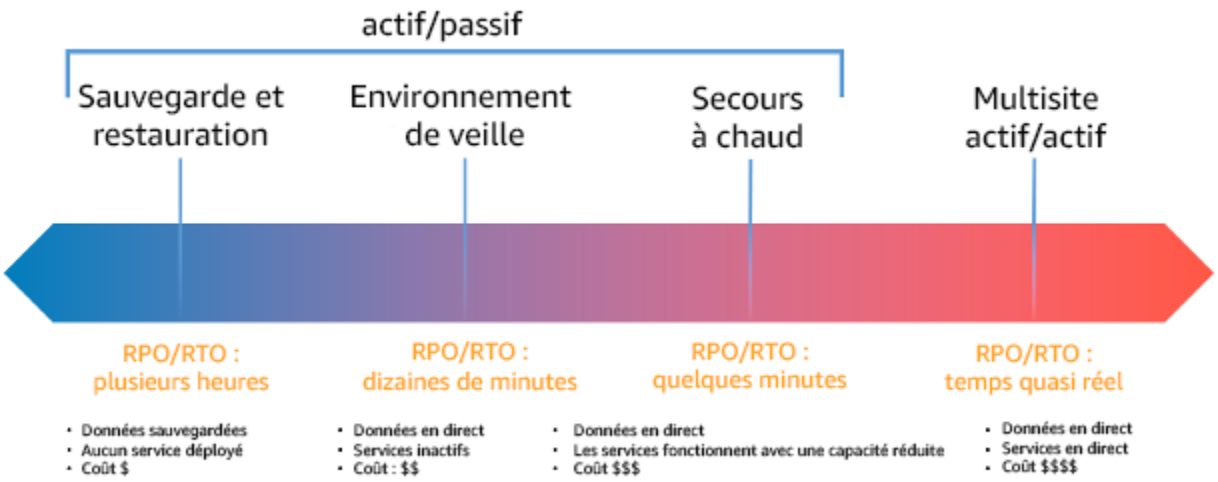


Figure 17 : stratégies de reprise après sinistre

- Sauvegarde et restauration (RPO en heures, RTO de 24 heures maximum) : sauvegardez vos données et applications dans la région de reprise après sinistre. L'utilisation de sauvegardes automatisées ou continues permet une récupération ponctuelle, ce qui peut réduire le RPO à seulement 5 minutes dans certains cas. En cas de sinistre, vous déployez votre infrastructure (en utilisant l'infrastructure en tant que code pour réduire le RTO), déployez votre code et restaurez les données sauvegardées pour vous remettre d'un sinistre dans la région de reprise.
- Environnement en veille (RPO de quelques minutes, RTO de dizaines de minutes) : allouez une copie de votre infrastructure de charge de travail principale dans la région de reprise. Répliquez vos données dans la région de reprise et créez-y des sauvegardes. Les ressources requises pour prendre en charge la réplication et la sauvegarde des données, telles que les bases de données et le stockage d'objets, sont toujours actives. D'autres éléments tels que les serveurs d'applications ou le calcul sans serveur ne sont pas déployés, mais peuvent être créés si nécessaire avec la configuration et le code d'application requis.
- Secours à chaud (RPO de quelques secondes, RTO de quelques minutes) : maintenez une version réduite d'une charge de travail entièrement fonctionnelle qui s'exécute toujours dans la région de reprise. Les systèmes stratégiques sont entièrement dupliqués et sont toujours opérationnels, mais avec une flotte réduite. Les données sont répliquées dans la région de reprise et y sont hébergées. Lorsque vient le moment de la reprise, le système est rapidement mis à l'échelle pour gérer la charge de production. Plus l'échelle du secours à chaud est élevée, plus la dépendance au RTO et au plan de contrôle est faible. Lorsqu'elle est complètement graduée, on parle de zone hébergée.

- Multi-région (multi-site) actif-actif (RPO proche de zéro, RTO potentiellement nul) : votre charge de travail est déployée et dessert activement le trafic à partir de plusieurs Régions AWS. Cette stratégie vous oblige à synchroniser les données entre les régions. Il est important d'éviter ou de gérer les éventuels conflits causés par des écritures sur le même enregistrement dans deux réplicas régionaux différents, ce qui peut être complexe. La réplication des données est utile pour la synchronisation des données et vous protège contre certains types de sinistres. Toutefois, elle ne vous protège pas contre la corruption ou la destruction des données à moins que votre solution n'inclue également des options de récupération ponctuelle.

> **Note**
>
> La différence entre l'environnement en veille et le secours à chaud est parfois difficile à cerner. Ces deux stratégies incluent un environnement dans votre région de reprise avec des copies des ressources de votre région principale. L'environnement en veille diffère en ce qu'il ne peut pas traiter les demandes sans qu'une action supplémentaire soit entreprise au préalable, tandis que le secours à chaud peut gérer le trafic (à des niveaux de capacité réduits) immédiatement. L'environnement en veille vous oblige à allumer des serveurs, à déployer éventuellement une infrastructure supplémentaire (non essentielle) et à augmenter l'échelle, tandis que le secours à chaud nécessite uniquement une augmentation de l'échelle (tout est déjà déployé et en cours d'exécution). Choisissez entre ces options en fonction de vos besoins en termes de RTO et de RPO.
>
> Si le coût est un problème et que vous souhaitez atteindre des objectifs de RPO et RTO similaires à ceux définis dans la stratégie de secours à chaud, vous pouvez envisager des solutions natives du cloud, comme AWS Elastic Disaster Recovery, qui adoptent l'approche de l'environnement de veille et offrent des objectifs de RPO et RTO améliorés.

Étapes d'implémentation

1. Déterminez une stratégie de reprise après sinistre qui répond aux exigences de récupération pour cette charge de travail.

Le choix d'une stratégie de reprise après sinistre vise à trouver un juste milieu entre la réduction des temps d'arrêt et de la perte de données (RTO et RPO) et le coût et la complexité liées à la mise en œuvre de cette stratégie. Évitez de mettre en œuvre une stratégie plus stricte que nécessaire, car cela entraînerait des coûts inutiles.

Par exemple, dans le diagramme suivant, l'entreprise a déterminé son RTO maximal autorisé ainsi que la limite de dépenses possible pour sa stratégie de restauration de service. Compte tenu des objectifs de l'entreprise, les stratégies environnement en veille et secours à chaud satisfont à la fois aux critères de RTO et de coût.

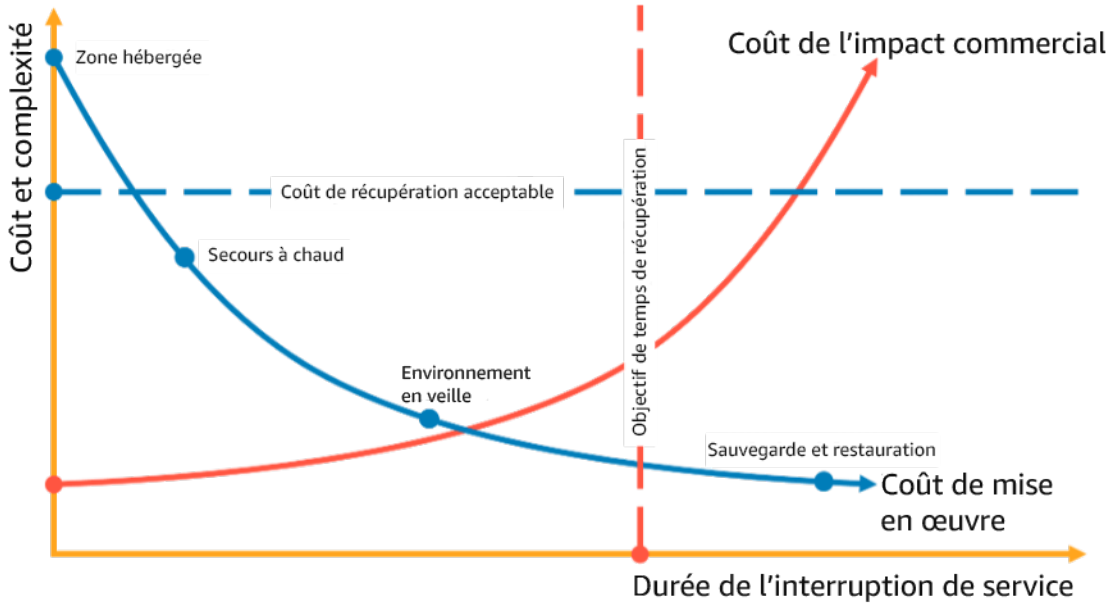


Figure 18 : choix d'une stratégie de reprise après sinistre basée sur le RTO et le coût

Pour en savoir plus, consultez Business Continuity Plan (BCP) [Plan de continuité d'activité (PCA)].

2. Passez en revue les modèles de mise en œuvre de la stratégie de reprise après sinistre sélectionnée.

Cette étape consiste à comprendre comment mettre en œuvre la stratégie sélectionnée. Les stratégies reposent sur l'utilisation de Régions AWS comme site principal et site de reprise. Cependant, vous pouvez également choisir d'utiliser des zones de disponibilité dans une seule région comme stratégie de reprise après sinistre, ce qui permet d'exploiter des éléments de plusieurs de ces stratégies.

Dans les étapes suivantes, vous pouvez appliquer la stratégie à votre charge de travail spécifique.

Sauvegarde et restauration

La sauvegarde et restauration est la stratégie la moins complexe à mettre en œuvre, mais nécessite plus de temps et d'efforts pour la restauration de la charge de travail, ce qui entraîne un RTO et un

RPO plus élevés. Il est conseillé de toujours faire des sauvegardes de vos données et de les copier sur un autre site (comme une autre Région AWS).

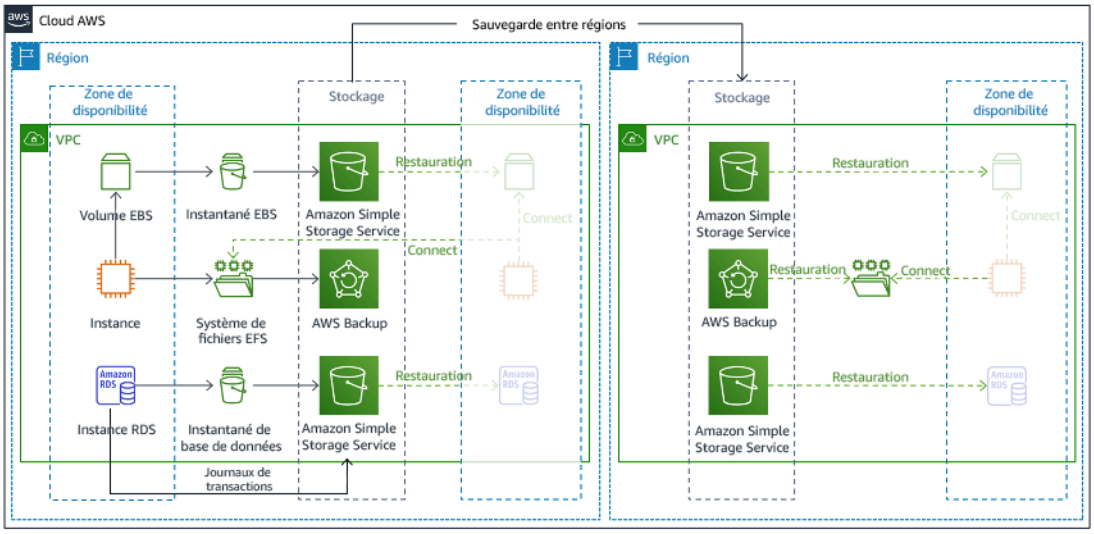


Figure 19 : architecture de sauvegarde et de restauration

Pour obtenir plus de détails sur cette stratégie, consultez Disaster Recovery (DR) Architecture on AWS, Part II: Backup and Restore with Rapid Recovery [Architecture de reprise après sinistre (DR) sur AWS, partie II : sauvegarde et restauration avec récupération rapide].

