Secure Content Delivery with Amazon CloudFront

AWS Whitepaper

Secure Content Delivery with Amazon CloudFront: AWS Whitepaper

Table of Contents

Secure Content Delivery with Amazon CloudFront

Secure Content Delivery with Amazon CloudFront

Publication date: **January 11, 2022** (*Document revisions* (p. 18))

Abstract

Securing delivery over the public internet is an important part of cloud security. This whitepaper describes how Amazon CloudFront, a highly secure, managed service, can help architects and developers secure the delivery of their applications and content by providing useful, security-supporting features.

Introduction

As more businesses move to cloud computing, public awareness of the significance of cloud security increases as well. Cloud computing uses public internet to deliver the results to users. Securing this delivery is one of the important parts of cloud security.

Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally, with low latency and high transfer speeds. CloudFront is integrated with Amazon Web Services (AWS). The physical points of presence (PoPs) are directly connected with AWS global infrastructure, and the service works seamlessly with AWS services, including Amazon Simple Storage Service (Amazon S3), AWS Shield, Amazon CloudWatch, and Lambda@Edge. Because CloudFront is the component nearest to end users (sometimes called "viewers") in many workloads, and by default its endpoint is open to public internet, CloudFront is one of the first points on which the security of a customer's application relies on.

AWS follows the shared responsibility security model, and because CloudFront is a fully managed service, AWS responsibility includes physical infrastructure, network, servers, operating systems, and software. Securing the data itself is still the customer's responsibility. To strengthen your applications' security posture, it is crucial to understand what kind of security measures are used in CloudFront, and what kind of security features you can utilize.

This document discusses how AWS protects CloudFront infrastructure (security *of* the cloud) and how you can harden your applications' security (security *in* the cloud) by leveraging CloudFront features.

How AWS Provides Security of the Cloud for Amazon CloudFront

AWS security processes

AWS operates the global cloud infrastructure that you use to provision a variety of basic computing resources and services such as compute and storage. The AWS global infrastructure is designed and managed according to security best practices, as well as a variety of security compliance standards. As an AWS customer, be assured that you're building web architectures on some of the most secure computing infrastructure in the world.

As a managed service, CloudFront is protected by these AWS global infrastructure security processes. The implemented controls include physical and environmental security such as fire detection and suppression, logical AWS access controls such as account review and audit, network security (fault-tolerant design), and other important secure design principles and practices.

Security is built into every layer of the AWS infrastructure, and carries into each of the services that run in it. AWS services, including CloudFront, are architected to work efficiently and securely with all AWS networks and platforms. Each service provides extensive security features designed to help you protect sensitive data and applications. For more information, see the Best Practices for Security, Identity, and Compliance.

Compliance validation for CloudFront

Third-party auditors assess the security and compliance of CloudFront as part of multiple AWS compliance programs. You can download the third-party audit reports using AWS Artifact, a central resource for compliance-related information that provides on-demand access to AWS security and compliance reports, and select online agreements. Reports available in AWS Artifact include our Service Organization Control (SOC) reports and Payment Card Industry (PCI) reports, among other certifications.

Your compliance responsibility when using CloudFront is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help you determine and approach your compliance requirements:

- Security and Compliance Quick Start guides — These deployment guides discuss architectural considerations, and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- *Architecting for HIPAA Security and Compliance* whitepaper: — This whitepaper describes how companies can use AWS to create Health Insurance Portability and Accountability Act (HIPAA)-compliant applications. The AWS HIPAA compliance program includes CloudFront as a HIPAA eligible service. If you have an executed Business Associate Addendum (BAA) with AWS, you can use CloudFront to deliver content that contains protected health information (PHI). For more information, see HIPAA Compliance.
- AWS Compliance Resources — This collection of workbooks and guides might apply to your industry and location.
- AWS Config — This AWS Service enables you to evaluate the configuration settings of your AWS resources and assess how well your resource configurations comply with internal practices, industry guidelines, and regulations.

