
Amazon CloudFront

Developer Guide

API Version 2016-09-29



Amazon CloudFront: Developer Guide

Copyright © 2018 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

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

Table of Contents

| | |
|---|----|
| What Is Amazon CloudFront? | 1 |
| How CloudFront Delivers Content | 4 |
| How CloudFront Works with Regional Edge Caches | 6 |
| Locations and IP Address Ranges of CloudFront Edge Servers | 7 |
| CloudFront Compliance | 8 |
| PCI DSS | 8 |
| HIPAA | 8 |
| Logging CloudFront Usage Data for Auditing | 8 |
| AWS Billing and Usage Reports for CloudFront | 9 |
| AWS Billing Report for CloudFront | 10 |
| AWS Usage Report for CloudFront | 11 |
| Interpreting Your AWS Bill and the AWS Usage Report for CloudFront | 12 |
| CloudFront Reports | 15 |
| CloudFront Cache Statistics Reports | 16 |
| Downloading Data in CSV Format | 17 |
| How Cache Statistics Charts Are Related to Data in the CloudFront Access Logs | 19 |
| CloudFront Popular Objects Report | 20 |
| Downloading Data in CSV Format | 21 |
| How Data in the Popular Objects Report Is Related to Data in the CloudFront Access Logs | 22 |
| CloudFront Top Referrers Report | 23 |
| Downloading Data in CSV Format | 24 |
| How Data in the Top Referrers Report Is Related to Data in the CloudFront Access Logs | 25 |
| CloudFront Usage Reports | 26 |
| Downloading Data in CSV Format | 27 |
| How the Usage Charts Are Related to Data in the CloudFront Usage Report | 29 |
| CloudFront Viewers Reports | 30 |
| Displaying Viewers Charts and Reports | 31 |
| Downloading Data in CSV Format | 31 |
| How Data in the Locations Report Is Related to Data in the CloudFront Access Logs | 37 |
| Getting Started | 38 |
| Step 1: Sign up for Amazon Web Services | 38 |
| Step 2: Upload your content to Amazon S3 and grant object permissions | 38 |
| Step 3: Create a CloudFront Web Distribution | 40 |
| Step 4: Test your links | 46 |
| Working with Distributions | 47 |
| Overview of Web and RTMP Distributions | 48 |
| Web Distributions | 48 |
| RTMP Distributions | 48 |
| Creating Web and RTMP Distributions | 49 |
| Listing, Viewing, and Updating CloudFront Distributions | 49 |
| Deleting a Distribution | 50 |
| Using Alternate Domain Names (CNAMEs) | 50 |
| Using the * Wildcard in Alternate Domain Names | 51 |
| Restrictions on Using Alternate Domain Names | 51 |
| Adding an Alternate Domain Name | 52 |
| Choosing the Price Class for a CloudFront Distribution | 55 |
| Using CloudFront with Amazon S3 | 55 |
| Adding CloudFront When You're Distributing Content from Amazon S3 | 56 |
| Moving an Amazon S3 Bucket to a Different Region | 57 |
| Changes to the CloudFront API | 57 |
| Working with Web Distributions | 59 |
| Task List for Creating a Web Distribution | 59 |
| Creating or Updating a Web Distribution Using the CloudFront Console | 60 |
| Testing Your Web Distribution | 61 |