Environnement en veille

Avec l'approche de l'environnement en veille, vous répliquez vos données depuis la région principale vers la région de reprise. Les ressources principales utilisées pour l'infrastructure de charge de travail sont déployées dans la région de reprise, mais des ressources supplémentaires et toutes les dépendances sont toujours nécessaires pour en faire une pile fonctionnelle. Par exemple, dans la figure 20, aucune instance de calcul n'est déployée.

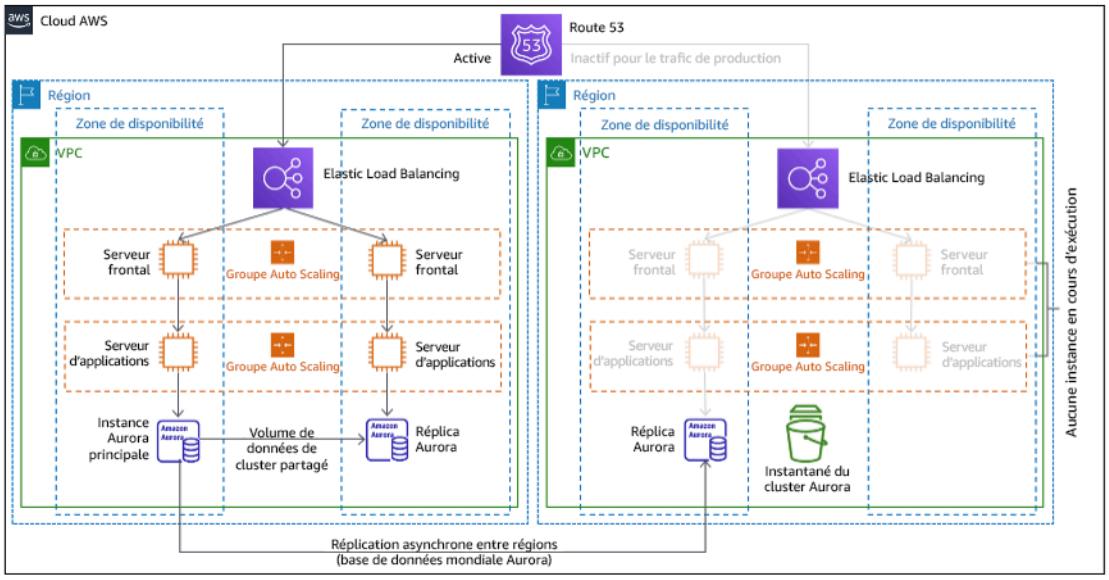


Figure 20 : architecture avec environnement en veille

Pour obtenir plus de détails sur cette stratégie, consultez Disaster Recovery (DR) Architecture on AWS, Part III: Pilot Light and Warm Standby [Architecture de reprise après sinistre (DR) sur AWS, partie III : environnement de veille et secours à chaud].

Secours à chaud

Le secours à chaud consiste à s'assurer qu'il existe une copie réduite, mais entièrement fonctionnelle, de votre environnement de production dans une autre région. Cette approche étend le concept d'environnement en veille et réduit le temps de récupération, car votre charge de travail reste active dans une autre région. Si la région de reprise est déployée à pleine capacité, on parle de zone hébergée.

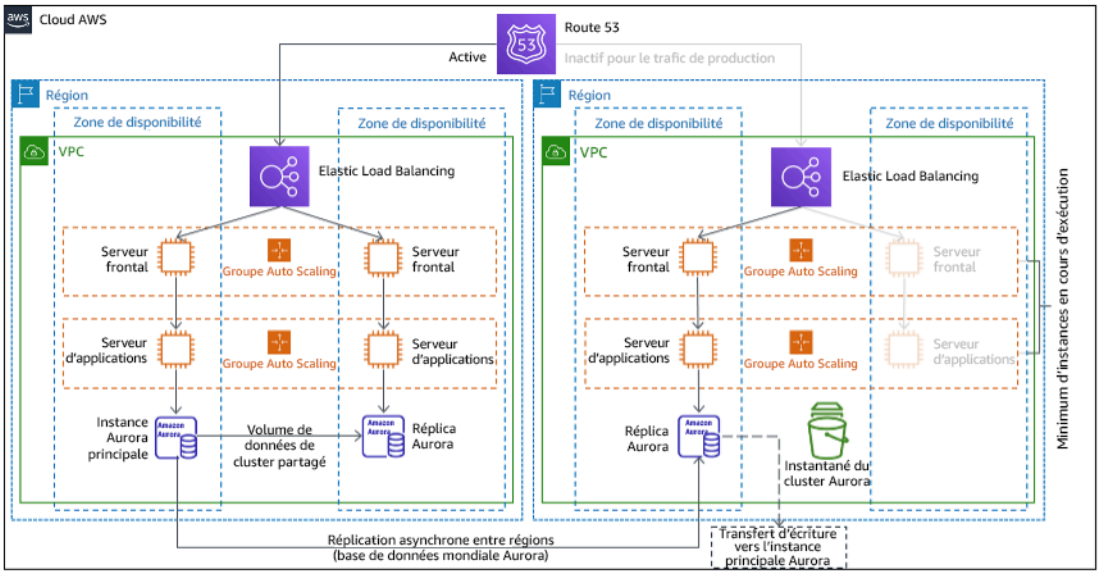


Figure 21 : Architecture de secours à chaud

L'utilisation du secours à chaud ou de l'environnement en veille nécessite une augmentation des ressources dans la région de reprise. Pour vérifier que la capacité est disponible en cas de besoin, envisagez l'utilisation des réserves de capacité pour les instances EC2. Si vous utilisez AWS Lambda, alors la simultanéité allouée peut provisionner des environnements d'exécution afin qu'ils soient prêts à répondre immédiatement aux appels de votre fonction.

Pour obtenir plus de détails sur cette stratégie, consultez Disaster Recovery (DR) Architecture on AWS, Part III: Pilot Light and Warm Standby [Architecture de reprise après sinistre (DR) sur AWS, partie III : environnement de veille et secours à chaud].

Multisite actif/actif

Vous pouvez exécuter votre charge de travail simultanément dans plusieurs régions dans le cadre d'une stratégie multisite actif/actif. Une stratégie multisite actif/actif dessert le trafic de toutes les régions dans lesquelles il est déployé. Les clients peuvent sélectionner cette stratégie pour des raisons autres que la reprise après sinistre. Elle peut être utilisée pour augmenter la disponibilité ou lors du déploiement d'une charge de travail auprès d'une audience mondiale (pour rapprocher le point de terminaison des utilisateurs et/ou déployer des piles localisées pour l'audience de cette région). En tant que stratégie de reprise après sinistre, si la charge de travail ne peut pas être prise en charge dans l'une des Régions AWS vers lesquelles elle est déployée, cette région est évacuée, et les régions restantes sont utilisées pour assurer la disponibilité. La stratégie de reprise après sinistre multisite actif/actif est la plus complexe sur le plan opérationnel et ne doit être sélectionnée que lorsque les besoins de l'entreprise l'exigent.

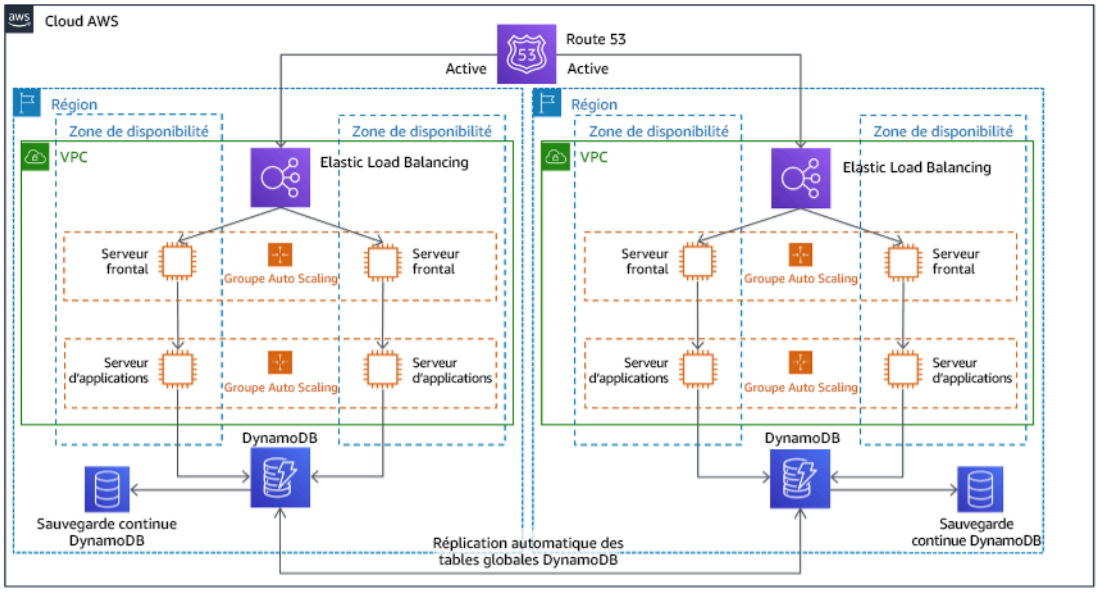


Figure 22 : architecture multisite de type actif/actif

Pour obtenir plus de détails sur cette stratégie, consultez Disaster Recovery (DR) Architecture on AWS, Part IV: Multi-site Active/Active [Architecture de reprise après sinistre (DR) sur AWS, partie IV : multisite actif/actif].

AWS Elastic Disaster Recovery

Si vous envisagez une stratégie d'environnement de veille ou de secours à chaud pour la reprise après sinistre, AWS Elastic Disaster Recovery pourrait constituer une approche alternative offrant de meilleurs avantages.Elastic Disaster Recovery peut offrir un objectif de RPO et de RTO similaire à celui du secours à chaud, tout en conservant l'approche économique de l'environnement de veille. Elastic Disaster Recovery réplique vos données de votre région principale vers votre région de reprise, en utilisant la protection continue des données pour atteindre un RPO mesuré en secondes et un RTO qui peut être mesuré en minutes. Seules les ressources nécessaires à la réplication des données sont déployées dans la région de reprise, ce qui permet de limiter les coûts, à l'instar de la stratégie de l'environnement de veille. En cas d'utilisation de Elastic Disaster Recovery, le service coordonne et orchestre la récupération des ressources informatiques lorsqu'elle est initiée dans le cadre d'un basculement ou d'une opération.