- AWS Security Hub — This AWS Service provides a comprehensive view of your security state within AWS. This helps you check your compliance with security industry standards and best practices.

Securing HTTPS delivery

HTTPS is a compulsory requirement to deliver data securely over the internet. CloudFront has security controls for handling the Transport Layer Security (TLS) certificates and private keys used in the TLS handshake. During the TLS handshake between the viewer and CloudFront, the CloudFront edge server must provide the correct TLS certificate, and use its private key to decrypt a shared secret that will be used in the TLS session. These TLS certificates are issued or uploaded in AWS Certificate Manager (ACM).

CloudFront encrypts the private certificate with a data encryption key provided by AWS Key Management Service (AWS KMS). After KMS encryption, CloudFront encrypts data again with its own internal encryption, and deploys the encrypted certificates to each CloudFront edge server. The encryption keys used for the transfer mechanism are rotated every 24 hours by CloudFront. Decrypted private keys are never physically stored on CloudFront edge servers. The server decrypts the key each time the private key is loaded into memory, and CloudFront records the access request to the private key information for auditing.

CloudFront uses the s2n library for TLS communication, an open-source library which is comparably new and more lightweight than OpenSSL. OpenSSL, initially released in 1998, is a software library that includes TLS implementation. Open SSL has been used by many web servers. OpenSSL has more than 500,000 lines of code, which makes it challenging to do code audits or reviews. The s2n implementation has around 6,000 lines of code which are regularly reviewed and tested, as well as being tested by Amazon's Automated Reasoning Group.

Resilience and availability

Several forms of DDoS mitigation are included automatically with AWS services. All AWS customers benefit from the automatic protections of AWS Shield Standard at no additional charge. AWS Shield Standard defends against most common, frequently occurring network and transport layer (Layer 3 and Layer 4) Distributed Denial-of-Service (DDoS) attacks that target your website or applications. In each AWS Region, DDoS attacks are detected by a system that automatically baselines traffic, identifies anomalies, and, as necessary, implements mitigations. You can use AWS Shield Standard as part of a DDoS-resilient architecture to protect both web and non-web applications. For more information about how to architect for DDoS resiliency, see the *AWS Best Practices for DDoS Resiliency* whitepaper.

Using services like CloudFront, which are part of the Amazon Global Edge Network, can further improve the DDoS resilience of your application when you serve web application traffic from edge locations distributed around the world. The scale of the Amazon Global Edge Network spans hundreds of PoPs, across dozens of cities and countries, adding new PoPs frequently. This scale offers resiliency and global presence, and provides protection against localized availability problems.

The benefits of using CloudFront include:

- AWS Shield DDoS mitigation systems that are directly integrated with AWS edge services, reducing time-to-mitigate from minutes to sub-seconds.
- Stateless SYN flood mitigation techniques that proxy and verify incoming connections before passing them to the protected service.
- Automatic traffic engineering systems that can disperse or isolate the impact of large volumetric DDoS attacks.
- Application layer defense, when combined with AWS Web Application Firewall (AWS WAF), that does not require changing your current application architecture (for example, in an AWS Region or on-premises datacenter).

- CloudFront enables you to cache static content and serve it from AWS edge locations, which can help reduce the load on your origin server. It can also help reduce server load by preventing non-web traffic from reaching your origin server.
- CloudFront can automatically close connections from slow reading or slow writing attackers (for example, a Slowloris attack).
- Protection from HTTP desync attacks, by integration with HTTP Desync Guardian.
- There is no charge for inbound data transfer on AWS.

How CloudFront Can Help You Ensure Security in the Cloud

This section of the whitepaper focuses on how you can harden your application's security posture by using various features that CloudFront provides.

Using HTTPS with CloudFront

When you use Hypertext Transfer Protocol (HTTP) to deliver data, your data is exposed to public internet, because the protocol transmits data in a clear text format. This can be exploited with man-in-the-middle attacks, which attempt to steal or tamper with private data. Hypertext Transfer Protocol Secure (HTTPS) is an extension of HTTP that provides secure and reliable communication over the internet. HTTPS uses TLS, which encrypts the transmitted data and checks the message integrity. TLS also provides a TLS certificate which authenticates a server's identity. Some security standards, including Payment Card Industry Data Security Standard (PCI DSS) or industry standard Apple iOS App Transport Security (ATS), require the use of HTTPS. There are other benefits, such as better search engine optimization, when using HTTPS.