| | |
|---|-----|
| Using Amazon S3 Origins and Custom Origins for Web Distributions | 62 |
| Using Amazon S3 Buckets for Your Origin | 62 |
| Using Amazon EC2 or Other Custom Origins | 63 |
| Values That You Specify When You Create or Update a Web Distribution | 63 |
| Delivery Method | 65 |
| Origin Settings | 65 |
| Cache Behavior Settings | 70 |
| Distribution Details | 77 |
| Custom Error Pages and Error Caching | 82 |
| Restrictions | 83 |
| Values that CloudFront Displays in the Console When You Create or Update a Web Distribution | 84 |
| Distribution ID (General Tab) | 84 |
| Distribution Status (General Tab) | 84 |
| Last Modified (General Tab) | 84 |
| Domain Name (General Tab) | 84 |
| Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins | 85 |
| Using Field-Level Encryption to Help Protect Sensitive Data | 85 |
| Overview of Field-Level Encryption | 87 |
| Setting Up Field-Level Encryption | 88 |
| Decrypting Data Fields at Your Origin | 91 |
| Using AWS WAF to Control Access to Your Content | 93 |
| Restricting the Geographic Distribution of Your Content | 94 |
| Using CloudFront Geo Restriction | 94 |
| Using a Third-Party Geolocation Service | 95 |
| Configuring On-Demand Smooth Streaming | 96 |
| Configuring On-Demand Progressive Downloads | 97 |
| Configuring On-Demand Apple HTTP Live Streaming (HLS) | 98 |
| Using CloudFront with Lambda@Edge | 99 |
| CloudFront Events That Can Trigger a Lambda Function | 100 |
| How You Decide Which CloudFront Event to Use to Trigger a Lambda Function | 101 |
| Requirements and Restrictions on Lambda Functions | 101 |
| CloudFront Distributions and Associations | 102 |
| CloudFront Triggers for Lambda Functions | 102 |
| CloudWatch Logs | 102 |
| Headers | 102 |
| HTTP Status Codes | 105 |
| Lambda Function Configuration and Execution Environment | 105 |
| Limits | 105 |
| Microsoft Smooth Streaming | 105 |
| Network Access | 105 |
| Query String Parameters | 105 |
| URI | 106 |
| URI and Query String Encoding | 106 |
| Creating Lambda Functions and Adding Triggers | 107 |
| Creating Lambda Functions (Console) | 107 |
| Adding Triggers for CloudFront Events to a Lambda Function | 107 |
| Event Structure | 108 |
| Request Event | 109 |
| Response Event | 112 |
| Generating HTTP Responses in Request Triggers | 114 |
| Programming Model | 114 |
| Updating HTTP Responses in an Origin-Response Trigger | 116 |
| CloudWatch Metrics and CloudWatch Logs for Lambda Functions | 117 |
| CloudWatch Metrics | 117 |
| CloudWatch Logs | 117 |
| Example Functions | 118 |
| Example: A/B Testing | 118 |

| | |
|---|-----|
| Example: Overriding a Response Header | 119 |
| Example: Serving Static Content (Generated Response) | 120 |
| Example: Generating an HTTP Redirect (Generated Response) | 121 |
| Example: Adding a Header Based on a Query String Parameter | 121 |
| Example: Normalizing Query String Parameters to Improve the Cache Hit Ratio | 122 |
| Example: Redirecting Unauthenticated Users to a Sign-In Page | 122 |
| Example: Redirecting Viewer Requests to a Country-Specific URL | 123 |
| Example: Serving Different Versions of an Object Based on the Device | 124 |
| Example: Serving Static Website Content as Gzip Compressed Content | 125 |
| Example: Using an Origin-Request Trigger to Change From a Custom Origin to an Amazon S3 Origin | 126 |
| Example: Using an Origin-Request Trigger to Change the Amazon S3 Origin Region | 127 |
| Example: Using an Origin-Request Trigger to Change From an Amazon S3 Origin to a Custom Origin | 127 |
| Example: Using an Origin-Request Trigger to Gradually Transfer Traffic From One Amazon S3 Bucket to Another | 128 |
| Example: Using an Origin-Request Trigger to Change the Origin Domain Name Based on the Country Header | 129 |
| Example: Using an Origin-Response Trigger to Update the Error Status Code to 200-OK | 129 |
| Example: Using an Origin-Response Trigger to Update the Error Status Code to 302-Found | 130 |
| Working with RTMP Distributions | 131 |
| How RTMP Distributions Work | 131 |
| Task List for Streaming Media Files Using RTMP | 132 |
| Creating an RTMP Distribution Using the CloudFront Console | 133 |
| Values that You Specify When You Create or Update an RTMP Distribution | 134 |
| Origin Domain Name (Amazon S3 Bucket) | 134 |
| Restrict Bucket Access (Amazon S3 Only) | 135 |
| Origin Access Identity (Amazon S3 Only) | 135 |
| Comment for New Identity(Amazon S3 Only) | 135 |
| Your Identities (Amazon S3 Only) | 135 |
| Grant Read Permissions on Bucket (Amazon S3 Only) | 135 |
| Price Class | 136 |
| Alternate Domain Names (CNAMEs) | 136 |
| Logging | 136 |
| Bucket for Logs | 136 |
| Log Prefix | 136 |
| Comment | 137 |
| Distribution State | 137 |
| Restrict Viewer Access (Use Signed URLs) | 137 |
| Trusted Signers | 137 |
| AWS Account Numbers | 138 |
| Values that CloudFront Displays in the Console When You Create or Update an RTMP Distribution | 138 |
| Distribution ID | 138 |
| Status | 138 |
| Last Modified | 138 |
| Domain Name | 139 |
| Configuring the Media Player | 139 |
| MPEG Files | 139 |
| Using an Amazon S3 Bucket as the Origin for an RTMP Distribution | 140 |
| Creating Multiple RTMP Distributions for an Origin Server | 140 |
| Restricting Access Using Crossdomain.xml | 141 |
| Error Codes for RTMP Distributions | 141 |
| Troubleshooting RTMP Distributions | 141 |
| Working with Objects | 143 |
| Format of URLs for Objects | 143 |
| Using Your Own Domain Name (Example.com) | 144 |
| Using a Trailing Slash (/) in URLs | 144 |