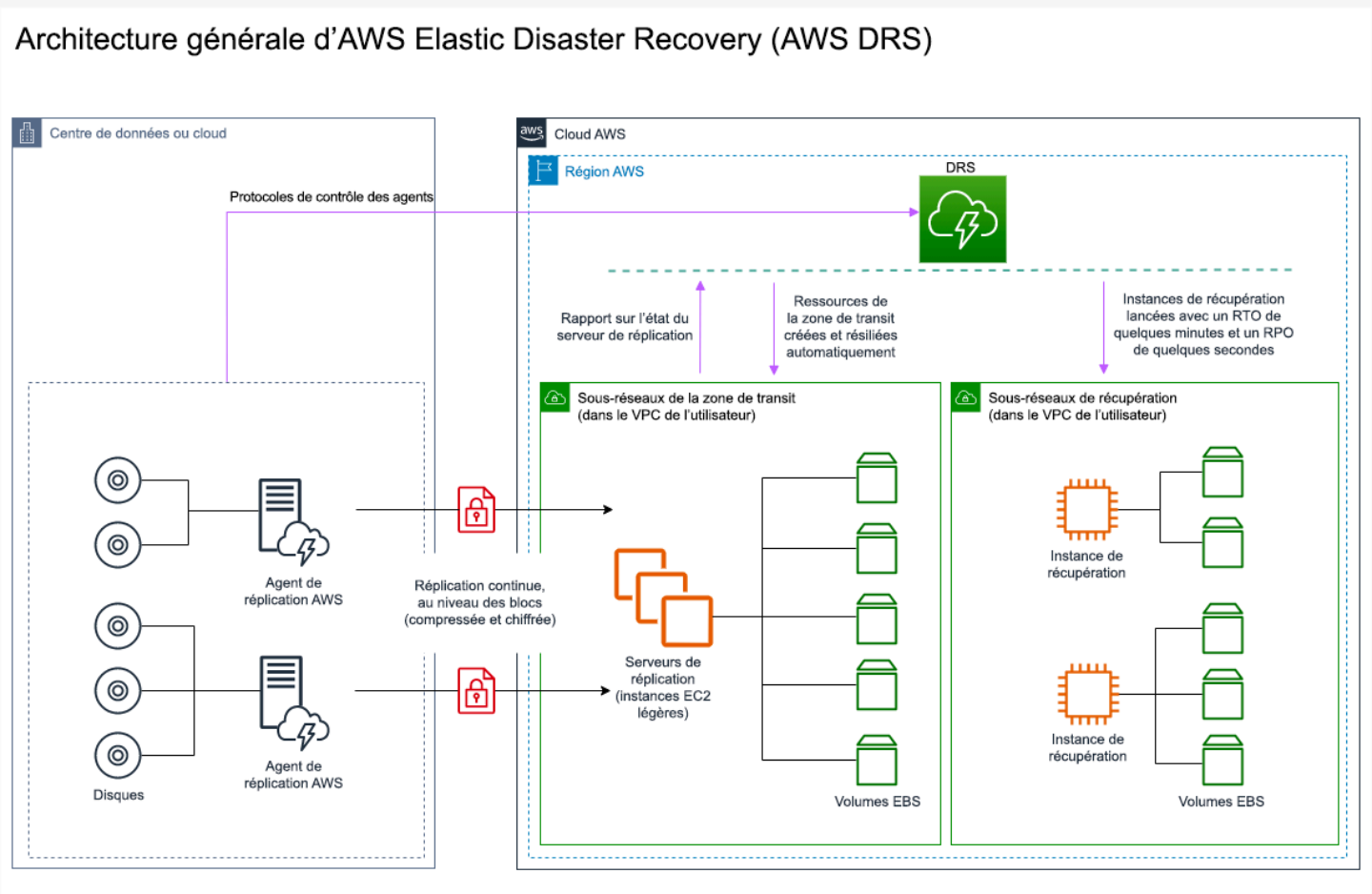


Figure 23 : architecture AWS Elastic Disaster Recovery

Pratiques supplémentaires de protection des données

Avec toutes les stratégies, vous devez également vous prémunir contre les catastrophes liées aux données La réplication continue des données vous protège contre certains types de sinistres, mais ne vous protège pas toujours contre la corruption ou la destruction des données, à moins que votre stratégie n'inclue également la gestion des versions des données stockées ou des options de récupération ponctuelle. Vous devez également sauvegarder les données répliquées sur le site de reprise pour créer des sauvegardes ponctuelles en plus des réplicas.

Utilisation de plusieurs zones de disponibilité (AZ) dans une seule Région AWS

Lorsque vous utilisez plusieurs AZ dans une même région, l'implémentation de la reprise après sinistre exploite plusieurs éléments des stratégies ci-dessus. Vous devez d'abord créer une architecture haute disponibilité (HA), en utilisant plusieurs AZ, comme illustré à la figure 23. Cette

architecture utilise une approche multisite actif/actif, car les instances Amazon EC2 et l' Elastic Load Balancer disposent de ressources déployées dans plusieurs zones de disponibilité, qui gèrent activement les requêtes. L'architecture présente également un système de zone hébergée qui permet, en cas de panne de l'instance principale Amazon RDS (ou de la zone de disponibilité elle-même), de faire passer l'instance de secours au rang d'instance principale.

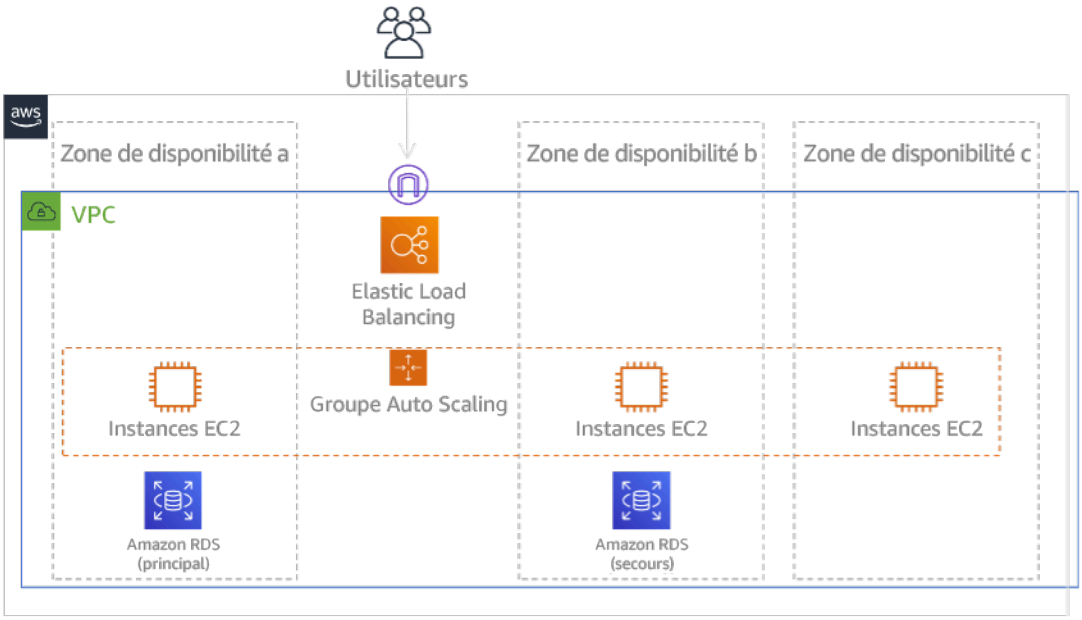


Figure 24 : architecture de multi-AZ

En plus de cette architecture haute disponibilité, vous devez ajouter des sauvegardes de toutes les données requises pour exécuter votre charge de travail. Ceci est particulièrement important pour les données limitées à une seule zone, telles que les Amazon EBS volumes ou les Amazon Redshift clusters. Si une zone de disponibilité tombe en panne, vous devrez restaurer ces données dans une autre zone de disponibilité. Dans la mesure du possible, vous devez également copier les sauvegardes de données dans une autre Région AWS comme couche de protection supplémentaire.

Une approche alternative moins courante de la reprise après sinistre à région unique et multi-AZ est illustrée dans l'article de blog, Building highly resilient applications using Amazon Route 53 Application Recovery Controller, Part 1: Single-Region stack (Création d'applications hautement résilientes à l'aide du contrôleur de récupération d'application d'Amazon Route 53, partie 1 : pile à région unique). Dans ce cas, la stratégie consiste à maintenir autant que possible l'isolement entre les zones de disponibilité, à l'instar du fonctionnement des régions. Avec cette stratégie alternative, vous pouvez choisir une approche active/active ou active/passive.

> **Note**
>
> Certaines charges de travail sont soumises à des exigences réglementaires en matière de situation géographique des données. Si cela s'applique à votre charge de travail dans une localité qui n'a actuellement qu'une seule Région AWS, plusieurs régions ne répondront pas aux besoins de votre entreprise. Les stratégies multi-AZ assurent une bonne protection contre la plupart des catastrophes.

3. Évaluez les ressources de votre charge de travail et déterminez quelle sera leur configuration dans la région de reprise avant le basculement (pendant le fonctionnement normal).

Pour l'infrastructure et les ressources AWS, utilisez l'infrastructure en tant que code telle que AWS CloudFormation ou des outils tiers comme Hashicorp Terraform. Pour un déploiement sur plusieurs comptes et régions en une seule opération, vous pouvez utiliser AWS CloudFormation StackSets. Pour les stratégies « Multisite actif/actif » et « Zone hébergée », l'infrastructure déployée dans la région de reprise dispose des mêmes ressources que la région principale. Pour les stratégies « Environnement en veille » et « Secours à chaud », l'infrastructure déployée nécessitera des actions supplémentaires pour être prête pour la production. En utilisant les paramètres de CloudFormation et la logique conditionnelle , vous pouvez contrôler si une pile déployée est active ou en veille avec un seul modèle. En utilisant Elastic Disaster Recovery, le service répliquera et orchestrera la restauration des configurations d'applications et des ressources informatiques.

Toutes les stratégies de reprise après sinistre exigent que les sources de données soient sauvegardées dans la Région AWS, puis que ces sauvegardes soient copiées dans la région de reprise. AWS Backup fournit une vue centralisée où vous pouvez configurer, planifier et surveiller les sauvegardes de ces ressources. Pour les stratégies « Environnement en veille », « Secours à chaud » et « Multisite actif/actif », vous devez également répliquer les données de la région principale vers les ressources de données de la région de reprise, telles que des instances de base de données Amazon Relational Database Service (Amazon RDS) ou des tables Amazon DynamoDB. Ces ressources de données sont donc actives et prêtes à répondre aux demandes dans la région de reprise.

Pour en savoir plus sur comment fonctionnent les services AWS dans les régions, consultez cette série de blogs intitulée Creating a Multi-Region Application with AWS Services (Création d'une application multirégion avec les services AWS).

4. Déterminez et mettez en œuvre la manière dont vous préparerez votre région de reprise pour le basculement en cas de besoin (lors d'un sinistre).

Pour la stratégie multisite actif/actif, le basculement consiste à évacuer une région et à s'appuyer sur les régions actives restantes. En général, ces régions sont prêtes à accepter du trafic. Pour les stratégies Environnement en veille et Secours à chaud, vos actions de reprise devront déployer les ressources manquantes, telles que les instances EC2 de la figure 20, ainsi que toute autre ressource manquante.

Pour toutes les stratégies ci-dessus, vous devrez peut-être promouvoir les instances en lecture seule des bases de données au rang d'instances principales en lecture/écriture.

Pour la sauvegarde et la restauration, la restauration des données à partir de la sauvegarde crée des ressources pour ces données, telles que des volumes EBS, des instances de base de données RDS et des tables DynamoDB. Vous devez également restaurer l'infrastructure et déployer le code. Vous pouvez utiliser AWS Backup pour restaurer les données dans la région de reprise. Consulter REL09-BP01 Identifier et sauvegarder toutes les données qui doivent être sauvegardées, ou reproduire les données à partir de sources pour en savoir plus. La reconstruction de l'infrastructure comprend la création de ressources telles que les instances EC2, en plus des Amazon Virtual Private Cloud(Amazon VPC), sous-réseaux et groupes de sécurité nécessaires. Vous pouvez automatiser une grande partie du processus de restauration. Pour savoir comment procéder, consultez cet article de blog.

5. Déterminez et mettez en œuvre la manière dont vous redirigerez le trafic vers le basculement en cas de besoin (lors d'un sinistre).

Cette opération de basculement peut être lancée automatiquement ou manuellement. Le basculement lancé automatiquement sur la base de vérifications de l'état ou d'alarmes doit être utilisé avec prudence, car un basculement inutile (fausse alerte) entraînerait des coûts tels que l'indisponibilité et la perte de données. Le basculement manuel est donc souvent utilisé. Dans ce cas, nous vous conseillons tout de même d'automatiser les étapes de basculement, de sorte que vous n'ayez à appuyer que sur un bouton pour lancer le basculement.

Il existe plusieurs options de gestion du trafic à prendre en compte lors de l'utilisation des services AWS. Une option consiste à utiliser Amazon Route 53. Avec Amazon Route 53, vous pouvez associer plusieurs points de terminaison IP dans une ou plusieurs Régions AWS avec un nom de domaine Route 53. Pour mettre en œuvre un basculement manuel, vous pouvez utiliser le Contrôleur de récupération d'application Amazon Route 53, qui fournit une API de plan de données

hautement disponible pour réacheminer le trafic vers la région de reprise. Lors de la mise en œuvre du basculement, utilisez les opérations du plan de données et évitez celles du plan de contrôle, comme décrit dans REL11-BP04 S'appuyer sur le plan de données et non sur le plan de contrôle pendant la récupération. »

Pour en savoir plus sur cette option et d'autres, consultez cette section du livre blanc sur la reprise après sinistre.