CloudFront comes with native HTTPS support. You can configure CloudFront to deliver contents via HTTPS from viewer to origin, end to end.

Viewer HTTPS configuration

In CloudFront, the HTTPS setting is configurable for two different connections:

- Viewer to CloudFront
- CloudFront to origin

In the context of HTTPS from viewer to CloudFront, CloudFront is a server which receives the TLS-initiating "Client Hello" in the TLS handshake process. Because the HTTPS is initiated by the client, CloudFront has options to:

- Accept both HTTP and HTTPS
- Let the client use HTTPS if they initiate the request via HTTP, or
- Ignore HTTP and respond with an error message

These options can be applied to a specific URI path so you can configure some part of your website to accept HTTP. However, this could open a possible vulnerability, or cause a bad user experience such as mixed content warning.

You may also need to set up a TLS certificate for your CloudFront distribution, when you serve the application with your own domain name (other than the default, *.cloudfront.net). In this case you will leverage AWS Certificate Manager (ACM), in which you can request the issuance of a domain validation certificate, or import your existing certificate with no additional cost. ACM supports certificate renewal for Amazon-issued certificates as well.

Another important configuration is choosing the TLS security policy. CloudFront provides a number of security policies, each of which defines available TLS versions and cipher suites for viewer HTTPS. The oldest security policy still includes SSLv3 to support legacy clients, but we recommend that you use the latest security policy. This way the viewer HTTPS connection will not use the old cipher, and it will get the benefit of the latest cipher; for example, perfect forward secrecy (PFS).

Finally, you can set up Server Name Indication (SNI) to be on or off. SNI is a TLS extension, which enables the client to put the server name in the initial TLS handshake "Client Hello" step. Because CloudFront serves multiple domains, this extension helps CloudFront to choose the correct TLS certificate to provide to client. SNI is supported by most modern browsers, but if you need to enable using a legacy client that doesn't support SNI, you can choose to turn SNI support off, which will incur extra cost. According to caniuse.com, more than 99% of clients use SNI-enabled agents.

CloudFront also supports Online Certificate Status Protocol (OCSP) stapling without further configuration, which enhances the performance of HTTPS delivery by providing a certificate validation response together with TLS certification during the TLS handshake.

For Viewer-side HTTPS, TLS 1.3 is automatically supported when the client initiates a TLS 1.3 handshake. TLS 1.3 eliminates weak cipher suites, provides perfect forward secrecy, and support 1-RTT TLS handshakes. Perfect forward secrecy means each session will not be compromised, even if the private key used in session key exchange is compromised in the future. Because previous sessions are protected, attackers have limited access to private data. TLS 1.3 mandates this by using only ephemeral key exchange algorithms such as Elliptic Curve Diffie-Hellman (ECDHE).

TLS 1.3 reduces TLS handshakes to one round trip, compared to two round trips in previous TLS 1.2. This improves the TLS handshake latency up to 33% using the AWS internal test.

