aws

用户指南

# AWS 已验证的访问权限

# AWS 已验证的访问权限: 用户指南

# Table of Contents

# 什么是 AWS Verified Access？

借助 AWS Verified Access，您无需使用虚拟专用网络（VPN）即可提供对应用程序的安全访问。Verified Access 会评估每个应用程序请求，并帮助确保用户只有在满足指定的安全要求时才能访问每个应用程序。

## Verified Access 的优势

- 改善安全状况 – 传统的安全模型只评估一次访问权限，并授予用户对所有应用程序的访问权限。Verified Access 会实时评估每个应用程序访问请求。这使得不良行为者很难从一个应用程序转移到另一个应用程序。
- 与安全服务集成 – Verified Access 与身份和设备管理服务（包括 AWS 和第三方服务）集成。利用这些服务提供的数据，Verified Access 根据一系列安全要求验证用户和设备的可信度，并确定用户是否应有权访问应用程序。
- 改善用户体验 – Verified Access 使用户无需使用 VPN 即可访问您的应用程序。这有助于减少由 VPN 相关问题引起的支持案例数量。
- 简化故障排除和审核 – Verified Access 会记录所有访问尝试，提供对应用程序访问的集中可见性，从而帮助您快速响应安全事件和审核请求。

## 访问 AWS Verified Access

您可以使用以下任一界面来使用 Verified Access：

- AWS Management Console – 提供可用于创建和管理 Verified Access 资源的 Web 界面。登录到 AWS Management Console并打开 Amazon VPC 控制台，网址：https://console.aws.amazon.com/vpc/。
- AWS Command Line Interface (AWS CLI) – 提供适用于各种 AWS 服务（包括 AWS Verified Access）的命令。AWS CLI 在 Windows、macOS 和 Linux 上受支持。要获取 AWS CLI，请参阅 AWS Command Line Interface。
- AWS 开发工具包 – 提供特定于语言的 API。AWS 开发工具包处理许多连接详细信息，例如计算签名以及处理请求重试和错误。有关更多信息，请参阅 AWS 软件开发工具包。
- 查询 API – 提供了您使用 HTTPS 请求调用的低级别 API 操作。使用查询 API 是访问 Verified Access 的最直接方式。但它需要您的应用程序处理低级别的详细信息，例如生成哈希值以签署请求以及处理错误。有关更多信息，请参阅《Amazon EC2 API 参考》中的 Verified Access 操作。

本指南介绍如何使用 AWS Management Console 来创建、访问和管理 Verified Access 资源。

# 定价

您需要为 Verified Access 上的每个应用程序按小时付费，并根据 Verified Access 处理的数据量付费。有关更多信息，请参阅 AWS Verified Access 定价。

# Verified Access 的工作原理

AWS Verified Access 会评估用户的每个应用程序请求，并允许基于以下内容的访问：

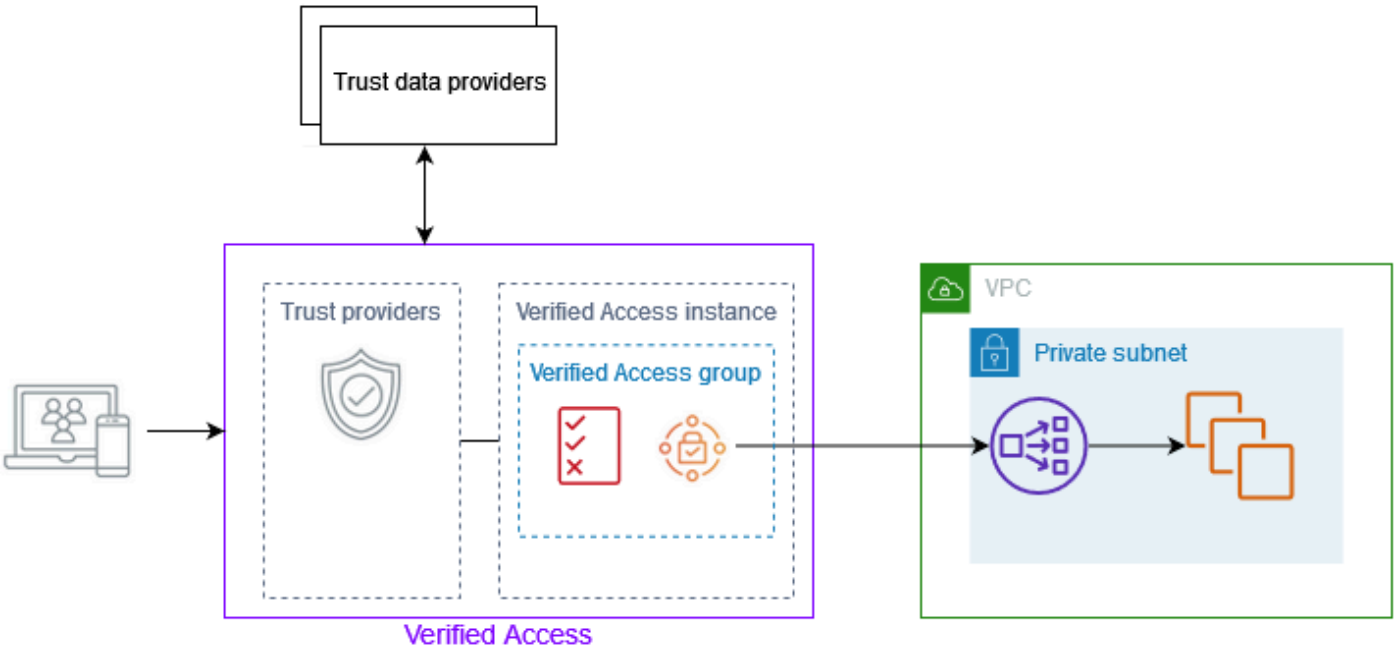
- 您选择的信任提供商发送的信任数据（来自 AWS 或第三方）。
- 您在 Verified Access 中创建的访问策略。

当用户尝试访问应用程序时，Verified Access 会从信任提供商处获取数据，并根据您为该应用程序设置的策略对其进行评估。只有当用户满足您指定的安全要求时，Verified Access 才会授予对所请求应用程序的访问权限。默认情况下，在定义策略前，所有应用程序请求都会被拒绝。

此外，Verified Access 还会记录每次访问尝试，以帮助您快速响应安全事件和审核请求。

## Verified Access 的关键组件

下图提供了 Verified Access 的简要概述。用户发送访问应用程序的请求。Verified Access 根据组的访问策略和任何特定于应用程序的端点策略来评估请求。如果允许访问，请求会通过端点发送到应用程序。




- Verified Access 实例 – 一个实例评估应用程序请求并仅在您的安全要求获得满足时才授予访问权限。
- Verified Access 端点 – 每个端点代表一个应用程序。您可以创建负载均衡器端点或网络接口端点。

- Verified Access 组 – Verified Access 端点的集合。我们建议您对具有相似安全要求的应用程序的端点进行分组，以简化策略管理。例如，您可以将所有销售应用程序的端点分到一组。
- 访问策略 – 一组用户定义的规则，用于确定是允许还是拒绝访问应用程序。您可以指定各种因素的组合，包括用户身份和设备安全状态。您可以为每个 Verified Access 组创建一个组访问策略，该策略会被组中的所有端点继承。您可以选择创建特定于应用程序的策略并将其附加到特定端点。
- 信任提供商 – 一种管理用户身份或设备安全状态的服务。Verified Access 可使用 AWS 和第三方信任提供商。每个 Verified Access 实例必须附加至少一个信任提供商。您可以将单个身份信任提供商和多个设备信任提供商附加到每个 Verified Access 实例。
- 信任数据 – 信任提供商发送给 Verified Access 的用户或设备的安全相关数据。也被称为用户声明或信任上下文。例如，用户的电子邮件地址或设备的操作系统版本。Verified Access 在收到每个访问应用程序的请求时，会根据您的访问策略评估这些数据。

# 教程：Verified Access 入门

利用本教程开始使用 AWS Verified Access。您将了解如何创建和配置 Verified Access 资源。

在将此应用程序添加到 Verified Access 之前，只能通过您的专用网络访问该应用程序。在本教程的最后，特定用户无需使用 VPN 即可通过 Internet 访问同一应用程序。

> **Note**
>
> 此示例不说明与基于设备的信任提供商的集成。在此示例中，我们只使用基于身份的信任提供商。

任务



## 先决条件

本教程的先决条件如下：

- 为说明这个使用 Verified Access 的示例，我们将使用两个 AWS 账户。一个账户将托管您的目标应用程序，Verified Access 资源将在另一个账户中创建。

- 在您所在的 AWS 区域 中启用 AWS IAM Identity Center。然后，您可以将 IAM Identity Center 用作 Verified Access 的信任提供商。有关更多信息，请参阅 AWS IAM Identity Center 用户指南中的启用 IAM Identity Center。
- 公共托管域以及更新该域的 DNS 记录所需的权限。
- 应用程序在 AWS 账户 中的内部负载均衡器后运行。我们将使用的示例应用程序域名为 `www.myapp.example.com`。
- 确保您的 IAM policy 具有 创建 Verified Access 实例的策略 中所述的创建 AWS Verified Access 实例所需的所有权限。

# 步骤 1：创建 Verified Access 实例

使用以下过程创建 Verified Access 实例。

要创建 Verified Access 实例

1. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。
2. 在 Amazon VPC 导航窗格中，选择 Verified Access 实例，然后选择创建 Verified Access 实例。
3. （可选）在名称和描述中，输入 Verified Access 实例的名称和描述。
4. 对于信任提供商，保留默认选项。
5. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
6. 选择创建 Verified Access 实例。

# 步骤 2：配置信任提供商

您可以设置 AWS IAM Identity Center 为信任提供商。

创建 IAM Identy Center 信任提供商

1. 在 Amazon VPC 导航窗格中，选择 Verified Access 信任提供商，然后选择创建 Verified Access 信任提供商。
2. （可选）在名称标签和描述中，输入 Verified Access 信任提供商的名称和描述。
3. 为策略引用名称输入一个自定义标识符，以便日后在处理策略规则时使用。例如，您可以输入 **`idc`**。
4. 在信任提供商类型下，选择用户信任提供商。

5. 在用户信任提供商类型下，选择 IAM Identity Center。
6. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
7. 选择创建 Verified Access 信任提供商。

## 步骤 3：将您的信任提供商附加到实例

使用以下过程将信任提供商附加到您的 Verified Access 实例。

要将信任提供商附加到实例

1. 在 Amazon VPC 导航窗格中，选择 Verified Access 实例。
2. 选择您的实例。
3. 选择操作、附加 Verified Access 信任提供商。
4. 对于 Verified Access 信任提供商，选择您的信任提供商。
5. 选择附加 Verified Access 信任提供商。

## 步骤 4：创建 Verified Access 组

让我们创建一个可用于下一步中创建的端点的组。

要创建 Verified Access 组

1. 在 Amazon VPC 导航窗格中，选择 Verified Access 组，然后选择创建 Verified Access 组。
2. （可选）在名称标签和描述中，输入组的名称和描述。
3. 对于 Verified Access 实例，请选择您的 Verified Access 实例。
4. 对于策略定义，将其留空。您将在本教程的后面部分中创建策略。
5. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
6. 选择创建 Verified Access 组。

## 步骤 5 ：通过 AWS Resource Access Manager 共享 Verified Access 组

在此步骤中，您将与正在运行目标应用程序的 AWS 账户 共享刚刚创建的组。要共享 Verified Access 组，您必须将其添加到资源共享。如果您没有资源共享，则必须先创建一个资源共享。

如果您是 AWS Organizations 中某组织的一部分并且已在您的组织中启用共享，组织中的使用者将自动获得对共享 Verified Access 组的访问权限。否则，使用者会收到加入资源共享的邀请，并在接受邀请后获得对共享 Verified Access 组的访问权限。

按照 AWS RAM 用户指南中创建资源共享的步骤操作。对于选择资源类型，选择 Verified Access 组，然后选中 Verified Access 组的复选框。

有关更多信息，请参阅《AWS RAM 用户指南》中的入门。

## 步骤 6：通过创建端点添加应用程序

使用以下过程创建端点。此步骤假定您有一个应用程序在 Elastic Load Balancing 的内部负载均衡器后运行。

创建 Verified Access 端点

1. 在 Amazon VPC 导航窗格中，选择 Verified Access 端点，然后选择创建 Verified Access 端点。
2. （可选）在名称标签和描述中，输入端点的名称和描述。
3. 对于 Verified Access 组，选择您的 Verified Access 组。
4. 对于应用程序详细信息，执行以下操作：
   a. 在应用程序域中，输入应用程序的 DNS 名称。
   b. 在域证书 ARN下，选择您的公共 TLS 证书的 Amazon 资源名称（ARN）。
5. 对于端点详细信息，执行以下操作：
   a. 对于附件类型，选择 VPC。
   b. 对于安全组，选择要与端点关联的安全组。
   c. 在端点域前缀中，输入一个自定义标识符。这将添加到 Verified Access 生成的 DNS 名称之前。在本示例中，我们可以使用 **`my-ava-app`**。
   d. 对于端点类型，选择负载均衡器。
   e. 对于协议，选择 HTTPS 或 HTTP。这取决于您的负载均衡器的配置。
   f. 对于端口，输入端口号。这取决于您的负载均衡器的配置。
   g. 对于负载均衡器 ARN，选择您的负载均衡器。
   h. 对于子网，选择与您的负载均衡器关联的子网。
6. 对于策略定义，此时不要输入策略。我们将在教程的后面部分介绍此内容。

7. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
8. 选择创建 Verified Access 端点。

# 步骤 7：配置 DNS 设置

在此步骤中，您将应用程序的域名（例如 www.myapp.example.com）映射到 Verified Access 端点的域名。要完成 DNS 映射，请在您的 DNS 提供商处创建规范名称记录（CNAME）。创建 CNAME 记录后，用户对您的应用程序的所有请求都将发送到 Verified Access。

获取端点的域名

1. 在 Amazon VPC 导航窗格中，选择 Verified Access 端点。
2. 选择您之前创建的端点。
3. 选择端点的详细信息选项卡。
4. 从端点域下复制端点域。

在本教程中，端点的域名将是 `my-ava-app.edge-1a2b3c4d5e6f7g.vai-1a2b3c4d5e6f7g.prod.verified-access.us-west-2.amazonaws.com`。

在您的 DNS 提供商处创建 CNAME 记录：

| 记录名称 | Type | 值 |
| --- | --- | --- |
| www.myapp.example.com | CNAME | my-ava-app.edge-1a2b3c4d5e6f7g.vai-1a2b3c4d5e6f7g.prod。verified-access.us-west-2.amazonaws.com |

# 步骤 8：测试与应用程序的连接性

现在，您可以测试与应用程序的连接性。在网页浏览器中输入应用程序的域名。Verified Access 策略的默认行为是拒绝所有请求。由于我们还没有制定允许所有人访问的策略，因此所有请求都应被拒绝。

# 步骤 9：配置组级访问策略

使用以下过程修改 Verified Access 组并配置允许连接您的应用程序的访问策略。该策略的详细信息将取决于在 IAM Identity Center 中配置的用户和组。有关创建策略的信息，请参阅 Verified Access 策略。

修改 Verified Access 组

1. 在 Amazon VPC 导航窗格中，选择 Verified Access 组。
2. 选择您的组。
3. 选择操作、修改 Verified Access 组策略。
4. 输入策略。
5. 选择修改 Verified Access 组策略。

# 步骤 10：重新测试连接性

现在，您的组策略已就位，您可以访问您的应用程序了。在网页浏览器中输入应用程序的域名。请求应该被允许，并且您应该被重定向到应用程序。

# 清理

完成测试后，按照以下步骤删除已创建的资源。

删除使用本教程创建的 Verified Access 资源

1. 在 Amazon VPC 导航窗格中，选择 Verified Access 端点。选择要删除的端点。选择操作、删除 Verified Access 端点。
2. 在导航窗格中，选择 Verified Access 组。选择要删除的组。选择操作、删除 Verified Access 组。注意 - 您可能需要等待几分钟，直到端点删除过程完成。
3. 在 Amazon VPC 导航窗格中，选择 Verified Access 实例。选择您为本教程创建的实例。选择操作、分离 Verified Access 信任提供商。从下拉列表中选择信任提供商，然后选择分离 Verified Access 信任提供商。
4. 在 Amazon VPC 导航窗格中，选择 Verified Access 信任提供商。选择您为本教程创建的信任提供商。选择操作、删除 Verified Access 信任提供商。
5. 在 Amazon VPC 导航窗格中，选择 Verified Access 实例。选择您为本教程创建的实例。选择操作、删除 Verified Access 实例。

# Verified Access 实例

AWS Verified Access 实例是一种 AWS 资源，可帮助您组织信任提供者和 Verified Access 组。

主题



## 创建 Verified Access 实例

使用以下过程创建 Verified Access 实例。

要创建 Verified Access 实例

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 实例，然后选择创建 Verified Access 实例。
3. （可选）在名称和描述中，输入 Verified Access 实例的名称和描述。
4. （可选）如果需要 Verified Access 符合 FIPS，请为联邦信息处理标准 (FIPS)选择启用。
5. （可选）对于信任提供商，选择要附加到 Verified Access 实例的信任提供商。
6. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
7. 选择创建 Verified Access 实例。

## 将信任提供商附加到实例

使用以下过程将信任提供商附加到 Verified Access 实例。

将信任提供商附加到 Verified Access 实例

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 实例。

3. 选择实例。
4. 选择操作、附加 Verified Access 信任提供商。
5. 对于 Verified Access 信任提供商，选择信任提供商。
6. 选择附加 Verified Access 信任提供商。

## 将信任提供商与实例分离

使用以下过程将信任提供商与 Verified Access 实例分离。

将信任提供商与 Verified Access 实例分离

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择实例。
4. 选择操作、分离 Verified Access 信任提供商。
5. 对于 Verified Access 信任提供商，选择信任提供商。
6. 选择分离 Verified Access 信任提供商。

## 删除 Verified Access 实例

用完 Verified Access 实例后可以将其删除。在删除实例之前，必须先移除所有关联的信任提供商或 Verified Access 组。

要删除 Verified Access 实例

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择 Verified Access 实例。
4. 选择操作、删除 Verified Access 实例。
5. 提示进行确认时，输入 **delete**，然后选择 Delete（删除）。

# 集成 AWS WAF

除了 Verified Access 强制执行的身份验证和授权规则外，您可能还需要应用外围保护。这可以帮助您保护应用程序免受其他威胁。您可以通过将 AWS WAF 集成到 Verified Access 部署中来实现此目的。AWS WAF 是一个网页应用程序防火墙，可以监视转发到受保护的网页应用程序资源的 HTTP(S) 请求。有关 AWS WAF 的更多信息，请参阅 AWS WAF 开发人员指南中的 AWS WAF。

通过将 AWS WAF Web 访问控制列表（ACL）与 Verified Access 实例关联，可以将 AWS WAF 与 Verified Access 集成。Web ACL 是一种 AWS WAF 资源，可让您对受保护资源响应的所有 HTTP(S) Web 请求进行精细控制。在处理 AWS WAF 关联或取消关联请求时，连接到实例的所有 Verified Access 端点的状态都显示为 `updating`。请求完成后，状态将恢复为 `active`。您可以在 AWS Management Console 中或使用 AWS CLI 描述端点来查看状态。

> **Note**
>
> 您还可以使用 AWS WAF 控制台或 API 来完成此集成。您将需要 Verified Access 实例的 Amazon 资源名称（ARN）。您可以使用以下格式构建此 ARN：`arn:${Partition}:ec2:${Region}:${Account}:verified-access-instance/${VerifiedAccessInstanceId}`。

主题



## 集成 AWS WAF 所需的 IAM 权限

将 AWS WAF 与 Verified Access 集成包括仅限权限操作，这些操作不会直接响应 API 操作。AWS Identity and Access Management 服务授权参考中用 `[permission only]` 指明这些操作。请参阅服务授权参考中的 Amazon EC2 的操作、资源和条件键。

要使用 Web ACL，您的 AWS Identity and Access Management 主体必须具有以下权限。

- `ec2:AssociateVerifiedAccessInstanceWebAcl`
- `ec2:DisassociateVerifiedAccessInstanceWebAcl`