6. Élaborez un plan pour déterminer la façon dont votre charge de travail se rétablira.

La restauration consiste à renvoyer l'exploitation de la charge de travail à la région principale, après qu'un événement de sinistre s'est atténué. La mise en service de l'infrastructure et du code dans la région principale suit généralement les mêmes étapes que celles utilisées initialement. Elle s'appuie notamment sur l'infrastructure en tant que code et les pipelines de déploiement de code. Le défi posé par la restauration consiste à restaurer les magasins de données et à garantir leur cohérence avec la région de reprise en cours d'exécution.

Lors de l'état de basculement, les bases de données de la région de reprise sont actives et disposent des données à jour. L'objectif est alors de resynchroniser les données de la région de reprise vers la région principale, en s'assurant qu'elle est à jour.

Certains services AWS effectuent cette opération automatiquement. Si vous utilisiez les tables globales Amazon DynamoDB, même si la table de la région principale devenait indisponible, DynamoDB reprendrait la propagation de toutes les écritures en attente lorsqu'elle se reconnecterait. Si vous utilisez Amazon Aurora Global Database et un basculement planifié géré, la topologie de réplication existante de la base de données globale Aurora est maintenue. Par conséquent, l'ancienne instance en lecture/écriture de la région principale deviendra un réplica et recevra les mises à jour de la région de reprise.

Dans les cas où cela n'est pas automatique, vous devrez rétablir la base de données dans la région principale en tant que réplica de la base de données dans la région de reprise. Dans de nombreux cas, cela implique la suppression de l'ancienne base de données principale et la création de nouveaux réplicas. Par exemple, pour obtenir des instructions sur la marche à suivre avec Amazon Aurora Global Database, en supposant un basculement non planifié, consultez cet atelier : Fail Back a Global Database (Failback d'une base de données globale).

Après un basculement, si vous pouvez poursuivre l'exécution dans la région de reprise, envisagez d'en faire la nouvelle région principale. Vous devriez alors suivre toutes les étapes ci-dessus pour convertir l'ancienne région principale en région de reprise. Certaines organisations effectuent une

rotation planifiée, en échangeant périodiquement leurs régions principale et de reprise (par exemple tous les trois mois).

Toutes les étapes nécessaires au basculement et au rétablissement doivent être conservées dans un playbook accessible à tous les membres de l'équipe et révisé périodiquement.

En utilisant Elastic Disaster Recovery, le service aidera à orchestrer et à automatiser le processus de failback. Pour obtenir plus de détails, consultez Effectuer un failback.

Niveau d'effort du plan d'implémentation : élevé

Ressources

Bonnes pratiques associées :

- the section called “REL09-BP01 Identifier et sauvegarder toutes les données qui doivent être sauvegardées, ou reproduire les données à partir de sources”
- the section called “REL11-BP04 S'appuyer sur le plan de données et non sur le plan de contrôle pendant la récupération”
- the section called “REL13-BP01 Définir les objectifs de reprise pour les temps d'arrêt et les pertes de données”

Documents connexes :

- Blog d'architecture AWS : série sur la reprise après sinistre
- Reprise après sinistre des charges de travail sur AWS : reprise dans le cloud (livre blanc AWS)
- Options de reprise après sinistre dans le cloud
- Créer une solution backend active-active sans serveur sur plusieurs régions en une heure
- Backend sans serveur sur plusieurs régions - rechargé
- RDS : réplication d'un réplica en lecture entre les régions
- Route 53 : configuration du basculement DNS
- S3 : réplication entre régions
- Qu'est-ce que AWS Backup ?
- What is Route 53 Application Recovery Controller? (Qu'est-ce que le contrôleur de récupération d'application d'Amazon Route 53 ?)
- AWS Elastic Disaster Recovery

- HashiCorp Terraform : Premiers pas - AWS
- Partenaire APN : partenaires pouvant faciliter la reprise après sinistre
- AWS Marketplace : produits pouvant être utilisés pour la reprise après sinistre

Vidéos connexes :

- Disaster Recovery of Workloads on AWS (Reprise après sinistre des charges de travail sur AWS)
- AWS re:Invent 2018: Architecture Patterns for Multi-Region Active-Active Applications (ARC209-R2)
- Démarrer avec AWS Elastic Disaster Recovery | Amazon Web Services

Exemples connexes :

- Atelier Well-Architected - Reprise après sinistre - Série d'ateliers illustrant les stratégies de reprise après sinistre

REL13-BP03 Effectuer un test de validation de la mise en œuvre de la reprise après sinistre

Testez régulièrement le basculement vers votre site de reprise pour vérifier qu'il fonctionne correctement et que les RTO et RPO sont respectés.

Anti-modèles courants :

- Ne jamais exécuter de basculements en production.

Avantages liés au respect de cette bonne pratique : en testant régulièrement votre plan de reprise après sinistre, vous vérifiez qu'il fonctionnera en cas de besoin et que votre équipe sait comment exécuter la stratégie.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : élevé

Directives d'implémentation

S'il y a bien un modèle à éviter, c'est celui qui consiste à développer des chemins de récupération rarement testés. Par exemple, vous pouvez avoir un magasin de données secondaire qui est utilisé

pour les requêtes en lecture seule. Lorsque vous écrivez dans un magasin de données et que l'instance principale connaît une défaillance, vous pouvez basculer vers le magasin de données secondaire. Si vous ne testez pas fréquemment ce basculement, vous constaterez peut-être que vos hypothèses sur les capacités du magasin de données secondaire sont incorrectes. La capacité du magasin de données secondaire, qui peut avoir été suffisante lors de votre dernier test, peut ne plus être en mesure de tolérer la charge dans le cadre de ce scénario. Notre expérience nous a montré que seul un chemin de récupération après erreur testé fréquemment fonctionne réellement. C'est pourquoi l'idéal est de n'avoir qu'un petit nombre de chemins de récupération. Vous pouvez établir des modèles de reprise et tester ceux-ci régulièrement. Si vous avez un chemin de récupération complexe ou critique, vous devez toujours exécuter régulièrement cette panne en production pour vous assurer du bon fonctionnement de ce chemin de récupération. Dans l'exemple que nous venons de présenter, vous devez procéder régulièrement au basculement vers l'instance de secours, quel que soit le besoin.

Étapes d'implémentation

1. Préparez vos charges de travail pour la reprise. Testez régulièrement vos chemins de récupération. L'informatique orientée récupération identifie les caractéristiques des systèmes qui améliorent la récupération : isolement et redondance, capacité de l'ensemble du système à réduire les modifications, capacité à surveiller et déterminer l'état de santé, capacité à fournir des diagnostics, reprise automatique, conception modulaire et capacité à redémarrer. Entraînez votre chemin de reprise pour vérifier qu'il peut s'effectuer au moment et à l'état spécifiés. Utilisez vos runbooks au cours de cette reprise pour documenter les problèmes et trouver des solutions pour les résoudre avant le prochain test.
2. Pour les charges de travail basées sur Amazon EC2, utilisez AWS Elastic Disaster Recovery pour mettre en œuvre et lancer des instances d'opérations dans le cadre de votre stratégie de reprise après sinistre. AWS Elastic Disaster Recovery permet d'exécuter efficacement des opérations, ce qui vous aide à vous préparer à un basculement. Vous pouvez également lancer fréquemment vos instances en utilisant Elastic Disaster Recovery à des fins de test et d'opération sans rediriger le trafic.

Ressources

Documents connexes :

- Partenaire APN : partenaires pouvant faciliter la reprise après sinistre
- Blog d'architecture AWS : série sur la reprise après sinistre

- AWS Marketplace : produits pouvant être utilisés pour la reprise après sinistre
- AWS Elastic Disaster Recovery
- Reprise après sinistre des charges de travail sur AWS : reprise dans le cloud (livre blanc AWS)
- AWS Elastic Disaster Recovery Preparing for Failover (Préparation au basculement : préparation au basculement)
- Projet informatique orientée reprise Berkeley/Stanford
- Qu'est qu'AWS Fault Injection Simulator (AWS FIS) ?

Vidéos connexes :

- AWS re:Invent 2018: Architecture Patterns for Multi-Region Active-Active Applications
- AWS re:Invent 2019: Backup-and-restore and disaster-recovery solutions with AWS

Exemples connexes :

- Atelier Well-Architected : tester la résilience

REL13-BP04 Gérer l'écart de configuration au niveau du site ou de la région de reprise après sinistre

Assurez-vous que l'infrastructure, les données et la configuration sont conformes aux besoins du site ou de la région de reprise après sinistre. Par exemple, vérifiez que les AMI et les quotas de service sont à jour.

AWS Config surveille et enregistre en permanence les configurations de vos ressources AWS. Il peut détecter tout écart et déclencher AWS Systems Manager Automation pour le corriger et activer les alarmes. AWS CloudFormation peut également détecter tout écart dans les piles que vous avez déployées.

Anti-modèles courants :

- Ne pas effectuer les mises à jour dans vos emplacements de récupération, lorsque vous apportez des modifications à la configuration ou à l'infrastructure sur les emplacements principaux.
- Ne pas prendre en compte des limitations potentielles (comme les différences de service) sur le site principal et le site de reprise.

Avantages liés au respect de cette bonne pratique : Pour une reprise complète, veillez à ce que votre environnement de reprise après sinistre soit cohérent avec votre environnement existant.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

- Assurez-vous que vos pipelines de diffusion assurent effectivement cette diffusion au niveau de votre site principal ainsi qu'au niveau de vos sites de sauvegarde. Les pipelines de diffusion pour le déploiement d'applications en production doivent être distribués à tous les emplacements spécifiés de la stratégie de DR, y compris les environnements de développement et de test.
- Activez AWS Config pour suivre les écart potentiels au niveau des emplacements. Utilisez les règles AWS Config pour créer des systèmes qui appliquent vos stratégies de reprise après sinistre et génèrent des alertes lorsqu'elles détectent un écart.
  - Correction des ressources AWS non conformes à l'aide des règles AWS Config Rules
  - AWS Systems Manager Automation
- Utilisez AWS CloudFormation pour déployer votre infrastructure. AWS CloudFormation peut détecter l'écart entre ce que vos modèles CloudFormation spécifient et ce qui est réellement déployé.
  - AWS CloudFormation : détection de tout écart à l'échelle d'une pile CloudFormation

Ressources

Documents connexes :

- Partenaire APN : partenaires pouvant faciliter la reprise après sinistre
- Blog d'architecture AWS : série sur la reprise après sinistre
- AWS CloudFormation : détection de tout écart à l'échelle d'une pile CloudFormation
- AWS Marketplace : produits pouvant être utilisés pour la reprise après sinistre
- AWS Systems Manager Automation
- Reprise après sinistre des charges de travail sur AWS : reprise dans le cloud (livre blanc AWS)
- Comment mettre en œuvre une solution de gestion de configuration d'infrastructure sur AWS ?
- Correction des ressources AWS non conformes à l'aide des règles AWS Config Rules

Vidéos connexes :

- AWS re:Invent 2018: Architecture Patterns for Multi-Region Active-Active Applications (ARC209-R2)

REL13-BP05 Automatiser la reprise

Utilisez AWS ou des outils tiers pour automatiser la reprise du système et acheminer le trafic vers le site ou la région de reprise après sinistre.

En fonction des vérifications de l'état configurées, les services AWS tels qu'Elastic Load Balancing et AWS Auto Scaling peuvent répartir la charge vers des zones de disponibilité saines, tandis que les services tels qu'AWS et Global Accelerator peuvent acheminer la charge vers des Régions AWS saines. Amazon Route 53 Application Recovery Controller vous aide à gérer et à coordonner le basculement à l'aide de vérifications de l'état de préparation et fonctionnalités de contrôle du routage. Ces fonctionnalités surveillent en permanence la capacité de votre application à se rétablir après une défaillance et vous permettent de contrôler la reprise de votre application dans plusieurs Régions AWS, zones de disponibilité et sur site.