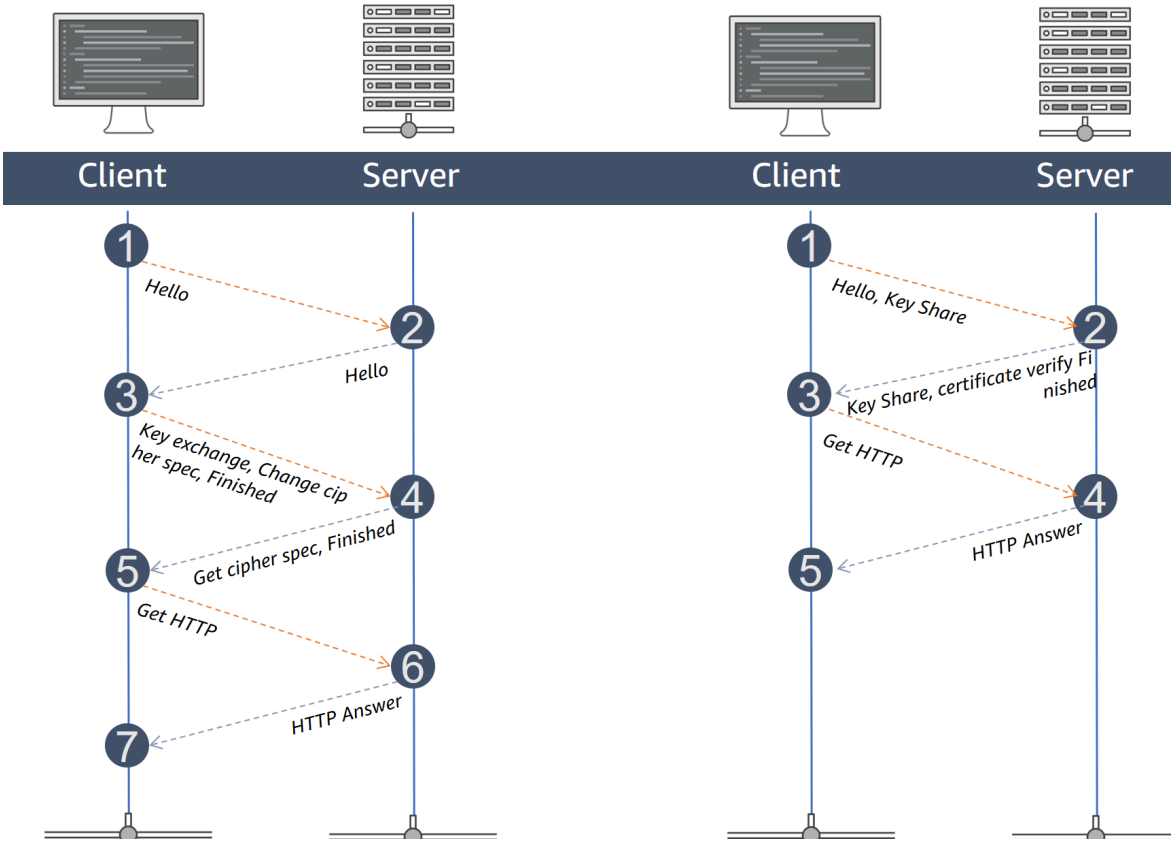

Figure 1 — 2-RTT TLS handshake (TLS 1.2) and 1-RTT TLS handshake (TLS 1.3)

Origin HTTPS configuration

When connecting to the origin with HTTPS, CloudFront becomes a client who initiates a TLS handshake. In this context, CloudFront operation is focused on authenticating the origin to avoid possible man-in-the-middle attacks. CloudFront trusts only the origin server when the TLS certificate provided by the origin server meets the following conditions:

- The certificate matches the requested domain name
- The certificate is issued by a trusted certificate authority (CA)
- The certificate chain is correct
- The certificate is not expired

If any of these conditions are not met, CloudFront drops the connection to the origin and responds with an error message to the client.

CloudFront validates either the Common Name (CN) or Subject Alternate Name (SAN) of the origin TLS certificate with the origin domain name that is registered in the CloudFront Distribution Origin Settings by default. For example, if a CloudFront Distribution is set up to deliver https://www.example.com/ to viewers and forward requests to origin.example.com, the origin TLS certificate must contain origin.example.com in its CN or SAN, either the exact name or in wildcard form, such as *.example.com.

CloudFront uses the Host header value instead if you configure it to forward the Host header, which means that the origin's TLS certificate must contain www.example.com in its CN or SAN. CloudFront also uses those domain names as an SNI in its initial TLS handshake. You may want to utilize this fact when using an Application Load Balancer (ALB) as an origin, because you must install a TLS certificate and its CN or SAN will not validate the default ALB domain (for example, alb-1234.us-east-1.elb.amazonaws.com) but will validate your own domain (www.example.com) instead. Alternatively, you can set a DNS record, such as origin.example.com, set the origin TLS certificate to validate that domain name, and associate the DNS record to the ALB via Amazon Route53 or another domain name server.