- `ec2:DescribeVerifiedAccessInstanceWebAclAssociations`
- `ec2:GetVerifiedAccessInstanceWebAcl`

## 关联 AWS WAF Web ACL

以下步骤说明了如何使用 AWS Management Console 将 AWS WAF Web 访问控制列表（ACL）与 Verified Access 实例关联。

> **Tip**
>
> 您需要有一个现有的 AWS WAF Web ACL 才能完成以下步骤。有关 Web ACL 的更多信息，请参阅 AWS WAF 开发人员指南 中的 Web 访问控制列表。

将 AWS WAF Web ACL 与 Verified Access 实例关联

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择 Verified Access 实例。
4. 选择集成选项卡。
5. 选择操作，然后选择关联 Web ACL。
6. 对于 Web ACL，选择现有 Web ACL，然后选择关联 Web ACL。

您也可以使用 AWS WAF 的 AWS Management Console 来完成此任务。有关更多信息，请参阅 AWS WAF 开发人员指南中的将 Web ACL 与 AWS 资源关联或取消关联。

## 检查 AWS WAF 集成的状态

您可以使用 AWS Management Console 验证 AWS WAF Web 访问控制列表（ACL）是否与 Verified Access 实例相关联。

查看 AWS WAF 与 Verified Access 实例集成的状态

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择 Verified Access 实例。

4. 选择集成选项卡。
5. 查看 WAF 集成状态下列出的详细信息。状态将显示为已关联或未关联，如果处于已关联状态，还会显示 Web ACL 标识符。

## 取消关联 AWS WAF Web ACL

以下步骤说明了如何使用 AWS Management Console 将 AWS WAF Web 访问控制列表（ACL）与 Verified Access 实例取消关联。

将 AWS WAF Web ACL 与 Verified Access 实例取消关联

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择 Verified Access 实例。
4. 选择集成选项卡。
5. 选择操作，然后选择取消关联 Web ACL。
6. 选择取消关联 Web ACL 进行确认。

您也可以使用 AWS WAF 的 AWS Management Console 来完成此任务。有关更多信息，请参阅 AWS WAF 开发人员指南中的将 Web ACL 与 AWS 资源关联或取消关联。

## Verified Access 的 FIPS 合规性

联邦信息处理标准 (FIPS) 是美国和加拿大政府标准，规定了对保护敏感信息的加密模块的安全要求。AWS Verified Access 提供了环境配置选项以遵守 FIPS 出版物 140-2。以下 AWS 区域提供 Verified Access 的 FIPS 合规性：

- 美国东部（俄亥俄州）
- 美国东部（弗吉尼亚州北部）
- 美国西部（北加利福尼亚）
- 美国西部（俄勒冈州）
- 加拿大（中部）

此页面展示如何将新的或现有 Verified Access 环境配置为符合 FIPS。

主题



## 配置现有的 Verified Access 环境以实现 FIPS 合规性

如果您有现有的 Verified Access 环境并且想要将其配置为符合 FIPS，则需要删除并重新创建某些资源才能启用 FIPS 合规性。

要将现有 AWS Verified Access 环境重新配置为符合 FIPS，请执行以下步骤。

1. 删除原始 Verified Access 端点、组和实例。您配置的信任提供商可以重复使用。
2. 创建 Verified Access 实例，确保在创建期间启用联邦信息处理标准 (FIPS)。此外，在创建过程中，附加要使用的 Verified Access 信任提供商，方法是从下拉列表中将其选中。
3. 创建 Verified Access 组。在组的创建过程中，将其与刚创建的 Verified Access 实例相关联。
4. 创建一个或多个 Verified Access 端点。在创建端点的过程中，将端点与在上一步创建的组相关联。

## 配置新的 Verified Access 环境以实现 FIPS 合规性

要配置符合 FIPS 的新 AWS Verified Access 环境，请执行以下步骤。

1. 配置信任提供商。根据您的需求，您需要创建用户身份信任提供商和（可选）基于设备的信任提供商。
2. 创建 Verified Access 实例，确保在此过程中启用联邦信息处理标准 (FIPS)。此外，在创建过程中，附加在上一步创建的 Verified Access 信任提供商，方法是从下拉列表中将其选中。
3. 创建 Verified Access 组。在组的创建过程中，将其与刚创建的 Verified Access 实例相关联。
4. 创建一个或多个 Verified Access 端点。在创建端点的过程中，将端点与在上一步创建的组相关联。

# Verified Access 的信任提供商

信任提供商是一种向 AWS Verified Access 发送有关用户和设备的信息的服务。此信息称为信任上下文。信任上下文可能包括基于用户身份的属性，例如电子邮件地址或“销售”组织的成员资格，或者设备信息，例如已安装的安全补丁或防病毒软件版本。

Verified Access 支持以下类别的信任提供商：

- 用户身份 – 一种身份提供者 (IdP) 服务，用于存储和管理用户的数字身份。
- 设备管理 – 适用于笔记本电脑、平板电脑和智能手机等设备的设备管理系统。

内容



## 用户身份信任提供商

您可以选择使用 AWS IAM Identity Center 或兼容 OpenID Connect 的用户身份信任提供商。

内容



### 使用 IAM Identity Center 作为信任提供商

您可以将 AWS IAM Identity Center 用作 AWS Verified Access 的用户身份信任提供商。

#### 先决条件和注意事项

- 您的 IAM Identity Center 实例必须是一个 AWS Organizations 实例。独立 AWS 账户 IAM Identity Center 实例将不起作用。
- 必须在要创建 Verified Access 信任提供商的同一 AWS 区域内启用 IAM Identity Center 实例。

有关不同实例类型的详细信息，请参阅《AWS IAM Identity Center 用户指南》中的 Manage organization and account instances of IAM Identity Center。

## 创建 IAM Identity Center 信任提供商

在您的 AWS 账户上启用 IAM Identity Center 后，您可以使用以下步骤将 IAM Identity Center 设置为 Verified Access 的信任提供商。

创建 IAM Identity Center 信任提供商（AWS 控制台）

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 信任提供商，然后选择创建 Verified Access 信任提供商。
3. （可选）在名称标签和描述中，输入信任提供商的名称和描述。
4. 在策略参考名称中输入一个标识符，以便日后处理策略规则时使用。
5. 在信任提供商类型下，选择用户信任提供商。
6. 在用户信任提供商类型下，选择 IAM Identity Center。
7. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
8. 选择创建 Verified Access 信任提供商。

创建 IAM Identity Center 信任提供商 (AWS CLI)

- create-verified-access-trust-provider (AWS CLI)

## 删除 IAM Identity Center 信任提供商

在删除信任提供商前，必须从附加该信任提供商的实例中删除所有端点和组配置。

删除 IAM Identity Center 信任提供商（AWS 控制台）

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 信任提供商，然后在 Verified Access 信任提供商下选择要删除的信任提供商。
3. 选择操作，然后选择删除 Verified Access 信任提供商。
4. 在文本框中输入 `delete` 以确认删除。
5. 选择删除。

删除 IAM Identity Center 信任提供商 (AWS CLI)

- delete-verified-access-trust-provider (AWS CLI)

# 使用 OpenID Connect 信任提供商

AWS Verified Access 支持使用标准 OpenID Connect (OIDC) 方法的身份提供商。您可以使用兼容 OIDC 的提供商作为 Verified Access 的用户身份信任提供商。但是，由于潜在的 OIDC 提供商数量众多，AWS 无法测试每个 OIDC 与 Verified Access 的集成。

Verified Access 从 OIDC 提供商的 `UserInfo Endpoint` 处获取其评估的信任数据。`Scope` 参数用于确定将检索哪几组信任数据。收到信任数据后，将根据该数据评估 Verified Access 策略。

> **Note**
>
> 在评估 Verified Access 策略时，Verified Access 不使用来自 OIDC 提供商发送的 `ID token` 的信任数据。仅根据策略评估来自 `UserInfo Endpoint` 的信任数据。

内容



## 创建 OIDC 信任提供商的先决条件

您需要直接从信任提供商服务中收集以下信息：

- Issuer
- 授权端点
- 令牌端点
- UserInfo 端点
- 客户端 ID
- 客户端密钥
- 范围

## 创建 OIDC 信任提供商

按照以下过程创建 OIDC 作为信任提供商。

创建 OIDC 信任提供商（AWS 控制台）

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 信任提供商，然后选择创建 Verified Access 信任提供商。
3. （可选）在名称标签和描述中，输入信任提供商的名称和描述。
4. 在策略参考名称中输入一个标识符，以便日后处理策略规则时使用。
5. 在信任提供商类型下，选择用户信任提供商。
6. 在用户信任提供者类型下，选择 OIDC (OpenID Connect)。
7. 在发布者中，输入 OIDC 发布者的标识符。
8. 在授权端点中，输入授权端点的完整 URL。
9. 在令牌端点中，输入令牌端点的完整 URL。
10. 在用户端点中，输入用户端点的完整 URL。
11. 在客户端 ID 中输入 OAuth 2.0 客户端标识符。
12. 在客户端密码中输入 OAuth 2.0 客户端密码。
13. 输入由您的身份提供商定义的以空格分隔的范围列表。范围至少需要“openid”范围。
14. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
15. 选择创建 Verified Access 信任提供商。

> **Note**
>
> 您需要向 OIDC 提供商的允许列表添加重定向 URI。为此，您需要使用 Verified Access 端点的 `ApplicationDomain`。这可以在 AWS Management Console 中找到，位置是 Verified Access 端点的详细信息选项卡下，或者使用 AWS CLI 描述端点。将以下内容添加到 OIDC 提供商的允许列表：https://`ApplicationDomain`/oauth2/idpresponse

创建 OIDC 信任提供商 (AWS CLI)

- create-verified-access-trust-provider (AWS CLI)

## 修改 OIDC 信任提供商

创建信任提供商后，您可以更新其配置。

修改 OIDC 信任提供商（AWS 控制台）

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 信任提供商，然后在 Verified Access 信任提供商下选择要修改的信任提供商。
3. 选择操作，然后选择修改 Verified Access 信任提供商。
4. 修改要更改的选项。
5. 选择修改 Verified Access 信任提供商。

修改 OIDC 信任提供商 (AWS CLI)

- modify-verified-access-trust-provider (AWS CLI)

### 删除 OIDC 信任提供商

在删除用户信任提供商前，您首先需要从附加了该信任提供商的实例中删除所有端点和组配置。

删除 OIDC 信任提供商（AWS 控制台）

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 信任提供商，然后在 Verified Access 信任提供商下选择要删除的信任提供商。
3. 选择操作，然后选择删除 Verified Access 信任提供商。
4. 在文本框中输入 `delete` 以确认删除。
5. 选择删除。

删除 OIDC 信任提供商 (AWS CLI)

- delete-verified-access-trust-provider (AWS CLI)

## 基于设备的信任提供商

您可以在 AWS Verified Access 中使用设备信任提供商。您可以在 Verified Access 实例中使用一个或多个设备信任提供商。

内容

## 支持的设备信任提供商

以下设备信任提供商可以与 Verified Access 集成：

- CrowdStrike – 使用 CrowdStrike 和 Verified Access 保护私有应用程序
- Jamf – 将 Verified Access 与 Jamf 设备身份集成
- JumpCloud – 集成 JumpCloud 和 AWS Verified Access

## 创建基于设备的信任提供商

按照以下步骤创建和配置用于 Verified Access 的设备信任提供商。

创建 Verified Access 设备信任提供商（AWS 控制台）

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 信任提供商，然后选择创建 Verified Access 信任提供商。
3. （可选）在名称标签和描述中，输入信任提供商的名称和描述。
4. 在策略参考名称中输入一个标识符，以便日后处理策略规则时使用。
5. 在信任提供商类型中选择设备身份。
6. 在设备身份类型中选择 Jamf、CrowdStrike 或 JumpCloud。
7. 在租户 ID 中输入租户应用程序的标识符。
8. （可选）对于公共签名密钥 URL，输入设备信任提供商共享的唯一密钥 URL。（Jamf、CrowdStrike 或 Jumpcloud 不需要此参数。）
9. 选择创建 Verified Access 信任提供商。

> **Note**
>
> 您需要向 OIDC 提供商的允许列表添加重定向 URI。为此，您需要使用 Verified Access 端点的 `DeviceValidationDomain`。这可以在 AWS Management Console 中找到，位置是

Verified Access 端点的详细信息选项卡下，或者使用 AWS CLI 描述端点。将以下内容添加到 OIDC 提供商的允许列表：https://DeviceValidationDomain/oauth2/idpresponse

创建 Verified Access 设备信任提供商 (AWS CLI)

- create-verified-access-trust-provider (AWS CLI)

## 修改基于设备的信任提供商

创建信任提供商后，您可以更新其配置。

修改 Verified Access 设备信任提供商（AWS 控制台）

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 信任提供商。
3. 选择信任提供商。
4. 选择操作，然后选择修改 Verified Access 信任提供商。
5. 根据需要修改描述。
6. （可选）对于公共签名密钥 URL，修改设备信任提供商共享的唯一密钥 URL。（如果您的设备信任提供商是 Jamf、CrowdStrike 或 Jumpcloud，则不需要此参数。）
7. 选择修改 Verified Access 信任提供商。

修改 Verified Access 设备信任提供商 (AWS CLI)

- modify-verified-access-trust-provider (AWS CLI)

## 删除基于设备的信任提供商

使用完信任提供商后可以将其删除。

删除 Verified Access 设备信任提供商（AWS 控制台）

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 信任提供商。
3. 在 Verified Access 信任提供商下选择要删除的信任提供商。

4. 选择操作，然后选择删除 Verified Access 信任提供商。
5. 提示进行确认时，输入 **delete**，然后选择 Delete（删除）。

### 删除 Verified Access 设备信任提供商 (AWS CLI)

- delete-verified-access-trust-provider (AWS CLI)

# Verified Access 组

AWS Verified Access 组是 Verified Access 端点和组级 Verified Access 策略的集合。组内的所有端点共享 Verified Access 策略。您可以使用组将具有共同安全要求的端点汇总在一起。通过使用一个策略来满足多个应用程序的安全需求，这有助于简化策略管理。

例如，您可以将所有销售应用程序分到一组并设置全组访问策略。然后，可以使用此策略为所有销售应用程序定义一组通用的最低安全要求。这种方法有助于简化策略管理。

创建组时，需要将组与 Verified Access 实例相关联。在创建端点的过程中，将端点与组相关联。

任务

- 创建 Verified Access 组
- 修改 Verified Access 组策略
- 删除 Verified Access 组

## 创建 Verified Access 组

使用以下过程创建 Verified Access 组。

要创建 Verified Access 组

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 组，然后选择创建 Verified Access 组。
3. （可选）在名称标签和描述中，输入组的名称和描述。
4. 对于 Verified Access 实例，选择要与该组关联的 Verified Access 实例。
5. （可选）在策略定义中，输入要应用于该组的 Verified Access 策略。
6. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
7. 选择创建 Verified Access 组。

## 修改 Verified Access 组策略

使用以下过程修改 Verified Access 组策略。

要修改 Verified Access 组策略

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 组，然后选择要修改其策略的组。
3. 选择操作，然后选择修改 Verified Access 组策略。
4. （可选）根据您的当前目标，打开或关闭启用策略。
5. （可选）在策略中，输入要应用于该组的 Verified Access 策略。
6. 选择修改 Verified Access 组策略。

# 删除 Verified Access 组

用完 Verified Access 组后可以将其删除。

要删除 Verified Access 组

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 组。
3. 选择组。
4. 选择操作、删除 Verified Access 组。
5. 提示进行确认时，输入 **`delete`**，然后选择 Delete（删除）。

# Verified Access 端点

一个 Verified Access 端点代表一个应用程序。每个端点都与一个 Verified Access 组相关联，并继承该组的访问策略。您可以选择将特定于应用程序的端点策略附加到每个端点。

目录



## Verified Access 端点类型

以下是可能的端点类型：

- 负载均衡器 – 将应用程序请求发送到负载均衡器以分发给您的应用程序。
- 网络接口 – 使用指定的协议和端口将应用程序请求发送到网络接口。

## 共享 VPC 和子网

以下是有关共享 VPC 子网的行为：

- VPC 子网共享支持 Verified Access 端点。参与者可以在共享子网中创建 Verified Access 端点。
- 创建了端点的参与者将是端点所有者，也是唯一被允许修改端点的一方。VPC 所有者将无权修改端点。
- 无法在 AWS 本地区域中创建 Verified Access 端点，因此无法通过本地区域进行共享。

有关更多信息，请参阅《Amazon VPC 用户指南》中的与其他账户共享 VPC。

# 为 Verified Access 创建负载均衡器端点

按照以下过程创建负载均衡器端点。有关负载均衡器的更多信息，请参阅 Elastic Load Balancing 用户指南。

要求

- 仅支持 IPv4 流量。
- 仅支持 HTTP 和 HTTPS 协议。
- 负载均衡器必须是应用程序负载均衡器或网络负载均衡器，并且必须是内部负载均衡器。
- 负载均衡器和子网必须属于同一虚拟私有云（VPC）。
- HTTPS 负载均衡器可以使用自签名证书或公共 TLS 证书。
- 您必须为应用程序提供域名。这是您的用户将用来访问您的应用程序的公共 DNS 名称。您还需要提供带有与此域名匹配的 CN 的公共 SSL 证书。可以使用 AWS Certificate Manager 创建或导入证书。

要创建负载均衡器端点

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 端点。
3. 选择创建 Verified Access 端点。
4. （可选）在名称标签和描述中，输入端点的名称和描述。
5. 在 Verified Access 组中，为端点选择一个 Verified Access 组。
6. 对于应用程序详细信息，执行以下操作：
   a. 在应用程序域中，输入应用程序的 DNS 名称。
   b. 在域证书 ARN 下，选择公共 TLS 证书。
7. 对于端点详细信息，执行以下操作：
   a. 对于附件类型，选择 VPC。
   b. 在安全组中，选择端点的安全组。来自 Verified Access 端点并进入负载均衡器的流量将与该安全组关联。
   c. 在端点域前缀中，输入一个自定义标识符，该标识符将添加到 Verified Access 为端点生成的 DNS 名称之前。
   d. 对于端点类型，选择负载均衡器。

e. 对于协议，选择 HTTP 或 HTTPS。

f. 在端口下，输入端口号。

g. 对于负载均衡器 ARN，选择负载均衡器。

h. 对于子网，选择负载均衡器的子网。

8. （可选）在策略定义中，输入端点的 Verified Access 策略。

9. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。

10. 选择创建 Verified Access 端点。

# 为 Verified Access 创建网络接口端点

使用以下过程创建网络接口端点。

要求

- 仅支持 IPv4 流量。
- 仅支持 HTTP 和 HTTPS 协议。
- 网络接口必须与安全组属于同一虚拟私有云（VPC）。
- 我们使用网络接口上的专用 IP 转发流量。
- 您必须为应用程序提供域名。这是您的用户将用来访问您的应用程序的公共 DNS 名称。您还需要提供带有与此域名匹配的 CN 的公共 SSL 证书。可以使用 AWS Certificate Manager 创建或导入证书。

要创建网络接口端点

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。

2. 在导航窗格中，选择 Verified Access 端点。

3. 选择创建 Verified Access 端点。

4. （可选）在名称标签和描述中，输入端点的名称和描述。

5. 在 Verified Access 组中，为端点选择一个 Verified Access 组。

6. 对于应用程序详细信息，执行以下操作：

a. 在应用程序域中，输入应用程序的 DNS 名称。

b. 在域证书 ARN 下，选择公共 TLS 证书。

7. 对于端点详细信息，执行以下操作：

   a. 对于附件类型，选择 VPC。
   b. 在安全组中，选择端点的安全组。来自 Verified Access 端点并进入网络接口的流量将与该安全组关联。
   c. 在端点域前缀中，输入一个自定义标识符，该标识符将添加到 Verified Access 为端点生成的 DNS 名称之前。
   d. 对于端点类型，选择网络接口。
   e. 对于协议，选择 HTTP 或 HTTPS。
   f. 在端口下，输入端口号。
   g. 对于网络接口，选择网络接口。
8. （可选）在策略定义中，输入端点的 Verified Access 策略。
9. （可选）若要添加标签，请选择 Add new tag（添加新标签），然后输入该标签的键和值。
10. 选择创建 Verified Access 端点。