Pour les charges de travail sur des centres de données physiques ou virtuels existants ou des clouds privés, AWS Elastic Disaster Recoverydisponible via AWS Marketplace, permet aux organisations de configurer une stratégie de reprise après sinistre automatisée pour AWS. CloudEndure prend également en charge la reprise après sinistre entre régions et zones de disponibilité dans AWS.

Anti-modèles courants :

- La mise en œuvre d'un système de basculement et de restauration automatisés identique peut entraîner une oscillation de chemin lorsqu'une défaillance se produit.

Avantages liés au respect de cette bonne pratique : La reprise automatique réduit le temps de reprise en éliminant les risques d'erreurs manuelles.

Niveau de risque exposé si cette bonne pratique n'est pas respectée : Moyenne entreprise

Directives d'implémentation

- Automatisez les chemins de récupération. Pour les temps de reprise courts, le jugement et les actions de l'humain ne peuvent pas être utilisés pour des scénarios à haute disponibilité. Le système doit absolument reprendre automatiquement, quelle que soit la situation.
  - Utilisez CloudEndure Disaster Recovery pour un basculement et une restauration automatisés. CloudEndure Disaster Recovery réplique en continu vos machines (notamment le système

d'exploitation, la configuration d'état du système, les bases de données, les applications et les fichiers) dans une zone intermédiaire économique de votre Compte AWS cible et de votre région préférée. En cas de sinistre, vous pouvez demander à CloudEndure Disaster Recovery de lancer automatiquement des milliers de vos machines dans leur état entièrement mis en service en quelques minutes.

- Exécution d'un basculement et d'une reprise après sinistre
- CloudEndure Disaster Recovery

Ressources

Documents connexes :

- Partenaire APN : partenaires pouvant faciliter la reprise après sinistre
- Blog d'architecture AWS : série sur la reprise après sinistre
- AWS Marketplace : produits pouvant être utilisés pour la reprise après sinistre
- AWS Systems Manager Automation
- CloudEndure Disaster Recovery vers AWS
- Reprise après sinistre des charges de travail sur AWS : reprise dans le cloud (livre blanc AWS)

Vidéos connexes :

- AWS re:Invent 2018: Architecture Patterns for Multi-Region Active-Active Applications (ARC209-R2)

Exemples de mises en œuvre pour les objectifs de disponibilité

Dans cette section, nous examinerons les conceptions de la charge de travail en utilisant le déploiement d'une application Web classique qui se compose d'un proxy inverse, de contenu statique sur Amazon S3, d'un serveur d'application et d'une base de données SQL pour le stockage permanent des données. Pour chaque cible de disponibilité, nous fournissons un exemple de mise en œuvre. Cette charge de travail pourrait plutôt utiliser des conteneurs ou AWS Lambda pour le calcul et NoSQL (comme Amazon DynamoDB) pour la base de données, mais les approches sont similaires. Dans chaque scénario, nous montrons comment atteindre les objectifs de disponibilité grâce à la conception de la charge de travail pour ces rubriques :

Rubrique	Pour plus d'informations, consultez cette section
Surveiller les ressources	Surveiller les ressources de charge de travail
s'adapter aux modifications de la demande ;	Concevoir votre charge de travail de sorte qu'elle s'adapte aux changements de demande
Implémentation de la modification	Implémentation de la modification
sauvegarder les données ;	sauvegarder les données ;
Conception de la résilience	Utilisation de l'isolation des défaillances pour protéger votre charge de travail Conception d'une charge de travail qui résiste aux défaillances des composants
tester la résilience ;	Test de la fiabilité
Planification de la reprise après sinistre	Planification de la reprise après sinistre

Sélection des dépendances

Nous avons choisi d'utiliser Amazon EC2 pour nos applications. Nous montrerons comment l'utilisation d'Amazon RDS et de plusieurs zones de disponibilité améliore la disponibilité de nos applications. Nous utiliserons Amazon Route 53 pour DNS. Pour utiliser plusieurs zones de disponibilité, nous utilisons la fonction Elastic Load Balancing. Amazon S3 est utilisé pour les sauvegardes et le contenu statique. Comme nous concevons dans le but d'une meilleure fiabilité, nous devons utiliser les services ayant eux-mêmes une plus grande disponibilité.

Scénarios à région unique

Rubriques

Scénario à deux 9 (99 %)

Ces charges de travail sont utiles pour l'entreprise, mais n'entraînent qu'un contretemps si elles ne sont pas disponibles. Ce type de charge de travail comprend notamment les outils internes, la gestion des connaissances internes ou le suivi de projets. Il peut également s'agir de charges de travail réelles orientées client, mais exécutées à partir d'un service expérimental, avec une fonction d'activation/désactivation qui peut masquer le service si nécessaire.

Ces charges de travail peuvent être déployées avec une région et une zone de disponibilité.

Surveiller les ressources

Nous avons une surveillance simple, indiquant si la page d'accueil du service affiche un statut HTTP 200 OK. Lorsque des problèmes se produisent, notre playbook indique que les journaux de l'instance seront utilisés pour établir la cause racine.

s'adapter aux modifications de la demande ;

Nous disposons de playbooks pour les défaillances matérielles courantes, les mises à jour logicielles urgentes et d'autres modifications perturbatrices.

Implémentation de la modification

Nous utilisons AWS CloudFormation pour définir notre infrastructure en tant que code, et spécifiquement pour accélérer la reconstruction en cas de panne.

Les mises à jour logicielles sont réalisées manuellement à l'aide d'un runbook, avec les temps d'arrêt requis pour l'installation et le redémarrage du service. Si un problème se produit pendant le déploiement, le runbook décrit comment restaurer la version précédente.

Toutes les corrections de l'erreur sont effectuées à l'aide de l'analyse des journaux par les équipes d'exploitation et de développement. La correction est ensuite déployée une fois le correctif hiérarchisé et terminé.

sauvegarder les données ;

Nous utilisons une solution de sauvegarde d'un fournisseur ou spécialisée pour envoyer des données de sauvegarde chiffrées dans Amazon S3 à l'aide d'un runbook. Nous vérifions que les sauvegardes fonctionnent en restaurant les données et en vérifiant qu'il est possible de les utiliser régulièrement à l'aide d'un runbook. Nous configurons la gestion des versions sur nos objets Amazon S3 et supprimons les autorisations de suppression des sauvegardes. Nous suivons une politique de cycle de vie de compartiment Amazon S3 pour archiver ou supprimer définitivement en fonction de nos besoins.

Conception de la résilience

Les charges de travail sont déployées avec une région et une zone de disponibilité. Nous déployons l’application, y compris la base de données, dans une seule instance.

tester la résilience ;

Le pipeline de déploiement d'un nouveau logiciel est planifié, avec des tests d'unité, mais principalement des essais de boîte blanche/boîte noire de l'ensemble de la charge de travail.

Planification de la reprise après sinistre

En cas de panne, nous en attendons la fin en acheminant éventuellement les requêtes vers un site Web statique grâce à la modification DNS via un runbook. Le temps de récupération pour cet événement sera déterminé par la vitesse à laquelle l'infrastructure peut être déployée et la base de données restaurée à la dernière sauvegarde. Ce déploiement peut se faire dans la même zone de

disponibilité ou dans une autre zone de disponibilité en cas de défaillance d'une zone de disponibilité à l'aide d'un runbook.

Objectif de la conception de la disponibilité

Nous prenons 30 minutes pour comprendre et décider d'exécuter la récupération, et 10 minutes pour déployer la pile entière dans AWS CloudFormation. Nous supposons que nous déployons sur une nouvelle zone de disponibilité et que la base de données peut être restaurée en 30 minutes. Cela signifie qu'il faut environ 70 minutes pour reprendre les activités après une défaillance. En supposant une défaillance par trimestre, notre impact estimé pour l'année est 280 minutes ou quatre heures et 40 minutes.

Cela signifie que la limite supérieure de disponibilité est de 99,9 %. La disponibilité réelle varie également selon le taux réel de défaillance, la durée des défaillances et la rapidité de récupération réelle après chaque panne. Pour cette architecture, nous avons besoin que l'application soit hors ligne pour les mises à jour (en estimant 24 heures par an : quatre heures par modification, six fois par an), ainsi que les événements réels. Ainsi, en nous reportant au tableau de disponibilité des applications présenté plus haut dans le livre blanc, nous voyons que notre objectif de la conception de la disponibilité est de 99 %.

Récapitulatif

Rubrique	Implémentation
Surveiller les ressources	Vérification de l'état du site uniquement ; aucune alerte.
s'adapter aux modifications de la demande ;	Mise à l'échelle verticale via redéploiement.
Implémentation de la modification	Runbook pour le déploiement et la restauration.
sauvegarder les données ;	Runbook pour la sauvegarde et la restauration.
Conception de la résilience	Reconstruction ; restauration à partir d'une sauvegarde.
tester la résilience ;	Reconstruction ; restauration à partir d'une sauvegarde.

Rubrique	Implémentation
Planification de la reprise après sinistre	Sauvegardes chiffrées, restauration dans une autre zone de disponibilité si nécessaire.

Scénario à trois 9 (99,9 %)

Le prochain objectif de disponibilité concerne les applications qui doivent être hautement disponibles, mais pour lesquelles il est possible de tolérer de courtes périodes d'indisponibilité. Ce type de charge de travail est généralement utilisé pour les opérations internes dont les pannes affectent les employés. Ce type de charge de travail peut également être orienté client sans générer de revenus commerciaux importants et peut tolérer un temps ou un point de récupération plus long. Ces charges de travail incluent des applications administratives pour la gestion des comptes ou des informations.

Nous pouvons améliorer la disponibilité des charges de travail en utilisant deux zones de disponibilité pour le déploiement et en séparant les applications à des niveaux distincts.

Surveiller les ressources

La surveillance est étendue pour émettre des alertes sur la disponibilité globale du site Web en recherchant un statut HTTP 200 OK sur la page d'accueil. Il y a en outre des alertes à chaque remplacement d'un serveur Web et lors du basculement de la base de données. Nous surveillons également la disponibilité du contenu statique sur Amazon S3 et envoyons une alerte en cas d'indisponibilité. Les logs sont regroupés pour faciliter la gestion et contribuer à l'analyse des causes racines.

s'adapter aux modifications de la demande ;

La mise à l'échelle automatique est configurée pour surveiller l'utilisation du CPU sur les instances EC2 et ajouter ou supprimer des instances pour maintenir la cible du CPU à 70 %, mais avec au moins une instance EC2 par zone de disponibilité. Si des modèles de charge sur notre instance RDS indiquent qu'un scaling-up est nécessaire, nous modifierons le type d'instance pendant une fenêtre de maintenance.

Implémentation de la modification

Les technologies de déploiement d'infrastructures restent identiques au scénario précédent.

La fourniture des nouveaux logiciels se fait selon un calendrier fixe, toutes les deux à quatre semaines. Les mises à jour logicielles sont automatisées, non pas avec des modèles de déploiement Canary ou bleu/vert, mais à l'aide de la fonction de remplacement sur place. La décision de restauration est prise à l'aide des runbooks.

Nous disposerons de playbooks pour déterminer la cause racine des problèmes. Après l'identification de la cause racine, une association des équipes Opérations et Développement identifie la correction de l'erreur. La correction est déployée une fois le correctif développé.

sauvegarder les données ;

La sauvegarde et la restauration peuvent être effectuées à l'aide d'Amazon RDS. Elles seront exécutées régulièrement à l'aide d'un runbook afin de garantir que nous pouvons répondre aux exigences de récupération.

Conception de la résilience

Nous pouvons améliorer la disponibilité des applications en utilisant deux zones de disponibilité pour le déploiement et en séparant les applications à des niveaux distincts. Nous allons utiliser des services qui fonctionnent dans plusieurs zones de disponibilité, commeElastic Load Balancing, Auto Scaling et Amazon RDS multi-AZ avec un stockage chiffré via AWS Key Management Service. Cela garantit la tolérance aux pannes au niveau des ressources et de la zone de disponibilité.