CloudFront verifies whether the origin TLS certificate was issued by a trusted certificate authority (CA) by comparing the issuer with the Mozilla included CA Certificate List. Certificates that are issued by other CAs, such as private certificates, will not be validated and the connection will subsequently fail.

CloudFront enables you to specify the minimum TLS protocol version to connect to the origin. Using the latest version of TLS on both CloudFront Distribution and origin setting will avoid known attacks, such as POODLE or BEAST, which exploit older versions of TLS.

Securing your contents with CloudFront

CloudFront provides a number of methods to help protect content from unwanted access. You can specify geographically-based restrictions, have CloudFront authorize access to content based on a signature (in the query string or in a cookie), and have CloudFront encrypt sensitive data fields so that only systems with the proper rights can access them.

Geo-based content access

When you deliver content that should have restrictions based on viewer's location, you can use CloudFront's Geo-Restrictions feature to implement this request. For example, video on demand (VOD) usually has a contractual condition that dictates only certain countries can play it. When Geo-Restrictions are enabled and configured to allow or deny the delivery to certain counties, CloudFront validates the

viewer's country information with this setting. When a viewer in a denied country tries to access the content, CloudFront responds with an error message.

Authorize access at the edge with signed URL and cookies

Another way you can help protect your content is to use signed URLs or signed cookies provided by CloudFront. When privileged or confidential content, such as paid streaming or confidential reports, needs to be delivered to authenticated viewers, you can leverage CloudFront's signed URLs. Signed URLs validate parameters in the query string or cookies, and allow access only when they are correctly signed with the private key of a pre-registered key pair. You can build an application that authenticates users and share the signed URL or cookies to minimize origin authentication efforts, cache the content in CloudFront, and protect against unauthenticated viewer access. Because signed URLs are based on a key pair system, content will not be compromised even if the public key is exposed. This can provide a better security posture in comparison to shared, key-based token authorization.

You can start using signed URLs by creating or uploading a key pair into your account security credentials. When a group of URLs, defined by a path pattern, is set to use signed URLs or cookies, CloudFront looks for the required parameters and validates their values on the request of those URLs. These parameters can be in the query string or cookie, where the former takes precedence if both are present. The fields in signed URLs or cookies vary slightly, based on which policy they use. Policy defines conditions which must be matched to access the contents.

There are two types of policies that you can use: a canned policy or a custom policy.

- A *canned policy* is the simpler of the two, and allows access if the request is made before the expiration time defined in the policy.
- A *custom policy* has conditions such as start date time, end date time, and IP address range in Classless Inter-domain Routing (CIDR) form.

When the condition is decided, signed URLs can be generated by signing the policy statement with the private key. See code examples in the Amazon CloudFront Developer Guide.

Using CloudFront to encrypt sensitive data at the edge

If there is more than one application or service behind the origin end point, you may want to protect sensitive data that the viewer submits via HTTP POST to the origin application stack. By using end-to-end HTTPS with CloudFront, you can help to ensure that sensitive data is encrypted from the client to the origin. You can add an additional layer of security by encrypting fields in the POST form with a public key at the edge, and allow only certain applications to decrypt it. For example, if an e-commerce system receives recipient information, the phone number or address can be separately encrypted, and is available only to the delivery application in decrypted form. CloudFront's Field-Level Encryption (FLE) can be used in this scenario.

To use CloudFront's FLE, you first need to upload the public key of a key pair. With this public key, you can make an FLE profile that defines which fields should be encrypted. Next, create a configuration that defines which content type (decided by an HTML form element's `enctype` attribute) is associated with an FLE profile. When these fields are encrypted from CloudFront and forwarded to the origin, the relevant application can decrypt them using a private key.