# 允许来自 Verified Access 端点的流量

您可以为应用程序配置安全组，以便它们允许来自您的 Verified Access 端点的流量。为此，您可以添加一条入站规则，将端点的安全组指定为源。我们建议您删除所有其他入站规则，以便您的应用程序仅接收来自您的 Verified Access 端点的流量。

我们建议您保留现有的出站规则。

更新应用程序的安全组规则

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 端点。
3. 选择 Verified Access 端点，在详细信息选项卡上找到安全组 ID，然后复制端点的安全组的 ID。
4. 在导航窗格中，选择 Security groups（安全组）。
5. 选中与目标关联的安全组的复选框，然后选择操作、编辑入站规则。
6. 要添加允许来自您的 Verified Access 端点的流量的安全组规则，请执行以下操作：

   a. 选择 Add rule。
   b. 对于类型，选择所有流量或要允许的特定流量。
   c. 对于源，选择自定义，然后粘帖端点的安全组的 ID。

7. （可选）如需要求流量仅来自您的 Verified Access 端点，请删除所有其他入站安全组规则。
8. 选择 Save rules（保存规则）。

# 修改 Verified Access 端点

创建 Verified Access 端点后，可以更新其配置。

要修改 Verified Access 端点

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 端点。
3. 选择端点。
4. 选择操作、修改 Verified Access 端点。
5. 根据需要修改端点详细信息。
6. 选择修改 Verified Access 端点。

# 修改 Verified Access 端点策略

创建 Verified Access 端点后，可以修改其策略。

要修改 Verified Access 端点策略

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 端点。
3. 选择要修改其策略的端点。
4. 选择操作、修改 Verified Access 端点策略。
5. （可选）根据您的当前目标，打开或关闭启用策略。
6. （可选）在策略中，输入要应用于端点的 Verified Access 策略。
7. 选择修改 Verified Access 端点策略。

# 删除 Verified Access 端点

用完 Verified Access 端点后可以将其删除。

### 要删除 Verified Access 端点

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 端点。
3. 选择端点。
4. 选择操作、删除 Verified Access 端点。
5. 提示进行确认时，输入 **delete**，然后选择 Delete（删除）。

# 信任提供商的信任数据

信任数据是从信任提供商发送到 AWS Verified Access 的数据。有时也被称为“用户声明”或“信任上下文”。这些数据通常包括有关用户或设备的信息。信任数据的示例包括用户电子邮件、组成员资格、设备操作系统版本、设备安全状态等。发送的信息因信任提供商而异，因此您应参阅信托提供商的文档，以获取完整且更新的信任数据列表。

但是，通过使用 Verified Access 日志记录功能，您还可以查看您的信任提供商正在发送哪些信任数据。在定义允许或拒绝访问应用程序的策略时，这可能非常有用。有关在日志中包含信任上下文的信息，请参阅 包括信任上下文。

本节包含示例信任数据和策略编写入门示例。此处提供的信息仅供说明之用，不作为官方参考。

内容



## Verified Access 默认上下文

AWS Verified Access 默认在所有 Cedar 评估中包含一些有关当前 HTTP 请求的元素，无论您配置的信任提供商如何。评估策略时，Verified Access 会将有关当前 HTTP 请求的数据包含在 Cedar 上下文中的 `context.http_request key` 下。如果您愿意，可以编写根据数据进行评估的策略。以下 JSON 架构显示了评估中包含的数据。

```
{
     "title": "HTTP Request data included by Verified Access",
     "type": "object",
     "properties": {
         "user_agent": {
             "type": "string",
             "description": "The value of the User-Agent request header"
         },
         "x_forwarded_for": {
             "type": "string",
             "description": "The value of the X-Forwarded-For request header"
         },
```

```
        "http_method": {
            "type": "string",
            "description": "The HTTP Method provided (e.g. GET or POST)"
        },
        "hostname": {
            "type": "string",
            "description": "The value of the Host request header"
        },
        "port": {
           "type": "integer",
           "description": "The value of the verified access endpoint port"
        },
        "client_ip": {
            "type": "string",
            "description": "User ip connecting to the verified access endpoint"
        }
    }
}
```

以下是针对 HTTP 请求数据进行评估的策略示例。

```
forbid(principal, action, resource) when {
   context.http_request.http_method == "POST"
   && !(context.identity.roles.contains("Administrator"))
};
```

# AWS IAM Identity Center

评估策略时，如果将 AWS IAM Identity Center 定义为信任提供商，AWS Verified Access 会将信任数据包含在 Cedar 上下文中、您在信任提供商配置中指定为“策略参考名称”的键下。如果您愿意，可以编写根据信任数据进行评估的策略。

> **Note**
>
> 您的信任提供商的上下文键来自您在创建该信任提供商时配置的策略参考名称。例如，如果您将策略参考名称配置为“idp123”，则上下文键将为“context.idp123”。创建策略时，请检查是否正在使用正确的上下文键。

以下 JSON 架构显示了评估中包含的数据。

```
{
   "title": "AWS IAM Identity Center context specification",
   "type": "object",
   "properties": {
     "user": {
       "type": "object",
       "properties": {
         "user_id": {
           "type": "string",
           "description": "a unique user id generated by AWS IdC"
         },
         "user_name": {
           "type": "string",
           "description": "username provided in the directory"
         },
         "email": {
           "type": "object",
           "properties": {
             "address": {
               "type": "email",
               "description": "email address associated with the user"
             },
             "verified": {
               "type": "boolean",
               "description": "whether the email address has been verified by AWS IdC"
             }
           }
         }
       }
     },
     "groups": {
       "type": "object",
       "description": "A list of groups the user is a member of",
       "patternProperties": {
         "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]
{12}$": {
           "type": "object",
           "description": "The Group ID of the group",
           "properties": {
             "group_name": {
               "type": "string",
               "description": "The customer-provided name of the group"
             }
```

```
            }
          }
        }
      }
    }
  }
```

以下是根据 AWS IAM Identity Center 提供的信任数据进行评估的策略示例。

```
permit(principal, action, resource) when {
   context.idc.user.email.verified == true
   // User is in the "sales" group with specific ID
   && context.idc.groups has "c242c5b0-6081-1845-6fa8-6e0d9513c107"
 };
```

> **Note**
>
> 由于组名称可以更改，因此 IAM Identity Center 使用组 ID 来引用组。这有助于避免在更改组名称时违反策略声明。

# 第三方信任提供商

本节介绍第三方信任提供商提供给 AWS Verified Access 的信任数据。

> **Note**
>
> 您的信任提供商的上下文键来自您在创建该信任提供商时配置的策略参考名称。例如，如果您将策略参考名称配置为“idp123”，则上下文键将为“context.idp123”。确保在创建策略时使用正确的上下文键。

内容

## 浏览器扩展

如果您计划将设备信任上下文纳入您的访问策略中，您需要使用 AWS Verified Access 浏览器扩展或其他合作伙伴的浏览器扩展。Verified Access 目前支持 Google Chrome 和 Mozilla Firefox 浏览器。

我们目前支持三个设备信任提供商：Jamf（支持 macOS 设备）、CrowdStrike（支持 Windows 11 和 Windows 10 设备）以及 JumpCloud（支持 Windows 和 MacOS）。

- 如果您在策略中使用 Jamf 信任数据，您的用户必须在其设备上从 Chrome 应用商店或 Firefox 附加组件网站下载并安装 AWS Verified Access 浏览器扩展。
- 如果您在策略中使用 CrowdStrike 信任数据，首先，您的用户需要安装 AWS Verified Access Native Messaging Host（直接下载链接）。此组件是从用户设备上运行的 CrowdStrike 代理获取信任数据所必需的。然后，安装此组件后，用户必须在其设备上从 Chrome 应用商店或 Firefox 附加组件网站安装 AWS Verified Access 浏览器扩展。
- 如果您使用的是 JumpCloud，用户必须在其设备上从 Chrome 应用商店或 Firefox 附加组件网站安装 JumpCloud 浏览器扩展。

### Jamf

Jamf 是第三方信任提供商。评估策略时，如果将 Jamf 定义为信任提供商，Verified Access 会将信任数据包含在 Cedar 上下文中、您在信任提供商配置中指定为“策略参考名称”的键下。如果您愿意，可以编写根据信任数据进行评估的策略。以下 JSON 架构显示了评估中包含的数据。

有关将 Jamf 与 AWS Verified Access 配合使用的更多信息，请参阅 Jamf 网站上的将 AWS Verified Access 与 Jamf 设备身份集成。

```
{
    "title": "Jamf device data specification",
    "type": "object",
    "properties": {
        "iss": {
            "type": "string",
            "description": "\"Issuer\" - the Jamf customer ID"
        },
        "iat": {
            "type": "integer",
            "description": "\"Issued at Time\" - a unixtime (seconds since epoch) value
 of when the device information data was generated"
        },
```

```
        "exp": {
            "type": "integer",
            "description": "\"Expiration\" - a unixtime (seconds since epoch) value for
 when this device information is no longer valid"
        },
        "sub": {
            "type": "string",
            "description": "\"Subject\" - either the hardware UID or a value generated
 based on device location"
        },
        "groups": {
            "type": "array",
            "description": "Group IDs from UEM connector sync",
            "items": {
                "type": "string"
            }
        },
        "risk": {
            "type": "string",
            "enum": [
                "HIGH",
                "MEDIUM",
                "LOW",
                "SECURE",
                "NOT_APPLICABLE"
            ],
            "description": "a Jamf-reported level of risk associated with the device."
        },
        "osv": {
            "type": "string",
            "description": "The version of the OS that is currently running, in Apple
 version number format (https://support.apple.com/en-us/HT201260)"
        }
    }
}
```

以下是根据 Jamf 提供的信任数据进行评估的策略示例。

```
permit(principal, action, resource) when {
   context.jamf.risk == "LOW"
};
```

Cedar 提供了一个有用的 `.contains()` 函数来帮助处理像 Jamf 风险评分这样的枚举。

```
permit(principal, action, resource) when {
   ["LOW", "SECURE"].contains(context.jamf.risk)
};
```

## CrowdStrike

CrowdStrike 是一个第三方信任提供商。评估策略时，如果将 CrowdStrike 定义为信任提供商，Verified Access 会将信任数据包含在 Cedar 上下文中、您在信任提供商配置中指定为“策略参考名称”的键下。如果您愿意，可以编写根据信任数据进行评估的策略。以下 JSON 架构显示了评估中包含的数据。

有关将 CrowdStrike与 AWS Verified Access 配合使用的更多信息，请参阅 GitHub 网站上的使用 CrowdStrike 和 AWS Verified Access 保护私有应用程序

```
{
  "title": "CrowdStrike device data specification",
  "type": "object",
  "properties": {
    "assessment": {
      "type": "object",
      "description": "Data about CrowdStrike's assessment of the device",
      "properties": {
        "overall": {
          "type": "integer",
          "description": "A single metric, between 1-100, that accounts as a weighted
 average of the OS and and Sensor Config scores"
        },
        "os": {
          "type": "integer",
          "description": "A single metric, between 1-100, that accounts for the OS-
specific settings monitored on the host"
        },
        "sensor_config": {
          "type": "integer",
          "description": "A single metric, between 1-100, that accounts for the
 different sensor policies monitored on the host"
        },
        "version": {
          "type": "string",
          "description": "The version of the scoring algorithm being used"
        }
      }
```

```
    },
    "cid": {
      "type": "string",
      "description": "Customer ID (CID) unique to the customer's environemnt"
    },
    "exp": {
      "type": "integer",
      "description": "unixtime, The expiration time of the token"
    },
    "iat": {
      "type": "integer",
      "description": "unixtime, The issued time of the token"
    },
    "jwk_url": {
      "type": "string",
      "description": "URL that details the JWT signing"
    },
    "platform": {
      "type": "string",
      "enum": ["Windows 10", "Windows 11", "macOS"],
      "description": "Operating system of the endpoint"
    },
    "serial_number": {
      "type": "string",
      "description": "The serial number of the device derived by unique system
 information"
    },
    "sub": {
      "type": "string",
      "description": "Unique CrowdStrike Agent ID (AID) of machine"
    },
    "typ": {
      "type": "string",
      "enum": ["crowdstrike-zta+jwt"],
      "description": "Generic name for this JWT media. Client MUST reject any other
 type"
    }
  }
}
```

以下是根据 CrowdStrike 提供的信任数据进行评估的策略示例。

```
permit(principal, action, resource) when {
```

```
    context.crowdstrike.assessment.overall > 50
};
```

## JumpCloud

JumpCloud 是第三方信任提供商。评估策略时，如果将 JumpCloud 定义为信任提供商，Verified Access 会将信任数据包含在 Cedar 上下文中、您在信任提供商配置中指定为“策略参考名称”的键下。如果您愿意，可以编写根据信任数据进行评估的策略。以下 JSON 架构显示了评估中包含的数据。

有关使用 JumpCloud 与 AWS Verified Access 的更多信息，请参阅 JumpCloud 网站上的集成 JumpCloud 和 AWS Verified Access。

```
{
  "title": "JumpCloud device data specification",
  "type": "object",
  "properties": {
    "device": {
      "type": "object",
      "description": "Properties of the device",
      "properties": {
        "is_managed": {
          "type": "boolean",
          "description": "Boolean to indicate if the device is under management"
        }
      }
    },
    "exp": {
      "type": "integer",
      "description": "Expiration. Unixtime of the token's expiration."
    },
    "durt_id": {
      "type": "string",
      "description": "Device User Refresh Token ID. Unique ID that represents the
 device + user."
    },
    "iat": {
      "type": "integer",
      "description": "Issued At. Unixtime of the token's issuance."
    },
    "iss": {
      "type": "string",
      "description": "Issuer. This will be 'go.jumpcloud.com'"
    },
```

```
    "org_id": {
      "type": "string",
      "description": "The JumpCloud Organization ID"
    },
    "sub": {
      "type": "string",
      "description": "Subject. The managed JumpCloud user ID on the device."
    },
    "system": {
      "type": "string",
      "description": "The JumpCloud system ID"
    }
  }
}
```

以下是根据 JumpCloud 提供的信任上下文进行评估的策略示例。

```
permit(principal, action, resource) when {
   context.jumpcloud.org_id = 'Unique_orgnaization_identifier'
};
```

# 用户声明传递和签名验证

AWS Verified Access 实例成功验证用户身份后，会将从 IdP 收到的用户声明发送到 Verified Access 端点。用户声明经过签名，以便应用程序可以验证签名和声明是否由 Verified Access 发送。在此过程中，添加以下 HTTP 标头：

`x-amzn-ava-user-context`

此标头包含 JSON Web 令牌 (JWT) 格式的用户声明。JWT 格式包括 base64 URL 编码的标头、负载和签名。Verified Access 使用 ES384（使用 SHA-384 哈希算法的 ECDSA 签名算法）生成 JWT 签名。

应用程序可以将这些声明用于个性化或其他特定于用户的体验。应用程序开发人员应在使用前自行了解身份提供商提供的每个声明的唯一性和验证级别。通常，sub 声明是识别给定用户的最佳方法。

内容

- 示例：检索和解码 JWT

## 示例：OIDC 用户声明的签名 JWT

以下示例说明了 OIDC 用户声明的标头和有效负载（JWT 格式）。

标头示例：

```
{
"alg": "ES384",
"kid": "12345678-1234-1234-1234-123456789012",
"signer": "arn:aws:ec2:us-east-1:123456789012:verified-access-instance/vai-
abc123xzy321a2b3c",
"iss": "OIDC Issuer URL"
"exp": "expiration" (120 secs)
}
```

有效负载示例：

```
{
"sub": "xyzsubject",
"email": "xxx@amazon.com",
"email_verified": true,
"groups": [
"Engineering",
"finance"
]
}
```

## 示例：IAM Identity Center 用户声明的签名 JWT

以下示例说明了 IAM Identity Center 用户声明的标头和有效负载（JWT 格式）。

> **Note**
>
> 对于 IAM Identity Center，声明中仅包含用户信息。

标头示例：

```
{
"alg": "ES384",
"kid": "12345678-1234-1234-1234-123456789012",
"signer": "arn:aws:ec2:us-east-1:123456789012:verified-access-instance/vai-
abc123xzy321a2b3c",
"iss": "arn:aws:ec2:us-east-1:123456789012:verified-access-trust-provider/vatp-
abc123xzy321a2b3c",
"exp": "expiration" (120 secs)
}
```

有效负载示例：

```
{
"user": {
        "user_id": "f478d4c8-a001-7064-6ea6-12423523",
        "user_name": "test-123",
        "email": {
            "address": "test@amazon.com",
            "verified": false
        }
    }
}
```

## 公钥

由于 Verified Access 实例不会对用户声明加密，因此我们建议将 Verified Access 端点配置为使用 HTTPS。如果将 Verified Access 端点配置为使用 HTTP，请务必使用安全组限制至该端点的流量。

我们建议在执行任何基于声明的授权之前验证签名。要获取公钥，请从 JWT 标头中获取密钥 ID 并使用它从终端节点查找公钥：每个 AWS 区域 的端点如下：

`https://public-keys.prod.verified-access.<region>.amazonaws.com/<key-id>`

## 示例：检索和解码 JWT

以下代码示例说明了如何在 Python 3.9 中获取密钥 ID、公钥和有效负载。

```
import jwt
import requests
import base64
```

```
import json

# Step 1: Get the key id from JWT headers (the kid field)
encoded_jwt = headers.dict['x-amzn-ava-user-context']
jwt_headers = encoded_jwt.split('.')[0]
decoded_jwt_headers = base64.b64decode(jwt_headers)
decoded_jwt_headers = decoded_jwt_headers.decode("utf-8")
decoded_json = json.loads(decoded_jwt_headers)
kid = decoded_json['kid']

# Step 2: Get the public key from Regional endpoint
url = 'https://public-keys.prod.verified-access.' + region + '.amazonaws.com/' + kid
req = requests.get(url)
pub_key = req.text

# Step 3: Get the payload
payload = jwt.decode(encoded_jwt, pub_key, algorithms=['ES384'])
```

# Verified Access 策略

AWS Verified Access 策略允许您定义对托管在 AWS 中的应用程序进行访问的规则。这些规则是用 AWS 策略语言 Cedar 编写的。使用 Cedar，您可以创建策略，根据您配置为用于 Verified Access 的身份或基于设备的信任提供商发送的信任上下文来评估这些策略。

有关 Cedar 策略语言的更多详细信息，请参阅 Cedar 参考指南。

本节介绍 Verified Access 策略的结构、包含的内容以及定义方法，并提供了一些示例。

内容



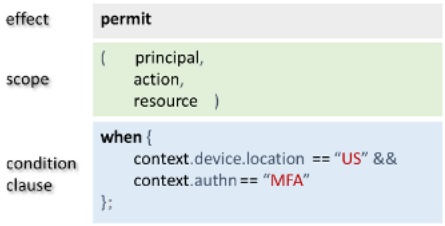
## 使用 Verified Access 的策略

在创建 Verified Access 组或创建 Verified Access 端点时，可以选择定义 Verified Access 策略。您可以在不定义 Verified Access 策略的情况下创建组或端点，但是在定义策略之前，所有访问请求都将被阻止。

要对现有 Verified Access 组或端点或在创建 Verified Access 组或端点后添加或更改策略，请参阅 修改 Verified Access 组策略 或 修改 Verified Access 端点策略。

## 策略声明结构

本节介绍 AWS Verified Access 策略声明及其评估方式。一个 Verified Access 策略中可以有多个声明。下图展示了 Verified Access 策略的结构。




策略包含以下几部分：

- 效果 – 指定策略语句是 `permit`（`Allow`）还是 `forbid`（`Deny`）。
- 范围 – 指定效果适用的主体、操作和资源。您可以通过不标识特定主体、操作或资源来使 Cedar 中的范围保持未定义状态（如前面的示例所示）。在这种情况下，策略适用于所有可能的主体、操作和资源。
- 条件子句 – 指定应用效果的上下文。

> **Important**
>
> 对于 Verified Access，通过在条件子句中引用信任上下文来完全表达策略。策略范围必须始终保持未定义状态。然后，您可以在条件子句中使用身份和设备信任上下文指定访问权限。