L'équilibreur de charge achemine uniquement le trafic vers les instances d'application saines. La vérification de l'état doit se faire au niveau de la couche d'application/du plan de données indiquant la capacité de l'application sur l'instance. Cette vérification ne doit pas être effectuée sur le plan de contrôle. Une URL de vérification de l'état de l'application Web est présente et configurée pour être utilisée par l'équilibreur de charge et Auto Scaling, afin que les instances défaillantes soient supprimées et remplacées. Amazon RDS gère le moteur de base de données actif afin qu'il soit disponible dans la seconde zone de disponibilité si l'instance échoue dans la zone de disponibilité principale, puis effectue les réparations pour restaurer la même résilience.

Une fois que nous avons séparé les niveaux, nous pouvons utiliser les modèles de résilience des systèmes distribués pour augmenter la fiabilité de l'application afin qu'elle puisse toujours être disponible, même quand la base de données est temporairement indisponible pendant le basculement d'une zone de disponibilité.

tester la résilience ;

Nous réalisons des tests fonctionnels, comme dans le scénario précédent. Nous ne testons pas les capacités d'auto-réparation d'ELB, d'Auto Scaling ou de basculement RDS.

Nous avons des playbooks pour les problèmes de base de données courants, les incidents liés à la sécurité et les échecs de déploiement.

Planification de la reprise après sinistre

Les runbooks sont destinés à la récupération totale des charges de travail et aux rapports courants. La récupération utilise les sauvegardes stockées dans la même région que la charge de travail.

Objectif de la conception de la disponibilité

Nous supposons que certaines défaillances nécessitent une prise de décision manuelle d'exécuter la récupération. Cependant, dans ce scénario comportant une meilleure automatisation, nous supposons que seulement deux événements par an exigeront cette décision. Nous prenons 30 minutes à décider d'exécuter la récupération, et supposons que la restauration est terminée dans un délai de 30 minutes. Cela signifie que 60 minutes sont nécessaires pour reprendre les activités après un incident. En supposant deux incidents par an, notre impact estimé pour l'année est de 120 minutes.

Cela signifie que la limite supérieure de disponibilité est de 99,95 %. La disponibilité réelle varie également selon le taux réel de défaillance, la durée des défaillances et la rapidité à laquelle chaque facteur récupère réellement. Pour cette architecture, nous avons besoin que l'application soit brièvement hors ligne pour les mises à jour, mais ces mises à jour sont automatisées. Nous estimons que cela prendra 150 minutes par an : 15 minutes par modification, 10 fois par an. Cela atteint un total de 270 minutes par an lorsque le service n'est pas disponible et, par conséquent, notre objectif de la conception de la disponibilité est de 99,9 %.

Récapitulatif

Rubrique	Implémentation
Surveiller les ressources	Vérification de l'état du site uniquement ; alertes envoyées en cas d'arrêt.

Rubrique	Implémentation
s'adapter aux modifications de la demande ;	ELB pour le niveau d'application de la mise à l'échelle Web et automatique ; redimensionnement RDS multi-AZ.
Implémentation de la modification	Déploiement automatisé sur place et runbook pour la restauration.
sauvegarder les données ;	Sauvegardes automatisées via RDS pour respecter le RPO et le runbook pour la restauration.
Conception de la résilience	Mise à l'échelle automatique pour assurer l'auto-régénération des niveaux Web et application ; RDS multi-AZ.
tester la résilience ;	Réparation automatique d'ELB et de l'application ; RDS multi-AZ ; aucun test explicite.
Planification de la reprise après sinistre	Sauvegardes chiffrées via RDS vers la même région AWS.

Scénario à quatre 9 (99,99 %)

Cet objectif de disponibilité pour les applications nécessite que l'application soit hautement disponible et tolère les pannes de composants. L'application doit être en mesure d'absorber les défaillances sans avoir besoin d'obtenir des ressources supplémentaires. Cet objectif de disponibilité concerne les applications critiques qui représentent des sources de revenus primaires ou considérables pour une entreprise, par exemple un site de commerce électronique, un service Web interentreprises, ou un site de médias/contenu à trafic dense.

Nous pouvons améliorer davantage la disponibilité en utilisant une architecture statiquement stable dans la région. Cet objectif de disponibilité n'exige pas de modification du plan de contrôle au niveau du comportement de notre charge de travail pour tolérer les défaillances. Par exemple, il devrait y avoir une capacité suffisante pour supporter la perte d'une zone de disponibilité. Nous ne devrions pas nécessiter de mises à jour du DNS Amazon Route 53. Nous ne devrions pas avoir

besoin de créer de nouvelles infrastructures, qu'il s'agisse de la création ou de la modification d'un compartiment S3, de la création de stratégies IAM (ou de la modification de politiques) ou de la modification des configurations de tâche Amazon ECS.

Surveiller les ressources

La surveillance inclut des indicateurs de réussite ainsi que des alertes en cas de problème. De plus, des alertes sont émises à chaque remplacement d'un serveur Web défectueux, lors du basculement de la base de données et lors de la défaillance d'une zone de disponibilité.

s'adapter aux modifications de la demande ;

Nous utiliserons Amazon Aurora comme RDS afin de bénéficier de la mise à l'échelle automatique des réplicas en lecture. Pour ces applications, l'ingénierie de la disponibilité du contenu principal en lecture sur la disponibilité en écriture représente également une décision essentielle concernant l'architecture. Aurora peut également augmenter automatiquement le stockage en fonction des besoins, par incréments de 10 Go jusqu'à 64 To.

Implémentation de la modification

Nous déployons les mises à jour en utilisant des modèles Canary ou bleu/vert dans chaque zone d'isolation séparément. Les déploiements sont entièrement automatisés, et permettent une restauration à un état antérieur si les indicateurs de performance clé indiquent un problème.

Les runbooks sont prévus pour les exigences de rapport strictes et le suivi des performances. Si des opérations fructueuses tendent vers un non-respect des objectifs de performance ou de disponibilité, un playbook est utilisé pour établir la cause de cette tendance. Les playbooks sont prévus pour les modes d'échec non découverts et les incidents de sécurité. Des playbooks sont également prévus pour établir la cause racine des défaillances. Nous collaborons également avec AWS Support pour l'offre Infrastructure Event Management.

L'équipe qui crée et exploite le site Web identifie la correction d'erreur de toute panne inattendue et hiérarchise le correctif à déployer une fois qu'il est mis en œuvre.

sauvegarder les données ;

La sauvegarde et la restauration peuvent être effectuées à l'aide d'Amazon RDS. Elles seront exécutées régulièrement à l'aide d'un runbook afin de garantir que nous pouvons répondre aux exigences de récupération.

Conception de la résilience

Nous recommandons trois zones de disponibilité pour cette approche. Grâce à un déploiement de trois zones de disponibilité, chaque zone de disponibilité possède une capacité statique maximale de 50 %. Il est possible d'utiliser deux zones de disponibilité, mais le coût de la capacité statiquement stable serait plus élevé, car les deux zones de disponibilité devraient avoir une capacité maximale de 100 %. Nous ajouterons Amazon CloudFront pour fournir une mise en cache géographique, ainsi que la réduction des requêtes sur le plan de données de notre application.

Nous utiliserons Amazon Aurora comme RDS et déploierons des réplicas en lecture dans les trois zones.

L'application est créée à l'aide des modèles de résilience de logiciel/application sur toutes les couches.

tester la résilience ;

Le pipeline de déploiement dispose d'une suite de tests complète, y compris des tests de performance, de charge et de provocation de pannes.

Les jeux de rôles nous permettent de nous entraîner constamment aux procédures de reprise après une défaillance, grâce aux runbooks qui nous permettent de nous assurer d'effectuer les tâches et de ne pas nous écarter des procédures. L'équipe qui crée le site Web exploite également ce dernier.

Planification de la reprise après sinistre

Les runbooks sont destinés à la récupération totale des charges de travail et aux rapports courants. La récupération utilise les sauvegardes stockées dans la même région que la charge de travail. Les procédures de restauration sont régulièrement utilisées dans le cadre des tests de simulation de pannes.

Objectif de la conception de la disponibilité

Nous supposons que certaines pannes nécessiteront une prise de décision manuelle pour exécuter la récupération, mais avec une plus grande automatisation dans ce scénario, nous supposons que seulement deux événements par an exigeront cette décision et que les actions de récupération seront rapides. Nous prenons 10 minutes à décider d'exécuter la récupération, et supposons que la restauration est terminée dans un délai de cinq minutes. Cela signifie que 15 minutes sont nécessaires pour reprendre les activités après un incident. En supposant deux défaillances par an, notre impact estimé pour l'année est de 30 minutes.

Cela signifie que la limite supérieure de disponibilité est de 99,99 %. La disponibilité réelle variera également selon le taux réel de défaillance, la durée des défaillances et la rapidité à laquelle chaque facteur récupère réellement. Pour cette architecture, nous supposons que l'application est en ligne continuellement par le biais de mises à jour. Par conséquent, nos objectif de la conception de la disponibilité est de 99,99 %.

Récapitulatif

Rubrique	Implémentation
Surveiller les ressources	Vérifications de l'état sur toutes les couches et sur les indicateurs de performance clés ; alertes envoyées au déclenchement des alarmes configurées ; alerte à la moindre panne. Les réunions opérationnelles sont rigoureuses pour détecter les tendances et gérer les objectifs de conception.
s'adapter aux modifications de la demande ;	ELB pour le niveau d'application de mise à l'échelle Web et automatique ; stockage de la mise à l'échelle automatique et réplicas en lecture dans plusieurs zones pour Aurora RDS.
Implémentation de la modification	Déploiement automatisé (Canary ou bleu/vert) et restauration automatique lorsque les indicateurs de performance clés ou les alertes indiquent des problèmes non détectés dans l'application. Les déploiements sont effectués par zone d'isolation.
sauvegarder les données ;	Sauvegardes automatisées via RDS pour respecter le RPO et la restauration automatique qui est appliquée régulièrement au cours d'un test de simulation de pannes.
Conception de la résilience	Implémentation de zones d'isolation des pannes pour l'application ; Auto Scaling pour

Rubrique	Implémentation
	fournir un niveau d'application et Web auto-régénérant ; RDS multi-AZ.
tester la résilience ;	Intégration du test des pannes des composant s et des zones d'isolation au pipeline et mise en œuvre régulière par le personnel opération nel au cours d'un test de simulation de pannes ; playbooks pour diagnostiquer les problèmes inconnus ; processus d'analyse des causes racines.
Planification de la reprise après sinistre	Sauvegardes chiffrées via RDS vers la même région AWS que celle impliquée dans un test de simulation de pannes.

Scénarios multirégion

La mise en œuvre de notre application dans plusieurs régions AWS augmente le coût d'exploitation, notamment parce que nous isolons les régions pour maintenir leur autonomie. La décision de poursuivre dans cette direction doit être mûrement réfléchie. Cela dit, les régions permettent une forte limite d'isolation et nous faisons beaucoup d'efforts pour éviter des défaillances en corrélation d'une région à une autre. L'utilisation de plusieurs régions vous permet de mieux contrôler votre temps de récupération en cas de défaillance de dépendance stricte sur un service AWS régional. Dans cette section, nous évoquons les divers modèles d'implémentation et leur disponibilité standard.

Rubriques

- 3½ 9 s (99,95 %) avec un temps de récupération compris entre 5 et 30 minutes
- Scénario à cinq 9 (99,999 %) ou supérieur avec un temps de récupération inférieur à 1 minute

3½ 9 s (99,95 %) avec un temps de récupération compris entre 5 et 30 minutes

Cet objectif de disponibilité pour les applications nécessite très peu de temps d'arrêt et de très faibles pertes de données pendant des périodes spécifiques. Les applications avec cet objectif

de disponibilité comprennent les applications dans les domaines suivants : services bancaires, investissements, services d'urgence et capture de données. Ces applications ont des temps de récupération et des points de récupération très courts.