Protecting your origin by allowing access to CloudFront only

Controlling access to the origin is necessary, along with controlling viewer access, to have secure delivery via CloudFront. In this context, origin should allow requests only from CloudFront, and shut down attempts from any others.

S3 origin with CloudFront

S3 provides access control in conjunction with AWS Identity and Access Management (AWS IAM), bucket policy, bucket ACL, and object ACL. When using S3 origin with CloudFront, you can use CloudFront Origin Access Control (OAC) to secure S3 bucket access. OAC creates a principal that S3 can authenticate with, and it is used in a CloudFront distribution. By allowing the CloudFront service principal, an `s3:GetObject` action in the bucket policy, S3 allows CloudFront distribution to access to the content.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowCloudFrontServicePrincipalReadOnly",
        "Effect": "Allow",
        "Principal": {
            "Service": "cloudfront.amazonaws.com"
        },
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::<S3 bucket name>/*",
        "Condition": {
            "StringEquals": {
                "AWS:SourceArn": "arn:aws:cloudfront::<AWS account ID>:distribution/
<CloudFront distribution ID>"
            }
        }
    }
}
```

With this bucket policy set, you can turn on “Block all public access” to make the S3 bucket reachable only through CloudFront.

Custom origin with CloudFront

Unlike S3, which is a static object storage system, custom origins such as web servers can inspect incoming HTTP requests and decide to discard the request. You can allow only trusted CloudFront distribution to access your origin by adding a custom header with a secret value to the origin request in CloudFront, and setting up header inspection from the origin side. ALB has a rule that can be used for this header inspection purpose, if the origin web server is on AWS.

Figure 2 — Adding a secret header from the CloudFront console

CloudFront publishes its IP address ranges along with other Amazon services, without any prior setting or cost requirements. This enables you to allow CloudFront IP address ranges in your origin firewall, or add them into security groups. You may need more than one security group to allow all CloudFront IP ranges.

By using an IP allowed list and header inspection together, custom origin allows only chosen CloudFront distribution to access its private content.

Improving security by enabling security specific headers

To improve the security of your content, you can use HTTP security headers that are natively supported by the HTTP protocol and most modern browsers. These security headers tell the browser how to behave when handling website content. They can do things such as enforced communications over HTTPS, or defining from where JavaScript content can be loaded.

The Open Web Application Security Project (OWASP), provides guidance on the implementation of HTTP security headers to improve the security of your application, through its secure headers project, the latest guidance includes examples and best practices.

Security headers are commonly implemented using the web application configuration, but alternatively you can configure CloudFront to add those security response headers for your application if required. CloudFront provides this configuration through a response headers policy, and it comes with some managed policies that already has security headers such as `Strict-Transport-Security`, `X-Frame-Options`, `X-Content-Type-Options`, and so on.

Another consideration for enhanced security using HTTP headers is the appropriate configuration of Cross-origin resource sharing (CORS). In modern applications, the use of cross-domain resources is a necessity. The default restriction from browsers that only allows content from the same origin is impractical. To allow requests that have different origins (domain, protocol, or port), CORS must be enabled.

A number of HTTP headers relate to CORS, but two response headers are most important for security:

- `Access-Control-Allow-Origin` specifies which domains can access a site.
- `Access-Control-Allow-Methods` specifies which HTTP request methods (`GET`, `PUT`, `DELETE`, and others) can be used to access resources.

CloudFront support the configuration of these CORS response headers with the response headers policy. You can choose to use managed policies, or you can customize CORS behavior to allow only a specific origin web site to use the resources that you're sharing.

To understand more about configuring the response headers policy, refer the Amazon CloudFront Developer Guide.

Protecting from external threats at the edge

When your application runs on AWS, you can leverage both CloudFront and AWS WAF to help defend against application layer DDoS attacks.