简单策略示例

```
permit(principal,action,resource)
when{
 context.<policy-reference-name>.<attribute> &&
 context.<policy-reference-name>.<attribute2>
};
```

在前面的示例中，请注意，您可以利用 && 运算符在一个策略语句中使用多个条件子句。使用 Cedar 策略语言可以创建自定义、精细和广泛的策略声明。有关其他示例，请参阅 策略示例。

# 策略评估

策略文档是一个或多个策略声明（`permit` 或 `forbid` 声明）的集合。如果条件子句（`when` 语句）为真，则策略适用。要使策略文档允许访问，文档中必须至少应用一个允许策略，并且不得适用任何禁止策略。如果没有应用允许策略和/或应用一个或多个禁止策略，则策略文档拒绝访问。如果您已为 Verified Access 组和 Verified Access 端点定义了策略文档，则这两个文档都必须允许访问。如果您尚未为 Verified Access 端点定义策略文档，则只有 Verified Access 组策略需要访问。

> **Note**
>
> AWS Verified Access 会在您创建策略时验证语法，但不会验证您在条件子句中输入的数据。

# 内置运算符

在使用各种条件创建 AWS Verified Access 策略的上下文（如 策略声明结构 中所讨论）时，您可以使用 && 运算符来添加其他条件。您还可以使用许多其他内置运算符来为您的策略条件添加更多的表达能力。下表包含所有内置运算符，以供参考。

| 运算符 | 类型和重载 | 描述 |
| --- | --- | --- |
| ! | Boolean → Boolean | 逻辑非。 |
| == | any → any | 等于。适用于任何类型的参数，即使类型不匹配。不同类型的值永远不会彼此相等。 |
| != | any → any | 不等于；与等于完全相反（见上文）。 |
| < | (long, long) → Boolean | 长整数小于。 |
| <= | (long, long) → Boolean | 长整数小于或等于。 |
| > | (long, long) → Boolean | 长整数大于。 |
| >= | (long, long) → Boolean | 长整数大于或等于。 |
| in | (entity, entity) → Boolean | 层次结构隶属（自反：A in A 始终为真）。 |
| | (entity, set(entity)) → Boolean | 层次结构隶属：A in [B, C, ...] 为真，如果 (A and B) \|\| (A in C) \|\| … 错误，如果集合包含非实体。 |
| && | (Boolean, Boolean) → Boolean | 逻辑与（短路）。 |

| 运算符 | 类型和重载 | 描述 |
| --- | --- | --- |
| \|\| | (Boolean, Boolean) → Boolean | 逻辑或（短路）。 |
| .exists() | entity → Boolean | 实体存在。 |
| has | (entity, attribute) → Boolean | 中缀运算符。`e has f` 测试记录或实体 `e` 是否具有属性 `f` 的绑定。如果 `e` 不存在或者 `e` 存在但没有属性 `f`，则返回 `false`。属性可以表示为标识符或字符串文字。 |
| like | (string, string) → Boolean | 中缀运算符。`t like p` 检查文本 `t` 是否与模式 `p` 匹配，其中可能包含与 0 个或多个任意字符匹配的通配符 `*`。为了匹配 `t` 中的文字星形字符，可以在 `p` 中使用特殊的转义字符序列 `\*`。 |
| .contains() | (set, any) → Boolean | 设置隶属关系（B 是 A 的元素吗）。 |
| .containsAll() | (set, set) → Boolean | 测试集合 A 是否包含集合 B 中的所有元素。 |
| .containsAny() | (set, set) → Boolean | 测试集合 A 是否包含集合 B 中的任意元素。 |

# 策略注释

您可以在 AWS Verified Access 策略中包含注释语句。注释被定义为以 `//` 开头和以换行符结尾的一行。

以下示例显示了策略中的注释语句。

```
// this policy grants access to users in a given domain with trusted devices
```

```
permit(principal, action, resource)
when {
  // the user's email address is in the @example.com domain
  context.idc.user.email.address.contains("@example.com")
  // Jamf thinks the user's computer is low risk or secure.
  && ["LOW", "SECURE"].contains(context.jamf.risk)
};
```

## 策略逻辑短路

您可能需要编写一个 AWS Verified Access 策略，以评估给定上下文中可能存在也可能不存在的数据。如果在上下文中引用了不存在的数据，Cedar 将产生错误并将策略评估为“禁止访问”，无论您的意图如何。例如，这将导致“禁止”，因为 `fake_provider` 和 `bogus_key` 在此上下文中不存在。

```
permit(principal, action, resource) when {
  context.fake_provider.bogus_key > 42
};
```

为避免这种情况，您可以使用 `has` 运算符检查是否存在某个键。如果 `has` 运算符返回假，则对链接语句的进一步评估将停止，Cedar 在尝试引用不存在的项目时不会产生错误。

```
permit(principal, action, resource) when {
  context.identity.user has "some_key" && context.identity.user.some_key > 42
};
```

这在指定引用两个不同信任提供商的策略时最为有用。

```
permit(principal, action, resource) when {
  // user is in an allowed group
  context.aws_idc.groups has "c242c5b0-6081-1845-6fa8-6e0d9513c107"
  &&(
    (
      // if CrowdStrike data is present,
      // permit if CrowdStrike's overall assessment is over 50
      context has "crowdstrike" && context.crowdstrike.assessment.overall > 50
    )
    ||
    (
      // if Jamf data is present,
      // permit if Jamf's risk score is acceptable
```

```
      context has "jamf" && ["LOW", "NOT_APPLICABLE", "MEDIUM",
 "SECURE"].contains(context.jamf.risk)
    )
  )
};
```

# 策略示例

示例 1：创建 IAM Identity Center 策略

> **Note**
>
> 由于组名称可以更改，因此 IAM Identity Center 使用组 ID 来引用组。这有助于避免在更改组名称时违反策略声明。

以下示例策略仅在用户属于 `finance` 组（组 ID 为 `c242c5b0-6081-1845-6fa8-6e0d9513c107`）且拥有经验证的电子邮件地址时才允许访问。

```
permit(principal,action,resource)
when {
    context.<policy-reference-name>.groups has "c242c5b0-6081-1845-6fa8-6e0d9513c107"
    && context.<policy-reference-name>.user.email.verified == true
};
```

示例 1b：向 IAM Identity Center 策略声明添加更多条件

以下示例策略仅在用户属于 `finance` 组（组 ID 为 `c242c5b0-6081-1845-6fa8-6e0d9513c107`）、拥有经过验证的电子邮件地址且 Jamf 设备风险评分为 `LOW` 时才允许访问。

```
permit(principal,action,resource)
when {
    context.<policy-reference-name>.groups has "c242c5b0-6081-1845-6fa8-6e0d9513c107"
    && context.<policy-reference-name>.user.email.verified == true
    && context.jamf.risk == "LOW"
};
```

示例 2：第三方 OIDC 提供商的相同策略

以下示例策略仅在用户来自“finance”组、拥有经过验证的电子邮件地址且 Jamf 设备风险评分为 LOW 时才允许访问。

```
permit(principal,action,resource)
when {
     context.<policy-reference-name>.groups.contains("finance")
     && context.<policy-reference-name>.email_verified == true
     && context.jamf.risk == "LOW"
};
```

示例 3：使用 CrowdStrike

以下示例策略在总体评估分数大于 50 时允许访问。

```
permit(principal,action,resource)
when {
    context.crwd.assessment.overall > 50
};
```

示例 4：使用特殊字符

以下示例说明当上下文属性使用 **:**（分号）时如何编写策略，该符号是策略语言中的保留字符。

```
permit(principal, action, resource)
when {
    context.<policy-reference-name>["namespace:groups"].contains("finance")
};
```

示例 5：允许特定的 IP 地址

以下示例展示了仅允许特定 IP 地址的策略。

```
permit(principal, action, resource)
when {
    context.http_request.client_ip == "192.0.2.1"
};
```

示例 5a：阻止特定 IP 地址

以下示例展示了将阻止特定 IP 地址的策略。

```
forbid(principal,action,resource)
when {
ip(context.http_request.client_ip).isInRange(ip("192.0.2.1/32"))
};
```

# Verified Access 策略助理

Verified Access 策略助理是 Verified Access 控制台中的一个工具，可用于测试和开发策略。Verified Access 策略助理在一个屏幕上显示端点策略、组策略和信任上下文，您可以在其中测试和编辑策略。

信任上下文格式因不同的信任提供商而异，有时 Verified Access 管理员可能不知道某个信任提供商使用的确切格式。因此，出于测试和开发目的，将信任上下文以及组和端点策略集中展示在一个位置会非常有帮助。

以下各节介绍了使用策略编辑器的基础知识。

任务

- 步骤 1：指定资源
- 步骤 2：编辑和测试策略
- 步骤 3：查看并应用更改

## 步骤 1：指定资源

在策略助理的第一页上，指定您希望使用的 Verified Access 端点。您还将指定用户（通过电子邮件地址标识），以及用户名和/或设备标识符（可选）。默认情况下，最新的授权决策提取自指定用户的 Verified Access 日志。您可以明确选择最新的允许或拒绝决定。

最后，信任上下文、授权决策、端点策略和组策略都显示在下一个屏幕上。

打开策略助理并指定资源

1. 通过以下网址打开 Amazon VPC 控制台：https://console.aws.amazon.com/vpc/。
2. 在导航窗格中，选择 Verified Access 实例，然后单击希望使用的 Verified Access 实例 ID。
3. 选择启动策略助理。
4. 对于用户电子邮件地址，输入用户的电子邮件地址。
5. 对于 Verified Access 端点，选择要编辑和测试策略的端点。
6. （可选）对于名称，提供用户的名称。

7. （可选）在设备标识符下，提供唯一设备标识符。
8. （可选）对于授权结果，选择要使用的最近授权结果的类型。默认情况下，将使用最新的授权结果。
9. 选择下一步。

## 步骤 2：编辑和测试策略

在此页面上，您将看到以下信息供您使用：

- 您的信任提供商为用户和（可选）您在上一步中指定的设备发送的信任上下文。
- 上一步中指定的 Verified Access 端点的 Cedar 策略。
- 端点所属的 Verified Access 组的 Cedar 策略。

可以在此页面上编辑 Verified Access 端点和组的 Cedar 策略，但信任上下文是静态的。现在，您可以使用此页面查看信任上下文以及 Cedar 策略。

通过选择测试策略按钮，即可根据信任上下文测试策略，授权结果将显示在屏幕上。您可以编辑策略并重新测试更改，根据需要重复该过程。

对策略所做的更改感到满意后，选择下一步继续进入策略助理的下一个屏幕。

## 步骤 3：查看并应用更改

在策略助理的最后一页上，您将看到对策略所做的更改，突出显示以便于查看。现在，您可以进行最后查看，然后选择应用更改即可提交更改。

您还可以通过选择上一页返回上一页，或者通过选择取消完全取消策略助理。

# AWS Verified Access 中的安全性

AWS 十分重视云安全性。为了满足对安全性最敏感的组织的需求，我们打造了具有超高安全性的数据中心和网络架构。作为 AWS 客户，您也将从这些数据中心和网络架构受益。

安全性是 AWS 和您的共同责任。责任共担模式将其描述为云的安全性和云中的安全性：

- 云的安全性 – AWS负责保护在 AWS Cloud 中运行 AWS 服务的基础设施。AWS 还向您提供可安全使用的服务。第三方审核员定期测试和验证我们的安全性的有效性，作为 AWS Compliance Programs 的一部分。要了解适用于 AWS Verified Access 的合规性计划，请参阅合规性计划范围内的 AWS 服务。
- 云中的安全性 - 您的责任由您使用的 AWS 服务决定。您还需要对其它因素负责，包括您的数据的敏感性、您公司的要求以及适用的法律法规。

此文档将帮助您了解如何在使用 Verified Access 时应用责任共担模型。以下主题说明如何配置 Verified Access 以实现您的安全性和合规性目标。您还会了解如何使用其他 AWS 服务以帮助您监控和保护 Verified Access 资源。

目录



## AWS Verified Access 中的数据保护

AWS 责任共担模式适用于 AWS Verified Access 中的数据保护。如该模式中所述，AWS 负责保护运行所有 AWS Cloud 的全球基础设施。您负责维护对托管在此基础设施上的内容控制。您还负责您所使用的 AWS 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问题解答。有关欧洲数据保护的信息，请参阅 AWS 安全性博客 上的博客文章 AWS Shared Responsibility Model and GDPR。

出于数据保护目的，建议您保护 AWS 账户 凭证并使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 设置单个用户。这样，每个用户只获得履行其工作职责所需的权限。我们还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 使用 SSL/TLS 与 AWS 资源进行通信。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用 AWS CloudTrail 设置 API 和用户活动日志记录。
- 使用 AWS 加密解决方案以及 AWS 服务中的所有默认安全控制。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果在通过命令行界面或 API 访问 AWS 时需要经过 FIPS 140-2 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-2 版》。

我们强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括使用控制台、API、AWS CLI 或 AWS SDK 处理 Verified Access 或其他 AWS 服务 时。在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供网址，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

## 传输中加密

Verified Access 将使用传输层安全性 (TLS) 1.2 或更高版本对通过 Internet 从最终用户传输到 Verified Access 端点的所有数据进行加密。

## 互联网络流量隐私

您可以配置 Verified Access 以限制对 VPC 中特定资源的访问。对于基于用户的身份验证，您还可以根据访问端点的用户组限制对网络各部分的访问。有关更多信息，请参阅 Verified Access 策略。

## AWS Verified Access 的静态数据加密

默认情况下，AWSVerified Access 使用 AWS 拥有的 KMS 密钥加密静态数据。当默认情况下对静态数据进行加密时，它有助于减少保护敏感数据所涉及的操作开销和复杂性。同时，它使您能够构建满足严格的加密合规性和监管要求的安全应用程序。以下各节详细介绍了 Verified Access 如何使用 KMS 密钥进行静态数据加密。

内容

- 为 Verified Access 资源指定客户托管密钥
- AWS Verified Access 加密上下文
- 监控 AWS Verified Access 的加密密钥

## Verified Access 和 KMS 密钥

### AWS 拥有的密钥

Verified Access 使用 KMS 密钥自动加密个人身份信息（PII）。这是默认操作，您无法自己查看、管理、使用或审核 AWS 拥有的密钥的使用情况。但是，您无需采取任何操作或更改任何程序即可保护用于加密数据的密钥。有关更多信息，请参阅 AWS Key Management Service 开发人员指南中的 AWS 拥有的密钥。

虽然您无法禁用此加密层或选择其他加密类型，但您可以在创建 Verified Access 资源时选择客户管理的密钥，从而在现有 AWS 拥有的加密密钥上添加第二层加密。

### 客户管理密钥

Verified Access 支持使用您创建和管理的对称客户托管密钥，在现有默认加密的基础上添加第二层加密。由于您可以完全控制这一层加密，因此可以执行以下任务：

- 制定和维护密钥策略
- 制定和维护 IAM 策略和授权
- 启用和禁用密钥策略
- 轮换密钥加密材料
- 添加标签
- 创建密钥别名
- 计划删除密钥

有关更多信息，请参阅 AWS Key Management Service 开发人员指南中的客户托管密钥。

> **Note**
>
> Verified Access 使用 AWS 拥有的密钥自动启用静态加密，从而免费保护个人身份数据。但是，当使用客户托管密钥时，将收取 AWS KMS 费用。有关定价的更多信息，请参阅 AWS Key Management Service 定价。

## 个人身份信息

下表汇总了 Verified Access 使用的个人身份信息（PII）以及加密方式。

| 数据类型 | AWS 自有密钥加密 | 客户托管密钥加密（可选） |
| --- | --- | --- |
| `Trust provider (user-type)`<br><br>用户类型的信任提供者包含 OIDC 选项，例如 AuthorizationEndpoint、、 UserInfoEndpoint ClientId ClientSecret、等，这些选项被视为 PII。 | 已启用 | 已启用 |
| `Trust provider (device-type)`<br><br>设备类型的信任提供者包含 TenantId，这被视为 PII。 | 已启用 | 已启用 |
| `Group policy`<br><br>在创建或修改 Verified Access 组时提供。包含授权访问请求的规则。可能包含 PII，例如用户名和电子邮件地址等。 | 已启用 | 已启用 |
| `Endpoint policy`<br><br>在创建或修改 Verified Access 端点时提供。包含授权访问请求的规则。可能包含 PII，例如用户名和电子邮件地址等。 | 已启用 | 已启用 |

## AWS Verified Access 如何使用 AWS KMS 中的授权

Verified Access 需要授权才能使用客户托管密钥。

当您创建使用客户托管密钥加密的已验证访问资源时，Verified Access 会通过向发送CreateGrant请求来代表您创建授权AWS KMS。AWS KMS 中的授权用于授予 Verified Access 访问您账户中的客户托管密钥的权限。

Verified Access 需要授权才能将客户托管密钥用于以下内部操作：

- 将 Decrypt 请求发送到 AWS KMS，以解密加密的数据密钥，以使它们可用于解密数据。
- 向发送RetireGrant请求AWS KMS以删除授权。

您可以随时撤销授予访问权限，或删除服务对客户托管密钥的访问权限。如果这样做，Verified Access 将无法访问由客户托管密钥加密的任何数据，这会影响依赖于该数据的操作。

## 将客户托管密钥用于 Verified Access

您可以使用 AWS Management Console 或 AWS KMS API 创建对称客户托管密钥。按照 AWS Key Management Service 开发人员指南中创建对称客户托管密钥的步骤进行操作。

密钥政策

密钥政策控制对客户托管式密钥的访问。每个客户管理型密钥必须只有一个密钥策略，其中包含确定谁可以使用密钥以及如何使用密钥的声明。创建客户管理型密钥时，可以指定密钥策略。有关更多信息，请参阅 AWS Key Management Service 开发人员指南中的管理对客户托管密钥的访问。

要将客户托管密钥与 Verified Access 资源结合使用，密钥政策中必须允许以下 API 操作：

- `kms:CreateGrant` – 向客户托管密钥添加授权。授予对指定 KMS 密钥的控制访问权限，该密钥允许访问 Verified Access 所需的授权操作。有关使用授权的更多信息，请参阅 AWS Key Management Service 开发人员指南。

  这允许 Verified Access 执行以下操作：

  - 调用 `GenerateDataKeyWithoutPlainText` 生成加密的数据密钥并将其存储，因为数据密钥不会立即用于加密。
  - 调用 `Decrypt` 来使用存储的加密数据密钥访问加密数据。
  - 设置停用主体以允许服务 `RetireGrant`。
- `kms:DescribeKey` – 提供客户托管密钥详细信息以允许 Verified Access 验证密钥。
- `kms:GenerateDataKey` – 允许 Verified Access 使用密钥加密数据。
- `kms:Decrypt` – 允许 Verified Access 解密已加密的数据密钥。

以下是可用于 Verified Access 的示例密钥政策。

```
"Statement" : [
    {
      "Sid" : "Allow access to principals authorized to use Verified Access",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "*"
      },
      "Action" : [
        "kms:DescribeKey",
        "kms:CreateGrant",
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource" : "*",
      "Condition" : {
        "StringEquals" : {
          "kms:ViaService" : "verified-access.region.amazonaws.com",
          "kms:CallerAccount" : "111122223333"
        }
    },
    {
      "Sid": "Allow access for key administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
       },
      "Action" : [
        "kms:*"
       ],
      "Resource": "arn:aws:kms:region:111122223333:key/key_ID"
    },
    {
      "Sid" : "Allow read-only access to key metadata to the account",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "arn:aws:iam::111122223333:root"
      },
      "Action" : [
        "kms:Describe*",
        "kms:Get*",
        "kms:List*",
        "kms:RevokeGrant"
```

```
      ],
      "Resource" : "*"
    }
  ]
```

有关在策略中指定权限的更多信息，请参阅 AWS Key Management Service 开发人员指南。

有关密钥访问故障排除的更多信息，请参阅 AWS Key Management Service 开发人员指南。

### 为 Verified Access 资源指定客户托管密钥

您可以指定客户托管密钥为以下资源提供第二层加密：

- Verified Access 组
- Verified Access 端点
- Verified Access 信任提供商

使用 AWS Management Console 创建这些资源中的任何一个时，可以在其他加密 -- 可选部分中指定客户托管密钥。在此过程中，选中自定义加密设置（高级）复选框，然后输入要使用的 AWS KMS 密钥 ID。也可以在修改现有资源时或使用 AWS CLI 来完成此操作。