Nous pouvons encore améliorer les temps de récupération en utilisant une Secours à chaud dans deux régions AWS. Nous déployons l'intégralité de la charge de travail pour les deux régions via notre site passif réduit et des données cohérentes à terme. Les deux déploiements seront statiquement stable au sein de leurs régions respectives. Les applications doivent être conçues à l'aide des modèles de résilience du système distribué. Nous allons devoir créer un composant de routage léger qui surveille à la fois l'état de l'application et toutes les dépendances régionales strictes que nous possédons.

Surveiller les ressources

Des alertes sont émises à chaque remplacement d'un serveur Web, lors du basculement de la base de données et du basculement de la région. Nous surveillons également la disponibilité du contenu statique sur Amazon S3 et envoyons une alerte en cas d'indisponibilité. Les logs sont regroupés pour faciliter la gestion et contribuer à l'analyse des causes racines dans chaque région.

Le composant de routage léger surveille à la fois l'état de l'application et toutes les dépendances régionales strictes que nous possédons.

s'adapter aux modifications de la demande ;

Identique au scénario 4 9s.

Implémentation de la modification

La fourniture des nouveaux logiciels se fait selon un calendrier fixe, toutes les deux à quatre semaines. Les mises à jour logicielles sont automatisées grâce aux modèles de déploiement Canary ou bleu/vert.

Les runbooks sont prévus pour le basculement de la région, pour les problèmes courants des clients qui se produisent au cours de ces événements, et pour les rapports courants.

Nous avons des playbooks pour les problèmes de base de données courants, les incidents liés à la sécurité, les échecs de déploiement, les problèmes inattendus des clients lors du basculement de la région et l'établissement de la cause racine des problèmes. Après l'identification de la cause racine, une association des équipes Opérations et Développement identifie la correction de l'erreur et la déploie après avoir développé le correctif.

Nous collaborons également avec AWS Support pour l'offre Infrastructure Event Management.

sauvegarder les données ;

À l'instar du scénario à quatre 9, nous réalisons des sauvegardes RDS automatiques et utilisons la gestion des versions S3. Les données sont automatiquement répliquées de manière asynchrone à partir du cluster Aurora RDS dans la région active vers des réplicas en lecture entre régions dans la région passive. La réplication entre régions S3 est utilisée pour déplacer automatiquement de manière asynchrone les données de la région active vers la région passive.

Conception de la résilience

Identique au scénario 4 9s, avec basculement régional possible. Cette opération est gérée manuellement. Pendant le basculement, nous acheminons les requêtes vers un site Web statique en utilisant le basculement DNS jusqu'à la récupération dans la seconde région.

tester la résilience ;

Identique au scénario 4 9s. Nous validerons également l'architecture au cours des tests de simulation des pannes à l'aide des runbooks. De plus, la correction RCA est prioritaire sur les versions de fonctions pour une implémentation et un déploiement immédiats

Planification de la reprise après sinistre

Le basculement régional est géré manuellement. Toutes les données sont répliquées de manière asynchrone. L'infrastructure de secours à chaud est mise à l'échelle. L'automatisation est possible via un flux de travail exécuté sur AWS Step Functions. AWS Systems Manager (SSM) facilite également cette automatisation, car vous pouvez créer des documents SSM qui mettent à jour les groupes Auto Scaling et redimensionnent les instances.

Objectif de la conception de la disponibilité

Nous supposons que certaines pannes nécessiteront une prise de décision manuelle pour exécuter la récupération, mais avec une bonne automatisation dans ce scénario, nous supposons que seulement deux événements par an exigeront cette décision. Nous prenons 20 minutes à décider d'exécuter la récupération, et supposons que la restauration est terminée dans un délai de 10 minutes. Cela signifie qu'il faut 30 minutes pour reprendre les activités après une défaillance. En supposant deux incidents par an, notre impact estimé pour l'année est de 60 minutes.

Cela signifie que la limite supérieure de disponibilité est de 99,95 %. La disponibilité réelle variera également selon le taux réel de défaillance, la durée des défaillances et la rapidité à laquelle chaque

facteur récupère réellement. Pour cette architecture, nous supposons que l'application est en ligne continuellement par le biais de mises à jour. Par conséquent, nos objectif de la conception de la disponibilité est de 99,95 %.

Récapitulatif

Rubrique	Implémentation
Contrôler les ressources	Vérifications de l'état de santé au niveau de toutes les couches, y compris l'état du DNS au niveau de la région AWS et au niveau des indicateurs clés de performance ; alertes envoyées au déclenchement des alarmes configurées ; alerte à la moindre panne. Les réunions opérationnelles sont rigoureuses pour détecter les tendances et gérer les objectifs de conception.
s'adapter aux modifications de la demande ;	ELB pour le niveau d'application de mise à l'échelle Web et automatique ; mise à l'échelle automatique du stockage et réplicas en lecture dans plusieurs zones dans les régions actives et passives pour Aurora RDS. Données et infrastructure synchronisées entre les régions AWS pour garantir la stabilité statique.
Implémentation de la modification	Déploiement automatisé (Canary ou bleu/vert) et restauration automatique lorsque les indicateurs clés de performance ou les alertes indiquent des problèmes non détectés dans l'application. Les déploiements sont effectués vers une zone d'isolement dans une seule région AWS à la fois.
sauvegarder les données ;	Sauvegardes automatisées dans chaque région AWS via RDS pour respecter le RPO et la restauration automatisée pratiquée régulièrement au cours d'un test de simulation des

Rubrique	Implémentation
	pannes. Les données Aurora RDS et S3 sont répliquées automatiquement de manière asynchrone de la région active vers la région passive.
Conception de la résilience	Mise à l'échelle automatique pour assurer l'auto-régénération au niveau Web et applicati on ; RDS multi-AZ ; basculement régional géré manuellement avec le site statique présenté lors du basculement.
tester la résilience ;	Intégration du test des pannes des composant s et des zones d'isolation au pipeline et mise en œuvre régulière par le personnel opération nel au cours d'un test de simulation de pannes ; playbooks pour diagnostiquer les problèmes inconnus ; processus d'analyse des causes racines avec acheminement des communica tions sur le type de problème, sa correction et sa prévention. La correction RCA est prioritai re sur les versions de fonctions pour une implémentation et un déploiement immédiats.
Planification de la reprise après sinistre	Secours à chaud déployé dans une autre région. L'infrastructure est mise à l'échelle au moyen de flux de travail exécutés via AWS Step Functions ou de documents d'AWS Systems Manager. Sauvegardes chiffrées via RDS. Réplicas en lecture entre régions entre deux régions AWS. Réplication entre régions des ressources statiques dans Amazon S3. La restauration s'adresse à la région AWS active actuelle. Elle est mise en pratique au cours d'un test de simulation des pannes et est coordonné e avec AWS.

Scénario à cinq 9 (99,999 %) ou supérieur avec un temps de récupération inférieur à 1 minute

Cet objectif de disponibilité pour les applications exige des temps d'arrêt et des pertes de données quasi-nuls pour les périodes spécifiques. Les applications qui pourraient avoir cet objectif de disponibilité comprennent, par exemple, certaines applications bancaires, d'investissement, financières, gouvernementales, et des applications commerciales critiques qui représentent l'activité principale d'une entreprise générant des revenus extrêmement élevés. L'objectif est d'atteindre des magasins de données ayant une cohérence forte et une redondance complète sur toutes les couches. Nous avons sélectionné un magasin de données SQL. Toutefois, dans certains scénarios, il est difficile d'atteindre un très petit RPO. Si vous pouvez partitionner les données, il est possible de n'avoir aucune perte de données. Pour cela, il peut être nécessaire d'ajouter la logique d'application et la latence afin de garantir la cohérence des données entre les emplacements géographiques, ainsi que la capacité de déplacer ou de copier des données entre les partitions. L'exécution de ce partitionnement peut être plus facile si vous utilisez une base de données NoSQL.

Nous pouvons améliorer davantage la disponibilité en utilisant une Actif-actif sur plusieurs régions AWS. La charge de travail sera déployée dans toutes les régions souhaitées qui sont statiquement stable parmi les régions (afin que les régions restantes puissent gérer la charge avec la perte d'une région). Un routage dirige le trafic vers des emplacements géographiques sains et modifie automatiquement la destination lorsqu'un emplacement est défectueux tout en arrêtant temporairement les couches de réplication de données. Amazon Route 53 permet des vérifications de l'état à des intervalles de 10 secondes, et une durée de vie (TTL) sur vos ensembles d'enregistrements atteignant seulement une seconde.

Contrôler les ressources

Identique au scénario 3½ 9 s, plus une alerte lorsqu'une région est détectée comme étant défectueuse ; le trafic est alors acheminé en dehors de celle-ci.

s'adapter aux modifications de la demande ;

Identique au scénario 3½ 9 s.

Implémentation de la modification

Le pipeline de déploiement dispose d'une suite de tests complète, y compris les tests de performance, de charge et de provocation de pannes. Nous déployons les mises à jour en utilisant les modèles de déploiement Canary ou bleu/vert dans une zone d'isolation à la fois, une région après

l'autre. Au cours du déploiement, les anciennes versions continueront à exécuter les instances afin de faciliter une restauration plus rapide. Elles sont entièrement automatisées, et permettent une restauration à un état antérieur si les indicateurs de performance clé indiquent un problème. La surveillance inclut des indicateurs de réussite ainsi que des alertes en cas de problème.

Les runbooks sont prévus pour les exigences de rapport strictes et le suivi des performances. Si des opérations fructueuses tendent vers un non-respect des objectifs de performance ou de disponibilité, un playbook est utilisé pour établir la cause de cette tendance. Les playbooks sont prévus pour les modes d'échec non découverts et les incidents de sécurité. Des playbooks sont également prévus pour établir la cause racine des défaillances.

L'équipe qui crée le site Web exploite également ce dernier. Cette équipe identifie la correction d'erreur de toute défaillance inattendue et hiérarchise le correctif à déployer une fois qu'il est mis en œuvre. Nous collaborons également avec AWS Support pour l'offre Infrastructure Event Management.

sauvegarder les données ;

Identique au scénario 3½ 9 s.

Conception de la résilience

Les applications doivent être créées à l'aide des modèles de résilience de logiciel/application. Il est possible que de nombreuses autres couches de routage soient requises pour mettre en œuvre la disponibilité nécessaire. Il ne faut pas sous-estimer la complexité de cette implémentation supplémentaire. L'application sera mise en œuvre dans les zones d'isolation des défaillances de déploiement, et partitionnée et déployée de façon à ce que même un événement à l'échelle de la région n'affecte pas tous les clients.

tester la résilience ;

Nous validons l'architecture par des jeux de rôle en utilisant les runbooks afin de nous assurer que nous pouvons effectuer les tâches sans nous écarter des procédures.

Planification de la reprise après sinistre

Actif-actif Déploiement multirégion actif-actif avec une infrastructure de charge de travail complète et des données dans plusieurs régions. Dans le cadre d'une stratégie globale en lecture locale et en écriture, une région constitue la base de données principale de toutes les écritures et les données sont répliquées pour les lectures vers d'autres régions. Si la région de base de données principale

échoue, une nouvelle base de données doit être promue. Dans un système de lecture local et d'écriture globale, les utilisateurs sont affectés à une région d'origine dans laquelle les écritures de base de données sont gérées. Cela permet aux utilisateurs de lire ou d'écrire à partir de n'importe quelle région. Ce système requiert toutefois une logique complexe de gestion des conflits de données potentiels entre les écritures dans différentes régions.

Lorsqu'une région est détectée comme défectueuse, la couche de routage achemine automatiquement le trafic vers les régions saines restantes. Aucune intervention manuelle n'est requise.

Les magasins de données doivent être répliqués entre les régions d'une manière permettant de résoudre les conflits potentiels. Il est nécessaire de créer des outils et processus automatisés pour copier ou déplacer les données entre les partitions pour les raisons de latence et pour équilibrer les requêtes ou les volumes de données dans chaque partition. La correction de la résolution des conflits de données nécessite également des runbooks opérationnels supplémentaires.