By using AWS WAF, you can configure web access control lists (Web ACLs) on your CloudFront distributions to filter and block requests based on request signatures. Each Web ACL consists of rules that you can configure to string match or Regular Expression (regex) match one or more request attributes, such as the URI, query string, HTTP method, or header key.

By using AWS WAF's rate-based rules, you can automatically block the IP addresses of bad actors when requests matching a rule exceed a threshold that you define. Requests from offending client IP addresses will receive a "403 Forbidden" error response, and will remain blocked until request rates drop below the threshold. This is useful for mitigating HTTP flood attacks that are disguised as regular web traffic.

To block attacks from known bad-acting IP addresses, you can create rules using IP match conditions. You can use AWS-managed WAF rules, or use Managed Rules for AWS WAF offered by sellers in the AWS Marketplace. These rule sets can block specific malicious IP addresses that are included in IP reputation lists. Both AWS WAF and CloudFront enable you to set geo-restrictions to block or allow requests from selected countries. This can help block attacks originating from geographic locations where you do not expect to serve users.

If you are subscribed to AWS Shield Advanced, you can engage the AWS DDoS Response Team (DRT) to help you create rules to mitigate an attack that is hurting your application's availability. For more information, see Configure AWS DRT support.

You can use AWS Firewall Manager to centrally configure and manage AWS WAF rules for your CloudFront distributions across your organization. Your AWS Organizations management account can designate an administrator account, which is authorized to create Firewall Manager policies. These policies allow you to define criteria, such as resource type and tags, which determine where rules are applied. This is useful in case you have many accounts and want to standardize your protection. Firewall Manager also allows you to create policies that manage AWS Shield-protected resources and VPC security groups.

- To learn more about using geo-restriction to limit access to your CloudFront distribution, see Restricting the Geographic Distribution of Your Content.
- To learn more about using AWS WAF, see Getting started with AWS WAF.
- To learn more about configuring rate-based rules, see Protect Web Sites & Services Using Rate-Based Rules for AWS WAF.
- To learn how to manage the deployment of AWS WAF rules across your AWS resources with AWS Firewall Manager, see Getting started with AWS Firewall ManagerAWS WAF policies.

Managing access permissions to your CloudFront resources

To perform operations to your CloudFront resources, such as creating a distribution or invalidating an object, you can use AWS IAM. IAM helps you securely control access to AWS resources by enabling you to control who is authenticated (signed in) and authorized (has permissions) to use resources. You can create and manage AWS users, groups, roles, and permissions to allow or deny actions within AWS services.

IAM offers AWS managed policies created and administrated by AWS that are designed to provide permissions for many common use cases. For example, `CloudFrontFullAccess` provides full access to the CloudFront console plus the ability to list S3 buckets. `CloudFrontReadOnlyAccess` provides access to CloudFront distribution configuration information, and list distributions. In some scenarios, you might want to grant a specific level of access to a resource that you specify. For example, you might allow update, delete, and create invalidations access to a distribution that you specify by the distribution's Amazon Resource Name (ARN). You can create your own customer-managed policies that you administer in your own AWS account.

You can attach these policies to multiple principal entities to give each entity the permissions that are defined in the policy. These are referred as *Identity-Based* policies. You can also attach policies to resources, referred as *resource-based* policies. Even though CloudFront does not support resource-based policies, it does support resource-level policies that enable you to use ARNs to specify individual resources in a policy.

You can define more granular permissions with tag-based policies by granting access to specific resources, based in an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can define which users can perform actions on a distribution, based on the

tag that it already has. For example, tagging resources and using tags in policy statement conditions can help you grant access to specific resources, as the number of those resources grows over time, without having to continuously add them to the policy.