> **Note**
>
> 如果用于向上述任何资源添加额外加密的客户托管密钥丢失，则将无法再访问这些资源的配置值。但是，可以通过使用AWS Management Console或AWS CLI修改资源来应用新的客户托管密钥并重置配置值。

## AWS Verified Access 加密上下文

加密上下文是一组可选的键值对，可以包含关于数据的额外上下文信息。AWS KMS 将加密上下文用作额外身份验证数据以支持身份验证加密。在请求中包含加密上下文以加密数据时，AWS KMS 将加密上下文绑定到加密的数据。要解密数据，请在请求中包含相同的加密上下文。

AWS Verified Access 加密上下文

Verified Access 在所有 AWS KMS 加密操作中使用相同的加密上下文，其中键为 `aws:verified-access:arn`，值为资源 Amazon 资源名称（ARN）。以下是 Verified Access 资源的加密上下文。

Verified Access 信任提供商

```
"encryptionContext": {
    "aws:verified-access:arn":
    "arn:aws:ec2:region:111122223333:VerifiedAccessTrustProviderId"
}
```

Verified Access 组

```
"encryptionContext": {
    "aws:verified-access:arn":
    "arn:aws:ec2:region:111122223333:VerifiedAccessGroupId"
}
```

Verified Access 端点

```
"encryptionContext": {
    "aws:verified-access:arn":
    "arn:aws:ec2:region:111122223333:VerifiedAccessEndpointId"
}
```

有关在授权或策略中使用加密上下文的更多信息，请参阅 AWS Key Management Service 开发人员指南中的加密上下文。

## 监控 AWS Verified Access 的加密密钥

将客户托管的 KMS 密钥与 AWS Verified Access 资源结合使用时，您可以使用 AWS CloudTrail 跟踪 Verified Access 发送到 AWS KMS 的请求。

以下示例是 `CreateGrant`、`RetireGrant`、`Decrypt`、`DescribeKey` 和 `GenerateDataKey` 的 AWS CloudTrail 事件，它们监控 Verified Access 调用的 KMS 操作以访问由客户托管的 KMS 密钥加密的数据：

CreateGrant

当使用客户托管密钥加密您的资源时，Verified Access 会代表您发送 `CreateGrant` 请求以访问您的 AWS 账户中的密钥。Verified Access 创建的授权特定于与客户托管密钥关联的资源。

以下示例事件记录了 `CreateGrant` 操作：

```
{
    "eventVersion": "1.08",
    "userIdentity": {
```

```
        "type": "AssumedRole",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-09-11T16:27:12Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "verified-access.amazonaws.com"
    },
    "eventTime": "2023-09-11T16:41:42Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "ca-central-1",
    "sourceIPAddress": "verified-access.amazonaws.com",
    "userAgent": "verified-access.amazonaws.com",
    "requestParameters": {
        "operations": [
            "Decrypt",
            "RetireGrant",
            "GenerateDataKey"
        ],
        "keyId": "arn:aws:kms:ca-central-1:111122223333:key/5ed79e7f-88c9-420c-
ae1a-61ee87104dae",
        "constraints": {
            "encryptionContextSubset": {
                "aws:verified-access:arn": "arn:aws:ec2:ca-
central-1:111122223333:verified-access-trust-provider/vatp-0e54f581e2e5c97a2"
            }
        },
        "granteePrincipal": "verified-access.ca-central-1.amazonaws.com",
        "retiringPrincipal": "verified-access.ca-central-1.amazonaws.com"
    },
```

```
    "responseElements": {
        "grantId":
 "e5a050fff9893ba1c43f83fddf61e5f9988f579beaadd6d4ad6d1df07df6048f",
        "keyId": "arn:aws:kms:ca-central-1:111122223333:key/5ed79e7f-88c9-420c-
ae1a-61ee87104dae"
    },
    "requestID": "0faa837e-5c69-4189-9736-3957278e6444",
    "eventID": "1b6dd8b8-cbee-4a83-9b9d-d95fa5f6fd08",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:ca-central-1:111122223333:key/5ed79e7f-88c9-420c-
ae1a-61ee87104dae"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

### RetireGrant

当您删除资源时，Verified Access 使用 `RetireGrant` 操作来移除授权。

以下示例事件记录了 `RetireGrant` 操作：

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
```

```
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-09-11T16:42:33Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "verified-access.amazonaws.com"
    },
    "eventTime": "2023-09-11T16:47:53Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RetireGrant",
    "awsRegion": "ca-central-1",
    "sourceIPAddress": "verified-access.amazonaws.com",
    "userAgent": "verified-access.amazonaws.com",
    "requestParameters": null,
    "responseElements": {
        "keyId": "arn:aws:kms:ca-central-1:111122223333:key/5ed79e7f-88c9-420c-
ae1a-61ee87104dae"
    },
    "additionalEventData": {
        "grantId":
 "b35e66f9bacb266cec214fcaa353c9cf750785e28773e61ba6f434d8c5c7632f"
    },
    "requestID": "7d4a31c2-d426-434b-8f86-336532a70462",
    "eventID": "17edc343-f25b-43d4-bbff-150d8fff4cf8",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:ca-central-1:111122223333:key/5ed79e7f-88c9-420c-
ae1a-61ee87104dae"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


## Decrypt

Verified Access 调用 `Decrypt` 操作以使用存储的加密数据密钥来访问加密数据。

以下示例事件记录了 Decrypt 操作：


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-09-11T17:19:33Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "verified-access.amazonaws.com"
    },
    "eventTime": "2023-09-11T17:47:05Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "ca-central-1",
    "sourceIPAddress": "verified-access.amazonaws.com",
    "userAgent": "verified-access.amazonaws.com",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "keyId": "arn:aws:kms:ca-
central-1:111122223333:key/380d006e-706a-464b-99c5-68768297114e",
        "encryptionContext": {
            "aws:verified-access:arn": "arn:aws:ec2:ca-
central-1:111122223333:verified-access-trust-provider/vatp-00f20a4e455e9340f",
            "aws-crypto-public-key": "AkK+vi1W/
acBKv7OR8p2DeUrA8EgpTffSrjBqNucODuBYhyZ3hlMuYYJz9x7CwQWZw=="
        }
    },
    "responseElements": null,
```

```
    "requestID": "2e920fd3-f2f6-41b2-a5e7-2c2cb6f853a9",
    "eventID": "3329e0a3-bcfb-44cf-9813-8106d6eee31d",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:ca-
central-1:111122223333:key/380d006e-706a-464b-99c5-68768297114e"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

### DescribeKey

Verified Access 使用 `DescribeKey` 操作来验证与您的资源关联的客户托管密钥是否存在于账户和区域中。

以下示例事件记录了 `DescribeKey` 操作：

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-09-11T17:19:33Z",
                "mfaAuthenticated": "false"
```

```
            }
        },
        "invokedBy": "verified-access.amazonaws.com"
    },
    "eventTime": "2023-09-11T17:46:48Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "ca-central-1",
    "sourceIPAddress": "verified-access.amazonaws.com",
    "userAgent": "verified-access.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:ca-
central-1:111122223333:key/380d006e-706a-464b-99c5-68768297114e"
    },
    "responseElements": null,
    "requestID": "5b127082-6691-48fa-bfb0-4d40e1503636",
    "eventID": "ffcfc2bb-f94b-4c00-b6fb-feac77daff2a",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:ca-
central-1:111122223333:key/380d006e-706a-464b-99c5-68768297114e"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

### GenerateDataKey

以下示例事件记录 GenerateDataKey 操作：

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-09-11T17:19:33Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "verified-access.amazonaws.com"
    },
    "eventTime": "2023-09-11T17:46:49Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "ca-central-1",
    "sourceIPAddress": "verified-access.amazonaws.com",
    "userAgent": "verified-access.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:verified-access:arn": "arn:aws:ec2:ca-
central-1:111122223333:verified-access-trust-provider/vatp-00f20a4e455e9340f",
            "aws-crypto-public-key": "A/ATGxaYatPUlOtM+l/mfDndkzHUmX5Hav+29IlIm
+JRBKFuXf24ulztmOIsqFQliw=="
        },
        "numberOfBytes": 32,
        "keyId": "arn:aws:kms:ca-
central-1:111122223333:key/380d006e-706a-464b-99c5-68768297114e"
    },
    "responseElements": null,
    "requestID": "06535808-7cce-4ae1-ab40-e3afbf158a43",
    "eventID": "1ce79601-5a5e-412c-90b3-978925036526",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:ca-
central-1:111122223333:key/380d006e-706a-464b-99c5-68768297114e"
        }
```

```
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

# AWS 验证访问权限的身份和访问管理

AWS Identity and Access Management (IAM) AWS 服务 可帮助管理员安全地控制对 AWS 资源的访问权限。IAM 管理员控制可以通过身份验证（登录）和授权（具有权限）使用 Verified Access 资源的人员。您可以使用 IAM AWS 服务 ，无需支付额外费用。

主题



## 受众

您的使用方式 AWS Identity and Access Management (IAM) 会有所不同，具体取决于您在验证访问权限中所做的工作。

服务用户 – 如果使用 Verified Access 服务来完成任务，则您的管理员会为您提供所需的凭证和权限。当您使用更多 Verified Access 功能来完成工作时，您可能需要额外权限。了解如何管理访问权限有助于您向管理员请求适合的权限。如果您无法访问 Verified Access 中的功能，请参阅 对 AWS 已验证的访问身份和访问进行故障排除。

服务管理员 – 如果您在公司负责管理 Verified Access 资源，则您可能具有 Verified Access 的完整访问权限。您有责任确定您的服务用户应访问哪些 Verified Access 功能和资源。然后，您必须向 IAM 管理员提交请求以更改服务用户的权限。请查看该页面上的信息以了解 IAM 的基本概念。要了解有关您的

公司如何将 IAM 与 Verified Access 搭配使用的更多信息，请参阅 AWS 验证访问权限如何与 IAM 配合使用。

IAM 管理员 – 如果您是 IAM 管理员，您可能希望了解有关如何编写策略以管理对 Verified Access 的访问权限的详细信息。要查看您可在 IAM 中使用的 Verified Access 基于身份的策略示例，请参阅 已验证访问权限的基于身份的 AWS 策略示例。

## 使用身份进行身份验证

身份验证是您 AWS 使用身份凭证登录的方式。您必须以 IAM 用户身份或通过担 AWS 账户根用户任 IAM 角色进行身份验证（登录 AWS）。

您可以使用通过身份源提供的凭据以 AWS 联合身份登录。 AWS IAM Identity Center （IAM Identity Center）用户、贵公司的单点登录身份验证以及您的 Google 或 Facebook 凭据就是联合身份的示例。当您以联合身份登录时，您的管理员以前使用 IAM 角色设置了身份联合验证。当你使用联合访问 AWS 时，你就是在间接扮演一个角色。

根据您的用户类型，您可以登录 AWS Management Console 或 AWS 访问门户。有关登录的更多信息 AWS，请参阅《AWS 登录 用户指南》中的如何登录到您 AWS 账户的。

如果您 AWS 以编程方式访问，则会 AWS 提供软件开发套件 (SDK) 和命令行接口 (CLI)，以便使用您的凭据对请求进行加密签名。如果您不使用 AWS 工具，则必须自己签署请求。有关使用推荐的方法自行签署请求的更多信息，请参阅 IAM 用户指南中的签署 AWS API 请求。

无论使用何种身份验证方法，您可能需要提供其他安全信息。例如， AWS 建议您使用多重身份验证 (MFA) 来提高账户的安全性。要了解更多信息，请参阅《AWS IAM Identity Center 用户指南》中的 多重身份验证和《IAM 用户指南》中的 在 AWS中使用多重身份验证（MFA）。

### AWS 账户 root 用户

创建时 AWS 账户，首先要有一个登录身份，该身份可以完全访问账户中的所有资源 AWS 服务 和资源。此身份被称为 AWS 账户 root 用户，使用您创建账户时使用的电子邮件地址和密码登录即可访问该身份。强烈建议您不要使用根用户执行日常任务。保护好根用户凭证，并使用这些凭证来执行仅根用户可以执行的任务。有关要求您以根用户身份登录的任务的完整列表，请参阅《IAM 用户指南》中的需要根用户凭证的任务。

### 联合身份

作为最佳实践，要求人类用户（包括需要管理员访问权限的用户）使用与身份提供商的联合身份验证 AWS 服务 通过临时证书进行访问。

联合身份是指您的企业用户目录、Web 身份提供商、Identity C enter 目录中的用户，或者任何使用 AWS 服务 通过身份源提供的凭据进行访问的用户。 AWS Directory Service当联合身份访问时 AWS 账户，他们将扮演角色，角色提供临时证书。

要集中管理访问权限，建议您使用 AWS IAM Identity Center。您可以在 IAM Identity Center 中创建用户和群组，也可以连接并同步到您自己的身份源中的一组用户和群组，以便在您的所有 AWS 账户 和应用程序中使用。有关 IAM Identity Center 的信息，请参阅《AWS IAM Identity Center 用户指南》中的什么是 IAM Identity Center？

## IAM 用户和群组

I AM 用户是您 AWS 账户 内部对个人或应用程序具有特定权限的身份。在可能的情况下，我们建议使用临时凭证，而不是创建具有长期凭证（如密码和访问密钥）的 IAM 用户。但是，如果您有一些特定的使用场景需要长期凭证以及 IAM 用户，建议您轮换访问密钥。有关更多信息，请参阅《IAM 用户指南》中的 对于需要长期凭证的使用场景定期轮换访问密钥。

IAM 组是一个指定一组 IAM 用户的身份。您不能使用组的身份登录。您可以使用组来一次性为多个用户指定权限。如果有大量用户，使用组可以更轻松地管理用户权限。例如，您可能具有一个名为 IAMAdmins 的组，并为该组授予权限以管理 IAM 资源。

用户与角色不同。用户唯一地与某个人员或应用程序关联，而角色旨在让需要它的任何人代入。用户具有永久的长期凭证，而角色提供临时凭证。要了解更多信息，请参阅《IAM 用户指南》中的何时创建 IAM 用户（而不是角色）。

## IAM 角色

I AM 角色是您内部具有特定权限 AWS 账户 的身份。它类似于 IAM 用户，但与特定人员不关联。您可以 AWS Management Console 通过切换角色在中临时担任 IAM 角色。您可以通过调用 AWS CLI 或 AWS API 操作或使用自定义 URL 来代入角色。有关使用角色的方法的更多信息，请参阅《IAM 用户指南》中的使用 IAM 角色。

具有临时凭证的 IAM 角色在以下情况下很有用：

- 联合用户访问 – 要向联合身份分配权限，请创建角色并为角色定义权限。当联合身份进行身份验证时，该身份将与角色相关联并被授予由此角色定义的权限。有关联合身份验证的角色的信息，请参阅《IAM 用户指南》中的为第三方身份提供商创建角色。如果您使用 IAM Identity Center，则需要配置权限集。为控制您的身份在进行身份验证后可以访问的内容，IAM Identity Center 将权限集与 IAM 中的角色相关联。有关权限集的信息，请参阅《AWS IAM Identity Center 用户指南》中的 权限集。
- 临时 IAM 用户权限 – IAM 用户可代入 IAM 用户或角色，以暂时获得针对特定任务的不同权限。

- 跨账户存取 – 您可以使用 IAM 角色以允许不同账户中的某个人（可信主体）访问您的账户中的资源。角色是授予跨账户访问权限的主要方式。但是，对于某些资源 AWS 服务，您可以将策略直接附加到资源（而不是使用角色作为代理）。要了解用于跨账户访问的角色和基于资源的策略之间的差别，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。
- 跨服务访问 — 有些 AWS 服务 使用其他 AWS 服务服务中的功能。例如，当您在某个服务中进行调用时，该服务通常会在 Amazon EC2 中运行应用程序或在 Simple Storage Service (Amazon S3) 中存储对象。服务可能会使用发出调用的主体的权限、使用服务角色或使用服务相关角色来执行此操作。
  - 转发访问会话 (FAS) — 当您使用 IAM 用户或角色在中执行操作时 AWS，您被视为委托人。使用某些服务时，您可能会执行一个操作，然后此操作在其他服务中启动另一个操作。FAS 使用调用委托人的权限以及 AWS 服务 向下游服务发出请求的请求。 AWS 服务只有当服务收到需要与其他 AWS 服务 或资源交互才能完成的请求时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两个操作的权限。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。
  - 服务角色 - 服务角色是服务代表您在您的账户中执行操作而分派的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 AWS 服务委派权限的角色。
  - 服务相关角色-服务相关角色是一种链接到的服务角色。 AWS 服务服务可以代入代表您执行操作的角色。服务相关角色出现在您的中 AWS 账户 ，并且归服务所有。IAM 管理员可以查看但不能编辑服务相关角色的权限。
- 在 A@@ mazon EC2 上运行的应用程序 — 您可以使用 IAM 角色管理在 EC2 实例上运行并发出 AWS CLI 或 AWS API 请求的应用程序的临时证书。这优先于在 EC2 实例中存储访问密钥。要向 EC2 实例分配 AWS 角色并使其可供其所有应用程序使用，您需要创建附加到该实例的实例配置文件。实例配置文件包含角色，并使 EC2 实例上运行的程序能够获得临时凭证。有关更多信息，请参阅《IAM 用户指南》中的使用 IAM 角色为 Amazon EC2 实例上运行的应用程序授予权限。

要了解是使用 IAM 角色还是 IAM 用户，请参阅IAM 用户指南中的何时创建 IAM 角色（而不是用户）。

## 使用策略管理访问

您可以 AWS 通过创建策略并将其附加到 AWS 身份或资源来控制中的访问权限。策略是其中的一个对象 AWS ，当与身份或资源关联时，它会定义其权限。 AWS 在委托人（用户、root 用户或角色会话）发出请求时评估这些策略。策略中的权限确定是允许还是拒绝请求。大多数策略都以 JSON 文档的 AWS 形式存储在中。有关 JSON 策略文档的结构和内容的更多信息，请参阅 IAM 用户指南中的 JSON 策略概览。

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

默认情况下，用户和角色没有权限。要授予用户对所需资源执行操作的权限，IAM 管理员可以创建 IAM 策略。管理员随后可以向角色添加 IAM policy，用户可以代入角色。

IAM 策略定义操作的权限，无关乎您使用哪种方法执行操作。例如，假设您有一个允许 `iam:GetRole` 操作的策略。拥有该策略的用户可以从 AWS Management Console AWS CLI、或 AWS API 获取角色信息。

## 基于身份的策略

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的创建 IAM policy。

基于身份的策略可以进一步归类为内联策略或托管策略。内联策略直接嵌入单个用户、组或角色中。托管策略是独立的策略，您可以将其附加到中的多个用户、群组和角色 AWS 账户。托管策略包括 AWS 托管策略和客户托管策略。要了解如何在托管式策略和内联策略之间进行选择，请参阅 IAM 用户指南中的在托管式策略与内联策略之间进行选择。

## 基于资源的策略

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件下执行。您必须在基于资源的策略中指定主体。委托人可以包括账户、用户、角色、联合用户或 AWS 服务。

基于资源的策略是位于该服务中的内联策略。您不能在基于资源的策略中使用 IAM 中的 AWS 托管策略。

## 访问控制列表 (ACL)

访问控制列表（ACL）控制哪些主体（账户成员、用户或角色）有权访问资源。ACL 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

Amazon S3 和 Amazon VPC 就是支持 ACL 的服务示例。 AWS WAF要了解有关 ACL 的更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的访问控制列表（ACL）概览。

### 其他策略类型

AWS 支持其他不太常见的策略类型。这些策略类型可以设置更常用的策略类型向您授予的最大权限。

- 权限边界：权限边界是一个高级特征，用于设置基于身份的策略可以为 IAM 实体（IAM 用户或角色）授予的最大权限。您可为实体设置权限边界。这些结果权限是实体基于身份的策略及其权限边界的交集。在 `Principal` 中指定用户或角色的基于资源的策略不受权限边界限制。任一项策略中的显式拒绝将覆盖允许。有关权限边界的更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。
- 服务控制策略 (SCP)-SCP 是 JSON 策略，用于指定组织或组织单位 (OU) 的最大权限。 AWS Organizations AWS Organizations 是一项用于对您的企业拥有的多 AWS 账户 项进行分组和集中管理的服务。如果在组织内启用了所有功能，则可对任意或全部账户应用服务控制策略 (SCP)。SCP 限制成员账户中的实体（包括每个 AWS 账户根用户实体）的权限。有关 Organizations 和 SCP 的更多信息，请参阅《AWS Organizations 用户指南》中的 SCP 的工作原理。
- 会话策略 – 会话策略是当您以编程方式为角色或联合用户创建临时会话时作为参数传递的高级策略。结果会话的权限是用户或角色的基于身份的策略和会话策略的交集。权限也可以来自基于资源的策略。任一项策略中的显式拒绝将覆盖允许。有关更多信息，请参阅《IAM 用户指南》中的会话策略。

### 多个策略类型

当多个类型的策略应用于一个请求时，生成的权限更加复杂和难以理解。要了解在涉及多种策略类型时如何 AWS 确定是否允许请求，请参阅 IAM 用户指南中的策略评估逻辑。

## AWS 验证访问权限如何与 IAM 配合使用

在使用 IAM 管理对 Verified Access 的访问之前，您应该了解哪些 IAM 功能可与 Verified Access 配合使用。

可在 AWS 已验证访问权限中使用的 IAM 功能

| IAM 功能 | Verified Access 支持 |
| --- | --- |
| 基于身份的策略 | 是 |
| 基于资源的策略 | 否 |

| IAM 功能 | Verified Access 支持 |
| --- | --- |
| 策略操作 | 是 |
| 策略资源 | 是 |
| 策略条件键 | 是 |
| ACL | 否 |
| ABAC（策略中的标签） | 部分 |
| 临时凭证 | 是 |
| 主体权限 | 是 |
| 服务角色 | 否 |
| 服务相关角色 | 是 |

要全面了解已验证访问权限和其他 AWS 服务如何与大多数 IAM 功能配合使用，请参阅 IAM 用户指南中的与 IAM 配合使用的AWS 服务。

## Verified Access 的基于身份的策略

| | |
| --- | --- |
| 支持基于身份的策略 | 是 |

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的创建 IAM policy。

通过使用 IAM 基于身份的策略，您可以指定允许或拒绝的操作和资源以及允许或拒绝操作的条件。您无法在基于身份的策略中指定主体，因为它适用于其附加的用户或角色。要了解可在 JSON 策略中使用的所有元素，请参阅《IAM 用户指南》中的 IAM JSON 策略元素引用。

### Verified Access 的基于身份的策略示例

要查看 Verified Access 基于身份的策略的示例，请参阅 已验证访问权限的基于身份的 AWS 策略示例。

## Verified Access 内基于资源的策略

| | |
|---|---|
| 支持基于资源的策略 | 否 |

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件下执行。您必须在基于资源的策略中指定主体。委托人可以包括账户、用户、角色、联合用户或 AWS 服务。

要启用跨账户存取，您可以将整个账户或其他账户中的 IAM 实体指定为基于资源的策略中的主体。将跨账户主体添加到基于资源的策略只是建立信任关系工作的一半而已。当委托人和资源处于不同位置时 AWS 账户，可信账户中的 IAM 管理员还必须向委托人实体（用户或角色）授予访问资源的权限。他们通过将基于身份的策略附加到实体以授予权限。但是，如果基于资源的策略向同一个账户中的主体授予访问权限，则不需要额外的基于身份的策略。有关更多信息，请参阅 IAM 用户指南中的跨账户访问 IAM 中的资源。

## Verified Access 的策略操作

| | |
|---|---|
| 支持策略操作 | 是 |

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

JSON 策略的 `Action` 元素描述可用于在策略中允许或拒绝访问的操作。策略操作通常与关联的 AWS API 操作同名。有一些例外情况，例如没有匹配 API 操作的仅限权限 操作。还有一些操作需要在策略中执行多个操作。这些附加操作称为相关操作。

在策略中包含操作以授予执行关联操作的权限。

要查看 Verified Access 操作的列表，请参阅《服务授权参考》中的 Amazon EC2 定义的操作。

Verified Access 中的策略操作在操作前使用以下前缀：