Objectif de la conception de la disponibilité

Nous supposons que d'importants investissements sont effectués pour automatiser toutes les restaurations, et que la récupération peut être terminée dans un délai d'une minute. Nous ne supposons aucune récupération déclenchée manuellement, mais au maximum une action de récupération automatisée par trimestre. Cela correspond à quatre minutes par an pour récupérer. Nous supposons que l'application est en ligne continuellement par le biais de mises à jour. Par conséquent, nos objectif de la conception de la disponibilité est de 99,999 %.

Résumé

Rubrique	Implémentation
Contrôler les ressources	Vérifications de l'état de santé au niveau de toutes les couches, y compris l'état du DNS au niveau de la région AWS et au niveau des indicateurs clés de performance ; alertes envoyées au déclenchement des alarmes configurées ; alerte à la moindre panne. Les réunions opérationnelles sont rigoureuses pour détecter les tendances et gérer les objectifs de conception.

Rubrique	Implémentation
s'adapter aux modifications de la demande ;	ELB pour le niveau d'application de mise à l'échelle Web et automatique ; mise à l'échelle automatique du stockage et réplicas en lecture dans plusieurs zones dans les régions actives et passives pour Aurora RDS. Données et infrastructure synchronisées entre les régions AWS pour garantir la stabilité statique.
Implémentation de la modification	Déploiement automatisé (Canary ou bleu/vert) et restauration automatique lorsque les indicateurs clés de performance ou les alertes indiquent des problèmes non détectés dans l'application. Les déploiements sont effectués vers une zone d'isolement dans une seule région AWS à la fois.
sauvegarder les données ;	Sauvegardes automatisées dans chaque région AWS via RDS pour respecter le RPO et la restauration automatisée pratiquée régulièrement au cours d'un test de simulation des pannes. Les données Aurora RDS et S3 sont répliquées automatiquement de manière asynchrone de la région active vers la région passive.
Conception de la résilience	Implémentation de zones d'isolation des pannes pour l'application ; Auto Scaling pour fournir un niveau d'application et Web auto-régénérant ; RDS multi-AZ ; basculement régional automatisé.

Rubrique	Implémentation
tester la résilience ;	Intégration du test des pannes des composant s et des zones d'isolation au pipeline et mise en œuvre régulière par le personnel opération nel au cours d'un test de simulation de pannes ; playbooks pour diagnostiquer les problèmes inconnus ; processus d'analyse des causes racines avec acheminement des communica tions sur le type de problème, sa correction et sa prévention. La correction RCA est prioritai re sur les versions de fonctions pour une implémentation et un déploiement immédiats.
Planification de la reprise après sinistre	Fonctionnalité active-active déployée dans au moins deux régions. L'infrastructure est entièrement mise à l'échelle et statistiquement stable d'une région à l'autre. Les données sont partitionnées et synchronisées entre les régions. Sauvegardes chiffrées via RDS. La défaillance d'une région est testée au cours d'un test de simulation des pannes et coordonnée avec AWS. Pendant la restaurat ion, il peut être nécessaire de promouvoir une nouvelle base de données principale.

Ressources

Documentation

- Bibliothèque Amazon Builders' Library - Comment Amazon crée et exploite les logiciels
- Centre d'architecture AWS

Ateliers

- Ateliers AWS Well-Architected sur la fiabilité

Liens externes

- Modèle de file d'attente adaptative : panne lors de la mise à l'échelle
- Disponibilité et plus encore : comprendre et améliorer la résilience des systèmes distribués sur AWS

Livres

- Robert S. Hammer « Patterns for Fault Tolerant Software »
- Andrew Tanenbaum et Marten van Steen « Distributed Systems: Principles and Paradigms »

Conclusion

Que vous commenciez juste à découvrir les sujets de la disponibilité et de la fiabilité ou que vous soyez un utilisateur aguerri cherchant plus de connaissances pour optimiser la disponibilité des charges de travail critiques, nous espérons que ce livre blanc vous a permis de réfléchir à ce sujet, vous a présenté de nouvelles idées ou a soulevé de nouvelles questions. Nous espérons que cela vous permettra d'obtenir une meilleure compréhension du niveau de disponibilité qui correspond aux besoins de votre entreprise et de la procédure de mise en œuvre de la fiabilité pour y parvenir. Nous vous encourageons à tirer parti des recommandations en matière de conception, d'opérations et de récupération présentées ici, ainsi que des connaissances et de l'expérience de nos architectes de solutions AWS. Nous aimerions faire votre connaissance : n'hésitez pas à nous communiquer vos témoignages de réussites pour atteindre des niveaux de disponibilité élevés sur AWS. Contactez l'équipe de votre compte ou utilisez la fonction Contactez-nous sur notre site Web.

Participants

Ont contribué à la préparation du présent document :

- Seth Eliot, Principal Developer Advocate, Amazon Web Services
- Mahanth Jayadeva, Jon Steele, architecte de solutions Well-Architected, Amazon Web Services
- Amulya Sharma, architecte de solutions principal, Amazon Web Services
- Jason DiDomenico, architecte de solutions senior, Cloud Foundations, Amazon Web Services
- Marcin Bednarz, architecte de solutions principal, Amazon Web Services
- Tyler Applebaum, architecte de solutions senior, Amazon Web Services
- Rodney Lester, architecte de solutions principal, modernisation des applications, Amazon Web Services
- Joe Chapman, architecte de solutions senior, Amazon Web Services
- Adrian Hornsby, ingénieur principal en développement système, Amazon Web Services
- Kevin Miller, vice-président, S3, Amazon Web Services
- Shannon Richards, responsable de programmes techniques, Amazon Web Services
- Laurent Domb, technologue en chef, services financiers fédéraux, Amazon Web Services
- Kevin Schwarz, architecte de solutions senior, Amazon Web Services
- Rob Martell, architecte principal de la résilience du cloud, Amazon Web Services
- Priyam Reddy, architecte de solutions senior et responsable DR, Amazon Web Services
- Jeff Ferris, technologue principal, Amazon Web Services
- Matias Battaglia, architecte de solutions senior Amazon Web Services

Autres lectures

Pour en savoir plus, voir :

- AWS Well-Architected Framework
- Centre d'architecture AWS

Révisions du document

Pour être informé des mises à jour de ce livre blanc, abonnez-vous au flux RSS.

Modification	Description	Date
Livre blanc mis à jour	Les bonnes pratiques ont été mises à jour avec de nouvelles recommandations en matière d'implémentation.	June 27, 2024
Mises à jour des conseils sur les bonnes pratiques	Les bonnes pratiques ont été mises à jour pour inclure de nouveaux conseils dans les domaines suivants : Concevoir des interactions dans un système distribué pour éviter les défaillances, Concevoir des interactions dans un système distribué pour résister aux défaillances ou les atténuer, Surveiller les ressources de charge de travail, Concevoir votre charge de travail de sorte qu'elle s'adapte aux changements de demande, Implémentation de la modification et Test de la fiabilité.	December 6, 2023
Mises à jour des conseils sur les bonnes pratiques	Les bonnes pratiques ont été mises à jour pour inclure de nouveaux conseils dans les domaines suivants : Surveiller les ressources de charge de travail et Conception d'une charge de travail qui	October 3, 2023

	résiste aux défaillances des composants.	
Mises à jour des conseils sur les bonnes pratiques	Les bonnes pratiques ont été mises à jour pour inclure de nouveaux conseils dans les domaines suivants : Conception de l'architecture de votre service de charge de travail, Concevoir des interactions dans un système distribué pour résister aux défaillances ou les atténuer et Surveiller les ressources de charge de travail.	July 13, 2023
Mise à jour mineure	Suppression du langage non inclusif.	April 13, 2023
Mises à jour du nouveau cadre	Les bonnes pratiques ont été mises à jour avec des recommandations et de nouvelles bonnes pratiques.	April 10, 2023
Livre blanc mis à jour	Les bonnes pratiques ont été mises à jour avec de nouvelles recommandations en matière d'implémentation.	December 15, 2022
Mises à jour mineures	Chiffres corrigés et modifications mineures.	November 17, 2022
Livre blanc mis à jour	Développement des bonnes pratiques et ajout de plans d'amélioration.	October 20, 2022

Livre blanc mis à jour	Ajout de deux nouvelles bonnes pratiques pour le pilier Fiabilité dans les sections Utilisation de l'isolation des pannes pour protéger votre charge de travail et Conceptio n d'une charge de travail qui résiste aux défaillances des composants.	May 5, 2022
Mise à jour mineure	Ajout du pilier Durabilité dans l'introduction.	December 2, 2021
Livre blanc mis à jour	Mise à jour des conseils de reprise après sinistre pour inclure Route 53 Application Recovery Controller. Ajout de références à DevOps Guru. Mise à jour de plusieurs liens de ressources et autres modifications éditoriales mineures.	October 26, 2021
Mise à jour mineure	Ajout d'informations sur AWS Fault Injection Service (AWS FIS).	March 15, 2021
Mise à jour mineure	Mise à jour mineure du texte.	January 4, 2021

Livre blanc mis à jour	Mise à jour de l'annexe A pour refléter l'objectif de conceptio n Disponibilité pour Amazon SQS, Amazon SNS et Amazon MQ. Réorganisation des lignes dans le tableau pour faciliter la recherche. Amélioration de l'explication des différenc es entre la disponibilité et la reprise après sinistre et comment elles contribuent toutes deux à la résilience. Détails supplémentaires sur la couverture des architect ures multirégionales (pour la disponibilité) et des stratégie s multirégionales (pour la reprise après sinistre). Mise à jour du livre référencé vers la dernière version. Détails supplémentaires sur les calculs de disponibilité pour inclure le calcul basé sur les demandes et les calculs de raccourci. Amélioration de la description des jeux de rôle.	December 7, 2020
Mise à jour mineure	Mise à jour de l'annexe A pour refléter l'objectif de la conception de la disponibilité pour AWS Lambda	October 27, 2020
Mise à jour mineure	Mise à jour de l'annexe A pour ajouter l'objectif de la conception de la disponibilité pour AWSGlobal Accelerator	July 24, 2020

Mises à jour pour le nouveau cadre	Mises à jour importantes et contenu nouveau/révisé, y compris : ajout de la section « Architecture de la charge de travail » sur les bonnes pratiques, réorganisation des bonnes pratiques dans les sections Gestion des modifications et Gestion des pannes, ressources mises à jour, mise à jour pour inclure les dernières ressources et les derniers services AWS comme AWS Global Accelerator, AWS Service Quotas, et AWS Transit Gateway, ajout/mise à jour des définitions des termes Fiabilité, Disponibilité et Résilience, Livre blanc mieux adapté à l'outil AWS Well-Architected Tool (questions et bonnes pratiques) utilisé pour les évaluations Well-Architected, réorganisation des principes de conception, déplacement de Reprise automatique après une panne avant Test des procédures de reprise, mise à jour des diagrammes et des formats pour les équations, suppression des sections Services clés et intégration à la place des références aux services AWS clés dans les bonnes pratiques.	July 8, 2020

Mise à jour mineure	Correction d'un lien rompu	October 1, 2019
Livre blanc mis à jour	Mise à jour de l'annexe A	April 1, 2019
Livre blanc mis à jour	Ajout de recommandations de mise en réseau spécifiques à AWS Direct Connect et d'objectifs de conception de service supplémentaires	September 1, 2018
Livre blanc mis à jour	Ajout des sections Principes de conception et Gestion des limites. Mise à jour des liens, suppression de l'ambiguïté des expressions en amont/en aval et ajout de références explicites aux rubriques restantes du pilier Fiabilité dans les scénarios de disponibilité.	June 1, 2018
Livre blanc mis à jour	Solution DynamoDB entre régions remplacée par DynamoDB Global Tables. Ajout d'objectifs de conception de service	March 1, 2018
Mises à jour mineures	Correction mineure du calcul de la disponibilité pour inclure la disponibilité de l'application	December 1, 2017
Livre blanc mis à jour	Mise à jour pour fournir des conseils sur les conceptions haute disponibilité, y compris les concepts, les bonnes pratiques et les exemples d'implémentation.	November 1, 2017

Publication initiale	Pilier Fiabilité - AWS Well-Architected Framework publié.	November 1, 2016