```
"Condition": {
         "StringEquals": {
             "aws:RequestTag/stage": [
                 "gamma",
                 "prod"
              ]
         }
     }
```

For a list of actions that you can specify to grant or deny permissions to use each action, see Actions, resources, and condition keys for Amazon CloudFront in the IAM User Guide.

Logging and monitoring in CloudFront

Logging and monitoring are an important part of maintaining the availability and performance of CloudFront. AWS provides several tools for monitoring your CloudFront resources and activity logs to help you respond to potential incidents.

The first service to consider is Amazon CloudWatch, because it provides you with data and actionable insights from your resources. CloudFront is integrated with Cloudwatch, and automatically publishes six operational metrics per distribution. You can monitor these metrics to detect anomalous behavior and create alarms. You can watch a single metric over a time period that you specify. If the metric exceeds a given threshold, a notification is sent to an Amazon SNS topic or an AWS Auto Scaling policy.

For example, receiving a notification when the 4xx error rate exceeds 1% (which helps you identify clients receiving 403 Forbidden errors) assists you to quickly identify the start of a possible DDoS attack. So does receiving a notification that the total amount of requests exceeds the expected value specified in units.

There are additional metrics you can enable for a cost. These units must be enabled for each individual distribution that requires it. For example, you can monitor and combine error rate and cache hit rate to measure CloudFront efficiency and alert on events as they occur, monitor a drop in the CHR, or monitor an increase in 5xx errors.

You have also the option to enable CloudFront access logs, which provide detailed records about requests that are made to a distribution. This access log information can be useful in security and access audits. You can enable standard logs, at no additional cost, that are delivered to the S3 buckets of your choice. Another option is real-time logs that, for a fee, are delivered within seconds of receiving the requests to the data stream of your choice in Amazon Kinesis Data Streams. Querying these logs enables you to explore users' surfing patterns across your web properties that are served by CloudFront. For example, you can query for detailed HTTP status code responses on a certain day or hour, or statistics based on the URI paths.

It's good practice to review CloudFront service activity with AWS CloudTrail, which provides a record of actions taken by a user, role, or AWS service in CloudFront by automatically recording and storing event logs. Using the information collected by CloudTrail, you can determine the request that was made to CloudFront, the IP address from which the request was made, who made the request, when it was made, and other additional details.

CloudTrail helps you track and automatically respond to activity threatening the security of your AWS resources with Amazon CloudWatch Events integration. You can monitor specific CloudFront API requests by creating a CloudWatch Event rule. A rule matches incoming events and routes them

to targets for processing. For example, you can create a rule to trigger an SNS topic when the API `UpdateDistribution` is requested.

Configuration management

To record and evaluate configurations of your AWS resources, you can use AWS Config, which provides you with a detailed view of the configuration of your distributions. This includes how the resources are related to one another and how they were configured in the past, so you can see how the configurations and relationships change over time. Examples of CloudFront-related resources are AWS WAF WebACL, AWS Certificate Manager, and S3 buckets. You can also evaluate configuration against desired configurations with AWS Config Rules.

For example, AWS Config Rules helps you to evaluate whether your CloudFront resources comply with common security best practices. You can choose managed rules like viewer policy HTTPS, SNI enabled, OAI enabled, origin failover enabled, WAF WebACL, or Shield resource policies to be triggered when the configuration changes. Managed rules can periodically run evaluations at a frequency that you choose; for example, every 24 hours. AWS Firewall Manager relies on AWS Config for automatic alerts and remediations.

Conclusion

This document has reviewed the security measures in Amazon CloudFront infrastructure, and reviewed how data can be securely delivered using CloudFront. CloudFront provides private data protection with HTTPS and various supporting features, and protects the application from external threats such as DDoS attacks. By combining other services, CloudFront can support your holistic security governance requirements.

Contributors

Contributors to this document include:

- Diana Alvarado, Technical Account Manager, Amazon Web Services
- Julian Ju, Edge Specialist Solutions Architect, Amazon Web Services.
- Rodrigo Ferroni, Security Specialist TAM, Amazon Web Services.

Further Reading

For additional information, see:

- *Guidelines for Implementing AWS WAF* (whitepaper)
- SIEM Solution using OpenSearch Service

Document revisions

To be notified about updates to this whitepaper, subscribe to the RSS feed.

Change	Description	Date
Update (p. 18)	Minor updates to align with new OWASP recommendations	January 11, 2022
Update (p. 18)	Significant updates	February 13, 2021
Initial publication (p. 18)	Whitepaper first published	July 1, 2017

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.