```
ec2
```

要在单个语句中指定多项操作，请使用逗号将它们隔开。

```
"Action": [
    "ec2:action1",
    "ec2:action2"
]
```

要查看 Verified Access 基于身份的策略的示例，请参阅 已验证访问权限的基于身份的 AWS 策略示例。

## Verified Access 的策略资源

| | |
|---|---|
| 支持策略资源 | 是 |

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体 可以对什么资源执行操作，以及在什么条件下执行。

`Resource` JSON 策略元素指定要向其应用操作的一个或多个对象。语句必须包含 `Resource` 或 `NotResource` 元素。作为最佳实践，请使用其 Amazon 资源名称 (ARN) 指定资源。对于支持特定资源类型（称为资源级权限）的操作，您可以执行此操作。

对于不支持资源级权限的操作（如列出操作），请使用通配符 (*) 指示语句应用于所有资源。

```
"Resource": "*"
```

要查看 Verified Access 资源类型及其 ARN 的列表，请参阅《服务授权参考》中的 Amazon EC2 定义的资源。要了解您可以在哪些操作中指定每个资源的 ARN，请参阅 Amazon EC2 定义的操作。

要查看 Verified Access 基于身份的策略的示例，请参阅 已验证访问权限的基于身份的 AWS 策略示例。

## Verified Access 的策略条件键

| | |
|---|---|
| 支持特定于服务的策略条件键 | 是 |

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

在 Condition 元素（或 Condition 块）中，可以指定语句生效的条件。Condition 元素是可选的。您可以创建使用条件运算符（例如，等于或小于）的条件表达式，以使策略中的条件与请求中的值相匹配。

如果您在一个语句中指定多个 Condition 元素，或在单个 Condition 元素中指定多个键，则 AWS 使用逻辑 AND 运算评估它们。如果您为单个条件键指定多个值，则使用逻辑OR运算来 AWS 评估条件。在授予语句的权限之前必须满足所有的条件。

在指定条件时，您也可以使用占位符变量。例如，只有在使用 IAM 用户名标记 IAM 用户时，您才能为其授予访问资源的权限。有关更多信息，请参阅《IAM 用户指南》中的 IAM policy 元素：变量和标签。

AWS 支持全局条件密钥和特定于服务的条件密钥。要查看所有 AWS 全局条件键，请参阅 IAM 用户指南中的AWS 全局条件上下文密钥。

要查看 Verified Access 条件键的列表，请参阅《服务授权参考》中的 Amazon EC2 的条件键。要了解您可以对哪些操作和资源使用条件键，请参阅 Amazon EC2 定义的操作。

要查看 Verified Access 基于身份的策略的示例，请参阅 已验证访问权限的基于身份的 AWS 策略示例。

## Verified Access 中的 ACL

| | |
|---|---|
| 支持 ACL | 否 |

访问控制列表 (ACL) 控制哪些主体（账户成员、用户或角色）有权访问资源。ACL 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

## ABAC 与 Verified Access

| | |
|---|---|
| 支持 ABAC（策略中的标签） | 部分 |

基于属性的访问权限控制（ABAC）是一种授权策略，该策略基于属性来定义权限。在中 AWS，这些属性称为标签。您可以向 IAM 实体（用户或角色）和许多 AWS 资源附加标签。标记实体和资源是

ABAC 的第一步。然后设计 ABAC 策略，以在主体的标签与他们尝试访问的资源标签匹配时允许操作。

ABAC 在快速增长的环境中非常有用，并在策略管理变得繁琐的情况下可以提供帮助。

要基于标签控制访问，您需要使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 条件键在策略的条件元素中提供标签信息。

如果某个服务对于每种资源类型都支持所有这三个条件键，则对于该服务，该值为是。如果某个服务仅对于部分资源类型支持所有这三个条件键，则该值为部分。

有关 ABAC 的更多信息,请参阅《IAM 用户指南》中的什么是 ABAC？。要查看设置 ABAC 步骤的教程，请参阅《IAM 用户指南》中的使用基于属性的访问权限控制（ABAC）。

## 将临时凭证用于 Verified Access

| | |
|---|---|
| 支持临时凭证 | 是 |

当你使用临时证书登录时，有些 AWS 服务 不起作用。有关更多信息，包括哪些 AWS 服务 适用于临时证书，请参阅 IAM 用户指南中的AWS 服务 与 IA M 配合使用的信息。

如果您使用除用户名和密码之外的任何方法登录，则 AWS Management Console 使用的是临时证书。例如，当您 AWS 使用公司的单点登录 (SSO) 链接进行访问时，该过程会自动创建临时证书。当您以用户身份登录控制台，然后切换角色时，您还会自动创建临时凭证。有关切换角色的更多信息，请参阅《IAM 用户指南》中的 切换到角色（控制台）。

您可以使用 AWS CLI 或 AWS API 手动创建临时证书。然后，您可以使用这些临时证书进行访问 AWS。 AWS 建议您动态生成临时证书，而不是使用长期访问密钥。有关更多信息，请参阅 IAM 中的临时安全凭证。

## Verified Access 的跨服务主体权限

| | |
|---|---|
| 支持转发访问会话 (FAS) | 是 |

当您使用 IAM 用户或角色在中执行操作时 AWS，您被视为委托人。使用某些服务时，您可能会执行一个操作，然后此操作在其他服务中启动另一个操作。FAS 使用调用委托人的权限以及 AWS 服务 向下游服务发出请求的请求。 AWS 服务只有当服务收到需要与其他 AWS 服务 或资源交互才能完成的请求

时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两个操作的权限。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。

### Verified Access 的服务角色

| | |
|---|---|
| 支持服务角色 | 否 |

服务角色是由一项服务担任、代表您执行操作的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 AWS 服务委派权限的角色。

### Verified Access 的服务相关角色

| | |
|---|---|
| 支持服务相关角色 | 是 |

服务相关角色是一种链接到的服务角色。 AWS 服务服务可以代入代表您执行操作的角色。服务相关角色出现在您的中 AWS 账户 ，并且归服务所有。IAM 管理员可以查看但不能编辑服务相关角色的权限。

有关创建或管理 Verified Access 服务相关角色的详细信息，请参阅 将服务相关角色用于 Verified Access。

## 已验证访问权限的基于身份的 AWS 策略示例

默认情况下，用户和角色没有创建或修改 Verified Access 资源的权限。他们也无法使用 AWS Management Console、 AWS Command Line Interface (AWS CLI) 或 AWS API 执行任务。要授予用户对所需资源执行操作的权限，IAM 管理员可以创建 IAM policy。管理员随后可以向角色添加 IAM 策略，用户可以代入角色。

要了解如何使用这些示例 JSON 策略文档创建基于 IAM 身份的策略，请参阅 IAM 用户指南中的 创建 IAM 策略。

有关 Verified Access 定义的操作和资源类型的详细信息，包括每种资源类型的 ARN 格式，请参阅《服务授权参考》中的 Amazon EC2 的操作、资源和条件键。

主题

- 策略最佳实践

- 创建 Verified Access 实例的策略
- 允许用户查看他们自己的权限

## 策略最佳实践

基于身份的策略确定某个人是否可以创建、访问或删除您账户中的 Verified Access 资源。这些操作可能会使 AWS 账户产生成本。创建或编辑基于身份的策略时，请遵循以下指南和建议：

- 开始使用 AWS 托管策略并转向最低权限权限 — 要开始向用户和工作负载授予权限，请使用为许多常见用例授予权限的AWS 托管策略。它们在你的版本中可用 AWS 账户。我们建议您通过定义针对您的用例的 AWS 客户托管策略来进一步减少权限。有关更多信息，请参阅《IAM 用户指南》中的 AWS 托管策略 或 工作职能的AWS 托管策略。
- 应用最低权限 – 在使用 IAM policy 设置权限时，请仅授予执行任务所需的权限。为此，您可以定义在特定条件下可以对特定资源执行的操作，也称为最低权限许可。有关使用 IAM 应用权限的更多信息，请参阅《IAM 用户指南》中的 IAM 中的策略和权限。
- 使用 IAM policy 中的条件进一步限制访问权限 – 您可以向策略添加条件来限制对操作和资源的访问。例如，您可以编写策略条件来指定必须使用 SSL 发送所有请求。如果服务操作是通过特定 AWS 服务的（例如）使用的，则也可以使用条件来授予对服务操作的访问权限 AWS CloudFormation。有关更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素：条件 。
- 使用 IAM Access Analyzer 验证您的 IAM policy，以确保权限的安全性和功能性 – IAM Access Analyzer 会验证新策略和现有策略，以确保策略符合 IAM policy语言 (JSON) 和 IAM 最佳实践。IAM Access Analyzer 提供 100 多项策略检查和可操作的建议，以帮助您制定安全且功能性强的策略。有关更多信息，请参阅《IAM 用户指南》中的 IAM Acess Analyzer 策略验证。
- 需要多重身份验证 (MFA)-如果 AWS 账户您的场景需要 IAM 用户或根用户，请启用 MFA 以提高安全性。若要在调用 API 操作时需要 MFA，请将 MFA 条件添加到您的策略中。有关更多信息，请参阅《IAM 用户指南》中的 配置受 MFA 保护的 API 访问。

有关 IAM 中的最佳实操的更多信息，请参阅《IAM 用户指南》中的 IAM 中的安全最佳实操。

## 创建 Verified Access 实例的策略

要创建 Verified Access 实例，IAM 主体需要将此附加语句添加到其 IAM policy 中。

```
{
    "Effect": "Allow",
    "Action": "verified-access:AllowVerifiedAccess",
```

```
    "Resource": "*"
}
```

> **Note**
>
> verified-access:AllowVerifiedAccess 是一个仅限操作的虚拟 API。它不支持基于资源、标签或条件键的授权。对 ec2:CreateVerifiedAccessInstance API 操作使用基于资源、标签或条件键的授权。

创建 Verified Access 实例的策略示例。在此示例中，123456789012是 AWS 账号，us-east-1是 AWS 区域。

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateVerifiedAccessInstance",
            "Resource": "arn:aws:ec2:us-east-1:123456789012:verified-access-instance/*"
        },
        {
            "Effect": "Allow",
            "Action": "verified-access:AllowVerifiedAccess",
            "Resource": "*"
        }
    ]
}
```

## 允许用户查看他们自己的权限

该示例说明了您如何创建策略，以允许 IAM 用户查看附加到其用户身份的内联和托管策略。此策略包括在控制台上或使用 AWS CLI 或 AWS API 以编程方式完成此操作的权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

# 对 AWS 已验证的访问身份和访问进行故障排除

使用以下信息可帮助您诊断和修复在使用 Verified Access 和 IAM 时可能遇到的常见问题。

问题

- 我无权在 Verified Access 中执行操作
- 我无权执行 iam：PassRole
- 我想允许我以外的人访问我的 AWS 账户 “已验证访问权限” 资源

## 我无权在 Verified Access 中执行操作

如果您收到错误提示，表明您无权执行某个操作，则您必须更新策略以允许执行该操作。

当 mateojackson IAM 用户尝试使用控制台查看有关虚构 *my-example-widget* 资源的详细信息，但不拥有虚构 ec2:*GetWidget* 权限时，会发生以下示例错误。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 ec2:GetWidget on resource: my-example-widget
```

在此情况下，必须更新 mateojackson 用户的策略，以允许使用 ec2:*GetWidget* 操作访问 *my-example-widget* 资源。

如果您需要帮助，请联系您的 AWS 管理员。您的管理员是提供登录凭证的人。

## 我无权执行 iam：PassRole

如果您收到一个错误，表明您无权执行 iam:PassRole 操作，则必须更新策略以允许您将角色传递给 Verified Access。

有些 AWS 服务 允许您将现有角色传递给该服务，而不是创建新的服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

当名为 marymajor 的 IAM 用户尝试使用控制台在 Verified Access 中执行操作时，会发生以下示例错误。但是，服务必须具有服务角色所授予的权限才可执行此操作。Mary 不具有将角色传递到服务的权限。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在这种情况下，必须更新 Mary 的策略以允许她执行 iam:PassRole 操作。

如果您需要帮助，请联系您的 AWS 管理员。您的管理员是提供登录凭证的人。

## 我想允许我以外的人访问我的 AWS 账户 “已验证访问权限” 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以担任角色。对于支持基于资源的策略或访问控制列表（ACL）的服务，您可以使用这些策略向人员授予对您的资源的访问权。

要了解更多信息，请参阅以下内容：

- 要了解 Verified Access 是否支持这些功能，请参阅 AWS 验证访问权限如何与 IAM 配合使用。
- 要了解如何提供对您拥有的资源的访问权限 AWS 账户 ，请参阅 IAM 用户指南中的向您拥有 AWS 账户 的另一个 IAM 用户提供访问权限。
- 要了解如何向第三方提供对您的资源的访问权限 AWS 账户，请参阅 IAM 用户指南中的向第三方提供访问权限。 AWS 账户

- 要了解如何通过联合身份验证提供访问权限，请参阅《IAM 用户指南》中的为经过外部身份验证的用户（联合身份验证）提供访问权限。
- 要了解使用角色和基于资源的策略进行跨账户访问的区别，请参阅 IA M 用户指南中的跨账户资源访问。

## 将服务相关角色用于 Verified Access

AWS Verified Access 使用 AWS Identity and Access Management (IAM) 服务相关角色。服务相关角色是一种独特类型的 IAM 角色，它与 Verified Access 直接相关。服务相关角色由 Verified Access 预定义，包括相应服务代表您调用其他 AWS 服务 所需的所有权限。

服务相关角色可让您更轻松地设置 Verified Access，因为您不必手动添加必要的权限。Verified Access 定义其服务相关角色的权限，除非另外定义，否则只有 Verified Access 可以代入该角色。定义的权限包括信任策略和权限策略，并且此权限策略不能附加到任何其他 IAM 实体。

有关支持服务相关角色的其他服务的信息，请参阅与 IAM 配合使用的 AWS 服务，并查找 Service-linked roles（服务相关角色）列中显示为 Yes（是）的服务。请选择 Yes 与查看该服务的服务相关角色文档的链接。

### Verified Access 的服务相关角色权限

Verified Access 使用名为 AWSServiceRoleForVPCVerifiedAccess 的服务相关角色在您的账户中预置使用该服务所需的资源。

AWSServiceRoleForVPCVerifiedAccess 服务相关角色信任以下服务以担任该角色：

- `verified-access.amazonaws.com`

名为 AWSVPCVerifiedAccessServiceRolePolicy 的角色权限策略允许 Verified Access 对指定资源完成以下操作：

- 对所有子网和安全组以及所有带有 `VerifiedAccessManaged=true` 标签的网络接口执行操作 `ec2:CreateNetworkInterface`
- 创建时对所有网络接口执行操作 `ec2:CreateTags`
- 对所有带有 `VerifiedAccessManaged=true` 标签的网络接口执行操作 `ec2:DeleteNetworkInterface`
- 对所有安全组以及所有带有 `VerifiedAccessManaged=true` 标签的网络接口执行操作 `ec2:ModifyNetworkInterfaceAttribute`

您还可以在 AWS Management Console AWSVPCVerifiedAccessServiceRolePolicy 中查看此策略的权限，或者在 AWS 托管式策略参考指南中查看 AWSVPCVerifiedAccessServiceRolePolicy 策略。

必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。有关更多信息，请参阅 IAM 用户指南中的服务相关角色权限。

### 为 Verified Access 创建服务相关角色

无需手动创建服务相关角色。在 AWS Management Console、AWS CLI 或 AWS API 中调用 CreateVerifiedAccessEndpoint 时，Verified Access 将创建服务相关角色。

如果删除此服务相关角色，然后需要再次创建，可以使用相同流程在账户中重新创建此角色。再次调用 CreateVerifiedAccessEndpoint 时，Verified Access 将再次创建服务相关角色。

### 编辑 Verified Access 的服务相关角色

Verified Access 不允许编辑 AWSServiceRoleForVPCVerifiedAccess 服务相关角色。创建服务相关角色后，将无法更改角色名称，因为可能有多个实体引用该角色。但是可以使用 IAM 编辑角色说明。有关更多信息，请参阅 IAM 用户指南中的编辑服务相关角色。

### 删除 Verified Access 的服务相关角色

您不需要手动删除 AWSServiceRoleForVPCVerifiedAccess 角色。在 AWS Management Console、AWS CLI 或 AWS API 中调用 DeleteVerifiedAccessEndpoint 时，Verified Access 将清除资源并删除服务相关角色。

使用 IAM 手动删除服务相关角色

使用 IAM 控制台、AWS CLI 或 AWS API 删除 AWSServiceRoleForVPCVerifiedAccess 服务相关角色。有关更多信息，请参阅 IAM 用户指南中的删除服务相关角色。

### Verified Access 服务相关角色的受支持区域

Verified Access 支持在该服务可用的所有 AWS 区域 中使用服务相关角色。有关更多信息，请参阅 AWS 区域和端点。

## AWS Verified Access 的 AWS 托管策略

AWS 托管策略是由 AWS 创建和管理的独立策略。AWS 托管策略旨在为许多常见用例提供权限，以便您可以开始为用户、组和角色分配权限。

请记住，AWS 托管式策略可能不会为您的特定使用场景授予最低权限，因为它们可供所有 AWS 客户使用。我们建议通过定义特定于使用场景的客户管理型策略来进一步减少权限。

您无法更改 AWS 托管式策略中定义的权限。如果 AWS 更新在 AWS 托管式策略中定义的权限，则更新会影响该策略所附加到的所有主体身份（用户、组和角色）。当新的 AWS 服务 启动或新的 API 操作可用于现有服务时，AWS 最有可能更新 AWS 托管式策略。

有关更多信息，请参阅《IAM 用户指南》中的 AWS 托管式策略。

## AWS 托管策略：AWSVPCVerifiedAccessServiceRolePolicy

此策略附加到服务相关角色，允许 Verified Access 代表您执行操作。有关更多信息，请参阅使用服务相关角色。要查看此策略的权限，可以在 AWS Management Console 中查看 AWSVPCVerifiedAccessServiceRolePolicy，也可以在 AWS 托管策略参考指南中查看 AWSVPCVerifiedAccessServiceRolePolicy 策略。

## AWS 托管策略的 Verified Access 更新

查看有关 Verified Access 的 AWS 托管策略更新的详细信息（从该服务开始跟踪这些更改开始）。有关此页面更改的自动提示，请订阅 Verified Access 文档历史记录页面上的 RSS 源。

| 更改 | 说明 | 日期 |
| --- | --- | --- |
| AWSVPCVerifiedAccessServiceRolePolicy - 策略更新 | Verified Access 更新了其托管策略以包含“sid”字段下所有操作的描述。 | 2023 年 11 月 17 日 |
| AWSVPCVerifiedAccessServiceRolePolicy - 策略更新 | Verified Access 更新了其托管策略，以将安全组资源添加到 `ec2:CreateNetworkInterface` 权限中。 | 2023 年 5 月 31 日 |
| AWSVPCVerifiedAccessServiceRolePolicy - 新策略 | Verified Access 添加了一条新策略，允许其在您的账户中配置使用该服务所需的资源。 | 2022 年 11 月 29 日 |
| Verified Access 开始跟踪更改 | Verified Access 开始跟踪其 AWS 托管策略的更改。 | 2022 年 11 月 29 日 |

# 验证访问权限的合规性 AWS 验证

AWS Verified Access 可以配置为支持联邦信息处理标准 (FIPS) 合规性。有关为 Verified Access 设置 FIPS 合规性的更多信息和详情，请转到 Verified Access 的 FIPS 合规性。

要了解是否属于特定合规计划的范围，请参阅AWS 服务 “按合规计划划分的范围” ”，然后选择您感兴趣的合规计划。 AWS 服务 有关一般信息，请参阅AWS 合规计划AWS。

您可以使用下载第三方审计报告 AWS Artifact。有关更多信息，请参阅中的 “下载报告” 中的 “ AWS Artifact。

您在使用 AWS 服务 时的合规责任取决于您的数据的敏感性、贵公司的合规目标以及适用的法律和法规。 AWS 提供了以下资源来帮助实现合规性：

- 安全与合规性快速入门指南 — 这些部署指南讨论了架构注意事项，并提供了在这些基础上 AWS 部署以安全性和合规性为重点的基准环境的步骤。
- 在 A@@ mazon Web Services 上构建 HIPAA 安全与合规架构 — 本白皮书描述了各公司如何使用 AWS 来创建符合 HIPAA 资格的应用程序。

> **Note**
>
> 并非所有 AWS 服务 人都符合 HIPAA 资格。有关更多信息，请参阅符合 HIPAA 要求的服务参考。

- AWS 合规资源AWS — 此工作簿和指南集可能适用于您所在的行业和所在地区。
- AWS 客户合规指南 — 从合规角度了解责任共担模式。这些指南总结了保护的最佳实践， AWS 服务并将指南映射到跨多个框架（包括美国国家标准与技术研究院 (NIST)、支付卡行业安全标准委员会 (PCI) 和国际标准化组织 (ISO)）的安全控制。
- 使用AWS Config 开发人员指南中的规则评估资源 — 该 AWS Config 服务评估您的资源配置在多大程度上符合内部实践、行业准则和法规。
- AWS Security Hub— 这 AWS 服务 提供了您内部安全状态的全面视图 AWS。Security Hub 通过安全控件评估您的 AWS 资源并检查其是否符合安全行业标准和最佳实践。有关受支持服务及控件的列表，请参阅 Security Hub 控件参考。
- Amazon GuardDuty — 它通过监控您的 AWS 账户环境中是否存在可疑和恶意活动，来 AWS 服务检测您的工作负载、容器和数据面临的潜在威胁。 GuardDuty 通过满足某些合规性框架规定的入侵检测要求，可以帮助您满足各种合规性要求，例如 PCI DSS。

- AWS Audit Manager— 这 AWS 服务 可以帮助您持续审计 AWS 使用情况，从而简化风险管理以及对法规和行业标准的合规性。

# AWS Verified Access 的故障恢复能力

AWS 全球基础设施围绕 AWS 区域 和可用区构建。AWS 区域 提供多个在物理上独立且隔离的可用区，这些可用区与延迟率低、吞吐量高且冗余性高的网络连接在一起。利用可用区，您可以设计和操作在可用区之间无中断地自动实现故障转移的应用程序和数据库。与传统的单个或多个数据中心基础设施相比，可用区具有更高的可用性、容错性和可扩展性。

有关 AWS 区域 和可用区的更多信息，请参阅 AWS 全球基础设施。

除了AWS 全球基础设施外，Verified Access 还提供以下功能，以帮助满足您的高可用性需求。

## 多个子网以实现高可用性

当您创建负载均衡器类型的 Verified Access 端点时，可以将多个子网关联到该端点。与端点关联的每个子网必须属于不同的可用区。通过关联多个子网，您可以使用多个可用区来确保高可用性。

# 监控 AWS Verified Access

监控是保持 AWS Verified Access 的可靠性、可用性和性能的重要方面。AWS 提供了以下监控工具来监控 Verified Access、在出现错误时进行报告并适时自动采取措施。

- 访问日志 – 捕获有关应用程序访问请求的详细信息。有关更多信息，请参阅the section called “Verified Access 日志”。
- AWS CloudTrail – 捕获由您的 AWS 账户 或代表该账户发出的 API 调用和相关事件，并将日志文件传输到您指定的 Amazon S3 存储桶。您可以标识哪些用户和账户调用了 AWS、从中发出调用的源 IP 地址以及调用的发生时间。有关更多信息，请参阅the section called “CloudTrail 日志”。

## Verified Access 日志

Veri AWS fied Access 评估每个访问请求后，它会记录所有访问尝试。这提供了对应用程序访问的集中可见性，并帮助您快速响应安全事件和审核请求。Verified Access 支持开放式网络安全架构框架 (OCSF) 日志记录格式。

启用日志记录后，您需要为要发送的日志配置目的地。用于配置日志记录目的地的 IAM 主体需要具有一定的权限才能使日记记录正常工作。可以在 日志记录权限 部分中查看每个日志记录目的地的必需 IAM 权限。Verified Access 支持将访问日志发布到以下目的地：

- Amazon Log CloudWatch s 日志组
- Amazon S3 存储桶
- Amazon Data Firehose 传送流

内容

# 日志记录版本

默认情况下，Verified Access 日志记录系统使用开放式网络安全架构框架 (OCSF) 版本 0.1。可以在 OCSF 版本 0.1 示例 部分中看到使用版本 0.1 的示例日志。

最新的日志记录版本与 OCSF 版本 1.0.0-rc.2 兼容。有关架构的具体细节可以在 OCSF 架构中找到。可以在 OCSF 版本 1.0.0-rc.2 示例 部分中看到使用版本 1.0.0-rc.2 的示例日志。

## 升级日志记录版本

如果要升级正在使用的日志记录版本，请按照以下步骤操作。

使用控制台升级日志记录版本

1. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择适当的 Verified Access 实例。
4. 在 Verified Access 实例日志记录配置选项卡上，选择修改 Verified Access 实例日志记录配置。
5. 从更新日志版本下拉列表中选择 ocsf-1.0.0-rc.2。
6. 选择修改 Verified Access 实例日志记录配置。

要升级日志版本，请使用 AWS CLI

使用-loggin modify-verified-access-instanceg- configuration 命令。

# 日志记录权限

用于配置日志记录目的地的 IAM 主体需要具有一定的权限才能使日记记录正常工作。下面是每个日志记录目的地所需的权限。

要发送到 CloudWatch 日志，请执行以下操作：

- 对 Verified Access 实例的 `ec2:ModifyVerifiedAccessInstanceLoggingConfiguration`
- 对所有资源的 `logs:CreateLogDelivery`、`logs:DeleteLogDelivery`、`logs:GetLogDelivery`、`logs:ListLo` 和 `logs:UpdateLogDelivery`
- 对目的地日志组的 `logs:DescribeLogGroups`、`logs:DescribeResourcePolicies` 和 `logs:PutResourcePolicy`

对于传输到 Amazon S3：

- 对 Verified Access 实例的 `ec2:ModifyVerifiedAccessInstanceLoggingConfiguration`
- 对所有资源的 `logs:CreateLogDelivery`、`logs:DeleteLogDelivery`、`logs:GetLogDelivery`、`logs:ListLo` 和 `logs:UpdateLogDelivery`
- 对目的地存储桶的 `s3:GetBucketPolicy` 和 `s3:PutBucketPolicy`

要配送到 Firehose，请执行以下操作：

- 对 Verified Access 实例的 `ec2:ModifyVerifiedAccessInstanceLoggingConfiguration`
- 对所有资源的 `firehose:TagDeliveryStream`
- 对所有资源的 `iam:CreateServiceLinkedRole`
- 对所有资源的 `logs:CreateLogDelivery`、`logs:DeleteLogDelivery`、`logs:GetLogDelivery`、`logs:ListLo` 和 `logs:UpdateLogDelivery`

## 启用或禁用日志

启用日志记录后，您需要为要发送的日志配置目的地。用于配置日志记录目的地的 IAM 主体需要具有一定的权限才能使日记记录正常工作。可以在 日志记录权限 部分中查看每个日志记录目的地的必需 IAM 权限。

内容



### 启用访问日志

启用 Verified Access 日志

1. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择 Verified Access 实例。
4. 在 Verified Access 实例日志记录配置选项卡上，选择修改 Verified Access 实例日志记录配置。

5. （可选）要在日志中包含从信任提供商发送的信任数据，请执行以下操作：
   a. 从更新日志版本下拉列表中选择 ocsf-1.0.0-rc.2。
   b. 选择包括信任上下文。
6. 请执行以下操作之一：
   - 打开 “传送到 Amazon CloudWatch 日志”。选择目的地日志组。
   - 开启传输到 Amazon S3。输入目的地存储桶的名称、所有者和前缀。
   - 开启 Deliver y to Firehose。创建目的地传输流。
7. 选择修改 Verified Access 实例日志记录配置。

要启用已验证访问日志，请使用 AWS CLI

使用-loggin modify-verified-access-instanceg- configuration 命令。

### 禁用访问日志

您可以随时禁用 Verified Access 实例的访问日志。禁用访问日志后，日志数据将保留在日志目的地，直到您将其删除。

禁用 Verified Access 日志

1. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择 Verified Access 实例。
4. 在 Verified Access 实例日志记录配置选项卡上，选择修改 Verified Access 实例日志记录配置。
5. 关闭日志传输。
6. 选择修改 Verified Access 实例日志记录配置。

要禁用已验证的访问日志，请使用 AWS CLI

使用-loggin modify-verified-access-instanceg- configuration 命令。

## 包括信任上下文

信任提供商发送的信任上下文可以选择性地包括在 Verified Access 日志中。在定义允许或拒绝访问应用程序的策略时，这可能非常有用。启用后，可在日志的 `data` 字段下找到信任上下文。如果禁

用，data 字段将设置为 null。要将 Verified Access 配置为在日志中包含信任上下文，请按照以下步骤操作。

> **Note**
>
> 在 Verified Access 日志中包含信任上下文需要升级到最新的日志记录版本 ocsf-1.0.0-rc.2。以下步骤假定您已启用日志记录。如果不是这样，请参阅 启用访问日志 了解完整过程。

内容



## 启用信任上下文

使用控制台在 Verified Access 日志中包含信任上下文

1. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。
2. 在导航窗格中，选择 Verified Access 实例。
3. 选择适当的 Verified Access 实例。
4. 在 Verified Access 实例日志记录配置选项卡上，选择修改 Verified Access 实例日志记录配置。
5. 从更新日志版本下拉列表中选择 ocsf-1.0.0-rc.2。
6. 开启包括信任上下文。
7. 选择修改 Verified Access 实例日志记录配置。

要在已验证的访问权限日志中包含信任上下文，请使用 AWS CLI

使用-loggin modify-verified-access-instanceg- configuration 命令。

## 禁用信任上下文

如果您不想再在日志中包含信任上下文，可以通过以下步骤将其删除。

使用控制台从 Verified Access 日志中删除信任上下文

1. 通过 https://console.aws.amazon.com/vpc/ 打开 Amazon VPC 控制台。

2. 在导航窗格中，选择 Verified Access 实例。
3. 选择适当的 Verified Access 实例。
4. 在 Verified Access 实例日志记录配置选项卡上，选择修改 Verified Access 实例日志记录配置。
5. 关闭包括信任上下文。
6. 选择修改 Verified Access 实例日志记录配置。

要从 “已验证访问权限” 日志中删除信任上下文 AWS CLI

使用-loggin modify-verified-access-instanceg- configuration 命令。

## Verified Access 日志的示例日志条目

以下是日志条目示例。

内容

- OCSF 版本 0.1 示例
- OCSF 版本 1.0.0-rc.2 示例

### OCSF 版本 0.1 示例

以下是使用默认日志记录 OCSF 版本 0.1 的日志示例。

示例

- 通过 OIDC 授予访问权限
- 通过 OIDC 和 JAMF 授予访问权限
- 通过 OIDC 授予访问权限以及 CrowdStrike
- 由于缺少 Cookie，访问被拒绝
- 访问被策略拒绝
- 未知日志条目

通过 OIDC 授予访问权限

在此示例日志条目中，Verified Access 允许通过 OIDC 用户信任提供商访问端点。

```
{
    "activity": "Access Granted",
```

```
    "activity_id": "1",
    "category_name": "Application Activity",
    "category_uid": "8",
    "class_name": "Access Logs",
    "class_uid": "208001",
    "device": {
        "ip": "10.2.7.68",
        "type": "Unknown",
        "type_id": 0
    },
    "duration": "0.004",
    "end_time": "1668580194344",
    "time": "1668580194344",
    "http_request": {
        "http_method": "GET",
        "url": {
            "hostname": "hello.app.example.com",
            "path": "/",
            "port": 443,
            "scheme": "https",
            "text": "https://hello.app.example.com:443/"
        },
        "user_agent": "python-requests/2.28.1",
        "version": "HTTP/1.1"
    },
    "http_response": {
        "code": 200
    },
    "identity": {
        "authorizations": [
            {
                "decision": "Allow",
                "policy": {
                    "name": "inline"
                }
            }
        ],
        "idp": {
            "name": "user",
            "uid": "vatp-09bc4cbce2EXAMPLE"
        },
        "user": {
            "email_addr": "johndoe@example.com",
            "name": "Test User Display",
```

```
            "uid": "johndoe@example.com",
            "uuid": "00u6wj48lbxTAEXAMPLE"
        }
    },
    "message": "",
    "metadata": {
        "uid": "Root=1-63748362-6408d24241120b942EXAMPLE",
        "logged_time": 1668580281337,
        "version": "0.1",
        "product": {
            "name": "Verified Access",
            "vendor_name": "AWS"
        }
    },
    "ref_time": "2022-11-16T06:29:54.344948Z",
    "proxy": {
        "ip": "192.168.34.167",
        "port": 443,
        "svc_name": "Verified Access",
        "uid": "vai-002fa341aeEXAMPLE"
    },
    "severity": "Informational",
    "severity_id": "1",
    "src_endpoint": {
        "ip": "172.24.57.68",
        "port": "48234"
    },
    "start_time": "1668580194340",
    "status_code": "100",
    "status_details": "Access Granted",
    "status_id": "1",
    "status": "Success",
    "type_uid": "20800101",
    "type_name": "AccessLogs: Access Granted",
    "unmapped": null
}
```


通过 OIDC 和 JAMF 授予访问权限

在此示例日志条目中，Verified Access 允许通过 OIDC 和 JAMF 设备信任提供商访问端点。


```
{
    "activity": "Access Granted",
    "activity_id": "1",
```

```
    "category_name": "Application Activity",
    "category_uid": "8",
    "class_name": "Access Logs",
    "class_uid": "208001",
    "device": {
        "ip": "10.2.7.68",
        "type": "Unknown",
        "type_id": 0,
        "uid": "41b07859-4222-4f41-f3b9-97dc1EXAMPLE"
    },
    "duration": "0.347",
    "end_time": "1668804944086",
    "time": "1668804944086",
    "http_request": {
        "http_method": "GET",
        "url": {
            "hostname": "hello.app.example.com",
            "path": "/",
            "port": 443,
            "scheme": "h2",
            "text": "https://hello.app.example.com:443/"
        },
        "user_agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36",
        "version": "HTTP/2.0"
    },
    "http_response": {
        "code": 304
    },
    "identity": {
        "authorizations": [
            {
                "decision": "Allow",
                "policy": {
                    "name": "inline"
                }
            }
        ],
        "idp": {
            "name": "oidc",
            "uid": "vatp-9778003bc2EXAMPLE"
        },
        "user": {
            "email_addr": "johndoe@example.com",
```

```
            "name": "Test User Display",
            "uid": "johndoe@example.com",
            "uuid": "4f040d0f96becEXAMPLE"
        }
    },
    "message": "",
    "metadata": {
        "uid": "Root=1-321318ce-6100d340adf4fb29dEXAMPLE",
        "logged_time": 1668805278555,
        "version": "0.1",
        "product": {
            "name": "Verified Access",
            "vendor_name": "AWS"
        }
    },
    "ref_time": "2022-11-18T20:55:44.086480Z",
    "proxy": {
        "ip": "10.5.192.96",
        "port": 443,
        "svc_name": "Verified Access",
        "uid": "vai-3598f66575EXAMPLE"
    },
    "severity": "Informational",
    "severity_id": "1",
    "src_endpoint": {
        "ip": "192.168.20.246",
        "port": 61769
    },
    "start_time": "1668804943739",
    "status_code": "100",
    "status_details": "Access Granted",
    "status_id": "1",
    "status": "Success",
    "type_uid": "20800101",
    "type_name": "AccessLogs: Access Granted",
    "unmapped": null
}
```


### 通过 OIDC 授予访问权限以及 CrowdStrike

在此示例日志条目中，Verified Access 允许通过 OIDC 和 CrowdStrike设备信任提供商访问端点。


```
{
    "activity": "Access Granted",
```

```
    "activity_id": "1",
    "category_name": "Application Activity",
    "category_uid": "8",
    "class_name": "Access Logs",
    "class_uid": "208001",
    "device": {
        "ip": "10.2.173.3",
        "os": {
            "name": "Windows 11",
            "type": "Windows",
            "type_id": 100
        },
        "type": "Unknown",
        "type_id": 0,
        "uid": "122978434f65093aee5dfbdc0EXAMPLE",
        "hw_info": {
            "serial_number": "751432a1-d504-fd5e-010d-5ed11EXAMPLE"
        }
    },
    "duration": "0.028",
    "end_time": "1668816620842",
    "time": "1668816620842",
    "http_request": {
        "http_method": "GET",
        "url": {
            "hostname": "test.app.example.com",
            "path": "/",
            "port": 443,
            "scheme": "h2",
            "text": "https://test.app.example.com:443/"
        },
        "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36",
        "version": "HTTP/2.0"
    },
    "http_response": {
        "code": 304
    },
    "identity": {
        "authorizations": [
            {
                "decision": "Allow",
                "policy": {
                    "name": "inline"
```

```
                }
            }
        ],
        "idp": {
            "name": "oidc",
            "uid": "vatp-506d9753f6EXAMPLE"
        },
        "user": {
            "email_addr": "johndoe@example.com",
            "name": "Test User Display",
            "uid": "johndoe@example.com",
            "uuid": "23bb45b16a389EXAMPLE"
        }
    },
    "message": "",
    "metadata": {
        "uid": "Root=1-c16c5a65-b641e4056cc6cb0eeEXAMPLE",
        "logged_time": 1668816977134,
        "version": "0.1",
        "product": {
            "name": "Verified Access",
            "vendor_name": "AWS"
        }
    },
    "ref_time": "2022-11-19T00:10:20.842295Z",
    "proxy": {
        "ip": "192.168.144.62",
        "port": 443,
        "svc_name": "Verified Access",
        "uid": "vai-2f80f37e64EXAMPLE"
    },
    "severity": "Informational",
    "severity_id": "1",
    "src_endpoint": {
        "ip": "10.14.173.3",
        "port": 55706
    },
    "start_time": "1668816620814",
    "status_code": "100",
    "status_details": "Access Granted",
    "status_id": "1",
    "status": "Success",
    "type_uid": "20800101",
    "type_name": "AccessLogs: Access Granted",
```

```
    "unmapped": null
}
```

由于缺少 Cookie，访问被拒绝

在此示例日志条目中，由于缺少身份验证 Cookie，Verified Access 拒绝访问。

```
{
    "activity": "Access Denied",
    "activity_id": "2",
    "category_name": "Application Activity",
    "category_uid": "8",
    "class_name": "Access Logs",
    "class_uid": "208001",
    "device": null,
    "duration": "0.0",
    "end_time": "1668593568259",
    "time": "1668593568259",
    "http_request": {
        "http_method": "POST",
        "url": {
            "hostname": "hello.app.example.com",
            "path": "/dns-query",
            "port": 443,
            "scheme": "h2",
            "text": "https://hello.app.example.com:443/dns-query"
        },
        "user_agent": "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML",
        "version": "HTTP/2.0"
    },
    "http_response": {
        "code": 302
    },
    "identity": null,
    "message": "",
    "metadata": {
        "uid": "Root=1-5cf1c832-a565309ce20cc7dafEXAMPLE",
        "logged_time": 1668593776720,
        "version": "0.1",
        "product": {
            "name": "Verified Access",
            "vendor_name": "AWS"
        }
    },
```

```
    "ref_time": "2022-11-16T10:12:48.259762Z",
    "proxy": {
        "ip": "192.168.34.167",
        "port": 443,
        "svc_name": "Verified Access",
        "uid": "vai-108ed7a672EXAMPLE"
    },
    "severity": "Informational",
    "severity_id": "1",
    "src_endpoint": {
        "ip": "10.7.178.16",
        "port": "46246"
    },
    "start_time": "1668593568258",
    "status_code": "200",
    "status_details": "Authentication Denied",
    "status_id": "2",
    "status": "Failure",
    "type_uid": "20800102",
    "type_name": "AccessLogs: Access Denied",
    "unmapped": null
}
```


访问被策略拒绝

在此示例日志条目中，Verified Access 拒绝了一个经过身份验证的请求，因为访问策略不允许该请求。


```
{
    "activity": "Access Denied",
    "activity_id": "2",
    "category_name": "Application Activity",
    "category_uid": "8",
    "class_name": "Access Logs",
    "class_uid": "208001",
    "device": {
        "ip": "10.4.133.137",
        "type": "Unknown",
        "type_id": 0
    },
    "duration": "0.023",
    "end_time": "1668573630978",
    "time": "1668573630978",
```

```
    "http_request": {
        "http_method": "GET",
        "url": {
            "hostname": "hello.app.example.com",
            "path": "/",
            "port": 443,
            "scheme": "h2",
            "text": "https://hello.app.example.com:443/"
        },
        "user_agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36",
        "version": "HTTP/2.0"
    },
    "http_response": {
        "code": 401
    },
    "identity": {
        "authorizations": [],
        "idp": {
            "name": "user",
            "uid": "vatp-e048b3e0f8EXAMPLE"
        },
        "user": {
            "email_addr": "johndoe@example.com",
            "name": "Test User Display",
            "uid": "johndoe@example.com",
            "uuid": "0e1281ad3580aEXAMPLE"
        }
    },
    "message": "",
    "metadata": {
        "uid": "Root=1-531a036a-09e95794c7b96aefbEXAMPLE",
        "logged_time": 1668573773753,
        "version": "0.1",
        "product": {
            "name": "Verified Access",
            "vendor_name": "AWS"
        }
    },
    "ref_time": "2022-11-16T04:40:30.978732Z",
    "proxy": {
        "ip": "3.223.34.167",
        "port": 443,
        "svc_name": "Verified Access",
```

```
        "uid": "vai-021d5eaed2EXAMPLE"
    },
    "severity": "Informational",
    "severity_id": "1",
    "src_endpoint": {
        "ip": "10.4.133.137",
        "port": "31746"
    },
    "start_time": "1668573630955",
    "status_code": "300",
    "status_details": "Authorization Denied",
    "status_id": "2",
    "status": "Failure",
    "type_uid": "20800102",
    "type_name": "AccessLogs: Access Denied",
    "unmapped": null
}
```

未知日志条目

在此示例日志条目中，Verified Access 无法生成完整的日志条目，因此它会发出未知的日志条目。这可以确保每个请求都出现在访问日志中。

```
{
    "activity": "Unknown",
    "activity_id": "0",
    "category_name": "Application Activity",
    "category_uid": "8",
    "class_name": "Access Logs",
    "class_uid": "208001",
    "device": null,
    "duration": "0.004",
    "end_time": "1668580207898",
    "time": "1668580207898",
    "http_request": {
        "http_method": "GET",
        "url": {
            "hostname": "hello.app.example.com",
            "path": "/",
            "port": 443,
            "scheme": "https",
            "text": "https://hello.app.example.com:443/"
        },
```

```
        "user_agent": "python-requests/2.28.1",
        "version": "HTTP/1.1"
    },
    "http_response": {
        "code": 200
    },
    "identity": null,
    "message": "",
    "metadata": {
        "uid": "Root=1-435eb955-6b5a1d529343f5adaEXAMPLE",
        "logged_time": 1668580579147,
        "version": "0.1",
        "product": {
            "name": "Verified Access",
            "vendor_name": "AWS"
        }
    },
    "ref_time": "2022-11-16T06:30:07.898344Z",
    "proxy": {
        "ip": "10.1.34.167",
        "port": 443,
        "svc_name": "Verified Access",
        "uid": "vai-6c32b53b3cEXAMPLE"
    },
    "severity": "Informational",
    "severity_id": "1",
    "src_endpoint": {
        "ip": "172.28.57.68",
        "port": "47220"
    },
    "start_time": "1668580207893",
    "status_code": "000",
    "status_details": "Unknown",
    "status_id": "0",
    "status": "Unknown",
    "type_uid": "20800100",
    "type_name": "AccessLogs: Unknown",
    "unmapped": null
}
```


## OCSF 版本 1.0.0-rc.2 示例

### 内容

- 在包含信任上下文的情况下授予访问权限
- 在忽略信任上下文的情况下授予访问权限

### 在包含信任上下文的情况下授予访问权限


```
{
    "activity_name": "Access Grant",
    "activity_id": "1",
    "actor": {
        "authorizations": [{
            "decision": "Allow",
            "policy": {
                "name": "inline"
            }
        }],
        "idp": {
            "name": "user",
            "uid": "vatp-09bc4cbce2EXAMPLE"
        },
        "invoked_by": "",
        "process": {},
        "user": {
            "email_addr": "johndoe@example.com",
            "name": "Test User Display",
            "uid": "johndoe@example.com",
            "uuid": "00u6wj48lbxTAEXAMPLE"
        },
        "session": {}
    },
    "category_name": "Audit Activity",
    "category_uid": "3",
    "class_name": "Access Activity",
    "class_uid": "3006",
    "device": {
        "ip": "10.2.7.68",
        "type": "Unknown",
        "type_id": 0
    },
    "duration": "0.004",
    "end_time": "1668580194344",
    "time": "1668580194344",
    "http_request": {
```

```
        "http_method": "GET",
        "url": {
            "hostname": "hello.app.example.com",
            "path": "/",
            "port": 443,
            "scheme": "https",
            "text": "https://hello.app.example.com:443/"
        },
        "user_agent": "python-requests/2.28.1",
        "version": "HTTP/1.1"
    },
    "http_response": {
        "code": 200
    },
    "message": "",
    "metadata": {
        "uid": "Root=1-63748362-6408d24241120b942EXAMPLE",
        "logged_time": 1668580281337,
        "version": "1.0.0-rc.2",
        "product": {
            "name": "Verified Access",
            "vendor_name": "AWS"
        }
    },
    "ref_time": "2022-11-16T06:29:54.344948Z",
    "proxy": {
        "ip": "192.168.34.167",
        "port": 443,
        "svc_name": "Verified Access",
        "uid": "vai-002fa341aeEXAMPLE"
    },
    "severity": "Informational",
    "severity_id": "1",
    "src_endpoint": {
        "ip": "172.24.57.68",
        "port": "48234"
    },
    "start_time": "1668580194340",
    "status_code": "100",
    "status_detail": "Access Granted",
    "status_id": "1",
    "status": "Success",
    "type_uid": "300601",
    "type_name": "Access Activity: Access Grant",
```

```
    "data": {
        "context": {
            "oidc": {
                "family_name": "Last",
                "zoneinfo": "America/Los_Angeles",
                "exp": 1670631145,
                "middle_name": "Middle",
                "given_name": "First",
                "email_verified": true,
                "name": "Test User Display",
                "updated_at": 1666305953,
                "preferred_username": "johndoe-user@test.com",
                "profile": "http://www.example.com",
                "locale": "US",
                "nickname": "Tester",
                "email": "johndoe-user@test.com"
            },
            "http_request": {
                "x_forwarded_for": "1.1.1.1,2.2.2.2",
                "http_method": "GET",
                "user_agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36",
                "port": "80",
                "hostname": "hostname.net"
            }
        }
    }
}
```

### 在忽略信任上下文的情况下授予访问权限

```
{
    "activity_name": "Access Grant",
    "activity_id": "1",
    "actor": {
        "authorizations": [{
            "decision": "Allow",
            "policy": {
                "name": "inline"
            }
        }],
        "idp": {
            "name": "user",
```

```
            "uid": "vatp-09bc4cbce2EXAMPLE"
        },
        "invoked_by": "",
        "process": {},
        "user": {
            "email_addr": "johndoe@example.com",
            "name": "Test User Display",
            "uid": "johndoe@example.com",
            "uuid": "00u6wj48lbxTAEXAMPLE"
        },
        "session": {}
    },
    "category_name": "Audit Activity",
    "category_uid": "3",
    "class_name": "Access Activity",
    "class_uid": "3006",
    "device": {
        "ip": "10.2.7.68",
        "type": "Unknown",
        "type_id": 0
    },
    "duration": "0.004",
    "end_time": "1668580194344",
    "time": "1668580194344",
    "http_request": {
        "http_method": "GET",
        "url": {
            "hostname": "hello.app.example.com",
            "path": "/",
            "port": 443,
            "scheme": "https",
            "text": "https://hello.app.example.com:443/"
        },
        "user_agent": "python-requests/2.28.1",
        "version": "HTTP/1.1"
    },
    "http_response": {
        "code": 200
    },
    "message": "",
    "metadata": {
        "uid": "Root=1-63748362-6408d24241120b942EXAMPLE",
        "logged_time": 1668580281337,
        "version": "1.0.0-rc.2",
```

```
        "product": {
            "name": "Verified Access",
            "vendor_name": "AWS"
        }
    },
    "ref_time": "2022-11-16T06:29:54.344948Z",
    "proxy": {
        "ip": "192.168.34.167",
        "port": 443,
        "svc_name": "Verified Access",
        "uid": "vai-002fa341aeEXAMPLE"
    },
    "severity": "Informational",
    "severity_id": "1",
    "src_endpoint": {
        "ip": "172.24.57.68",
        "port": "48234"
    },
    "start_time": "1668580194340",
    "status_code": "100",
    "status_detail": "Access Granted",
    "status_id": "1",
    "status": "Success",
    "type_uid": "300601",
    "type_name": "Access Activity: Access Grant",
    "data": null
}
```


# 使用 AWS CloudTrail 记录 AWS Verified Access API 调用

AWS Verified Access 与 AWS CloudTrail 集成，该服务提供 Verified Access 中的用户、角色或 AWS 服务 采取的操作记录。CloudTrail 将 Verified Access 的所有 API 调用作为事件捕获。捕获的调用包含来自 Verified Access 控制台的调用和对 Verified Access API 操作的代码调用。如果您创建跟踪，则可以使 CloudTrail 事件持续传送到 Amazon S3 存储桶（包括 Verified Access 的事件）。如果您不配置跟踪，则仍可在 CloudTrail 控制台中的 Event history（事件历史记录）中查看最新事件。使用 CloudTrail 收集的信息，您可以确定向 Verified Access 发出了什么请求、发出请求的 IP 地址、请求方、请求时间以及其他详细信息。

要了解有关 CloudTrail 的更多信息，请参阅《AWS CloudTrail 用户指南》。

## CloudTrail 中的 Verified Access 信息

在您创建 AWS 账户 时，将在该账户上启用 CloudTrail。当 Verified Access 中发生活动时，该活动将记录在 CloudTrail 事件中，并与其他 AWS 服务 事件一同保存在 Event history（事件历史记录）中。您可以在 AWS 账户中查看、搜索和下载最新事件。有关更多信息，请参阅使用 CloudTrail 事件历史记录查看事件。

要持续记录您的 AWS 账户中的事件（包括 Verified Access 的事件），请创建一个 trail（跟踪）。通过跟踪记录，CloudTrail 可将日志文件传送至 Simple Storage Service（Amazon S3）存储桶。预设情况下，在控制台中创建跟踪记录时，此跟踪记录应用于所有AWS 区域。此跟踪记录在 AWS 分区中记录所有区域中的事件，并将日志文件传送至您指定的 Simple Storage Service（Amazon S3）桶。此外，您可以配置其他 AWS 服务，进一步分析在 CloudTrail 日志中收集的事件数据并采取行动。有关更多信息，请参阅下列内容：

- 创建跟踪概览
- CloudTrail 支持的服务和集成
- 为 CloudTrail 配置 Amazon SNS 通知
- 从多个区域接收 CloudTrail 日志文件和从多个账户接收 CloudTrail 日志文件

CloudTrail 会记录所有 Verified Access 操作，Amazon EC2 API 参考中说明了这些操作。例如，对 `CreateVerifiedAccessInstance`、`DeleteVerifiedAccessInstance` 和 `ModifyVerifiedAccessInstance` 操作的调用会在 CloudTrail 日志文件中生成条目。

每个事件或日志条目都包含有关生成请求的人员信息。身份信息可帮助您确定以下内容：

- 请求是使用根用户凭证还是 AWS Identity and Access Management (IAM) 用户凭证发出的。
- 请求是使用角色还是联合身份用户的临时安全凭证发出的。
- 请求是否由其他 AWS 服务 发出。

有关更多信息，请参阅 CloudTrail userIdentity 元素。

## 了解 Verified Access 日志文件条目

跟踪是一种配置，可用于将事件作为日志文件传送到您指定的 Amazon S3 桶。CloudTrail 日志文件包含一个或多个日志条目。事件表示来自任何源的单个请求。它包括有关所请求操作的信息、操作的日期和时间、请求参数等。CloudTrail 日志文件不是公用 API 调用的有序堆栈跟踪，因此它们不会按任何特定顺序显示。

以下示例显示了用于 CreateVerifiedAccessInstance 操作的 CloudTrail 日志条目。


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIKK4OOINJWEXAMPLE:jdoe",
        "arn": "arn:aws:iam::123456789012:user/jdoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "jdoe"
    },
    "eventTime": "2022-11-18T20:44:04Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "CreateVerifiedAccessInstance",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "198.51.100.1",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "CreateVerifiedAccessInstanceRequest": {
            "Description": "",
            "ClientToken": "85893b1e-49f6-4d24-97de-280c664edf1b"
        }
    },
    "responseElements": {
        "CreateVerifiedAccessInstanceResponse": {
            "verifiedAccessInstance": {
                "creationTime": "2022-11-18T20:44:04",
                "description": "",
                "verifiedAccessInstanceId": "vai-0d79d91875542c549",
                "verifiedAccessTrustProviderSet": ""
            },
            "requestId": "2eae195d-6bfd-46d7-b46e-a68cb791de09"
        }
    },
    "requestID": "2eae195d-6bfd-46d7-b46e-a68cb791de09",
    "eventID": "297d6529-1144-40f6-abf8-3a76f18d88f0",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

# AWS Verified Access 的配额

对于每项 AWS 服务，您的 AWS 账户 都具有默认配额（以前被称为限制）。除非另有说明，否则，每个配额是区域特定的。

AWS 账户级配额

您的 AWS 账户 具有以下与 Verified Access 相关的配额。

| 名称 | 默认值 | 可调整 | 描述 |
| --- | --- | --- | --- |
| Verified Access 实例 | 5 | 是 | 客户可以在当前区域中创建的 Verified Access 实例的最大数量。 |
| Verified Access 组 | 10 | 是 | 客户可以在当前区域中创建的 Verified Access 组的最大数量。 |
| Verified Access 信任提供商 | 15 | 是 | 客户可以在当前区域中创建的 Verified Access 信任提供商的最大数量。 |
| Verified Access 端点 | 50 | 是 | 客户可以在当前区域中创建的 Verified Access 端点的最大数量。 |

HTTP 标头

HTTP 标头具有以下大小限制：

| 名称 | 默认值 | 可调整 |
| --- | --- | --- |
| 请求行 | 16K | 否 |
| 单个标头 | 16K | 否 |
| 整个响应标头 | 32 K | 否 |
| 整个请求标头 | 64K | 否 |

## OIDC 声明大小

以下是 OIDC 声明大小限制。

| 名称 | 默认值 | 可调整 |
| --- | --- | --- |
| OIDC 声明大小 | 11 K | 否 |

# Verified Access 用户指南的文档历史记录

下表说明了 Verified Access 的文档版本。

| 变更 | 说明 | 日期 |
| --- | --- | --- |
| 更新了 AWS 托管式策略 | 更新了针对 Verified Access 的 AWS 托管 IAM policy。 | 2023 年 11 月 17 日 |
| 静态数据加密 | 默认情况下，AWSVerified Access 使用 AWS 拥有的 KMS 密钥加密静态数据。 | 2023 年 9 月 28 日 |
| 支持 FIPS 合规性 | 配置 Verified Access 以符合 FIPS。 | 2023 年 9 月 26 日 |
| 增强的日志记录 | 增加了日志记录功能，可向日志添加信任上下文。 | 2023 年 6 月 19 日 |
| 更新了 AWS 托管式策略 | 更新了针对 Verified Access 的 AWS 托管 IAM policy。 | 2023 年 5 月 31 日 |
| GA 版本 | Verified Access 用户指南的 GA 版本。包括 AWS WAF 集成。 | 2023 年 4 月 27 日 |
| 预览版 | Verified Access 用户指南的预览版 | 2022 年 11 月 29 日 |

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。