| | |
|--|-----|
| Creating URLs for Streaming | 144 |
| Creating Signed URLs for Restricted Content | 144 |
| How CloudFront Processes HTTP and HTTPS Requests | 144 |
| Increasing the Proportion of Requests that Are Served from CloudFront Edge Caches | 145 |
| Specifying How Long CloudFront Caches Your Objects | 145 |
| Caching Based on Query String Parameters | 145 |
| Caching Based on Cookie Values | 146 |
| Caching Based on Request Headers | 147 |
| Serving Media Content by Using HTTP | 147 |
| Configuring CloudFront to Cache Based on Query String Parameters | 147 |
| Console and API Settings for Query String Forwarding and Caching | 149 |
| Optimizing Caching | 149 |
| Query String Parameters and CloudFront Access Logs | 150 |
| Configuring CloudFront to Cache Objects Based on Cookies | 150 |
| Configuring CloudFront to Cache Objects Based on Request Headers | 152 |
| Headers and Web Distributions | 152 |
| Headers and RTMP Distributions | 156 |
| Forwarding Custom Headers to Your Origin (Web Distributions Only) | 156 |
| Configuring CloudFront to Forward Custom Headers to Your Origin | 156 |
| Custom Headers that CloudFront Can't Forward to Your Origin | 157 |
| Using Custom Headers for Cross-Origin Resource Sharing (CORS) | 157 |
| Using Custom Headers to Restrict Access to Your Content on a Custom Origin | 158 |
| Adding, Removing, or Replacing Objects in a Distribution | 158 |
| Adding Objects that You Want CloudFront to Distribute | 159 |
| Updating Existing Objects Using Versioned Object Names | 159 |
| Updating Existing Objects Using the Same Object Names | 159 |
| Specifying How Long Objects Stay in a CloudFront Edge Cache (Expiration) | 160 |
| Invalidating Objects (Web Distributions Only) | 165 |
| Customizing Error Responses | 172 |
| Creating or Updating a Cache Behavior for Custom Error Pages | 173 |
| Changing Response Codes | 173 |
| Controlling How Long CloudFront Caches Errors | 174 |
| How CloudFront Responds When a Custom Error Page Is Unavailable | 175 |
| Pricing for Custom Error Pages | 175 |
| Configuring Error Response Behavior | 175 |
| How CloudFront Processes Partial Requests for an Object (Range GETs) | 176 |
| Specifying a Default Root Object (Web Distributions Only) | 177 |
| Serving Compressed Files | 179 |
| Using CloudFront to Compress Your Content | 179 |
| Using a Custom Origin to Compress Your Content | 182 |
| Request and Response Behavior | 184 |
| Request and Response Behavior for Amazon S3 Origins | 184 |
| How CloudFront Processes and Forwards Requests to Your Amazon S3 Origin Server | 184 |
| How CloudFront Processes Responses from Your Amazon S3 Origin Server | 189 |
| Request and Response Behavior for Custom Origins | 190 |
| How CloudFront Processes and Forwards Requests to Your Custom Origin Server | 190 |
| How CloudFront Processes Responses from Your Custom Origin Server | 199 |
| How CloudFront Processes HTTP 3xx Status Codes from Your Origin | 202 |
| How CloudFront Processes and Caches HTTP 4xx and 5xx Status Codes from Your Origin | 203 |
| How CloudFront Processes Errors When You Have Configured Custom Error Pages | 203 |
| How CloudFront Processes Errors When You Have Not Configured Custom Error Pages | 204 |
| HTTP 4xx and 5xx Status Codes that CloudFront Caches | 205 |
| Serving Private Content through CloudFront | 207 |
| Overview of Private Content | 207 |
| Restricting Access to Objects in CloudFront Edge Caches | 208 |
| Restricting Access to Objects in Amazon S3 Buckets | 209 |
| Using an HTTP Server for Private Content | 209 |

| | |
|---|-----|
| Task List: Serving Private Content | 210 |
| Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content | 211 |
| Creating a CloudFront Origin Access Identity and Adding it to Your Distribution | 212 |
| Granting the Origin Access Identity Permission to Read Objects in Your Amazon S3 Bucket | 213 |
| Using an Origin Access Identity in Amazon S3 Regions that Support Only Signature Version 4 Authentication | 215 |
| Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies (Trusted Signers) | 216 |
| Creating CloudFront Key Pairs for Your Trusted Signers | 217 |
| Reformatting the CloudFront Private Key (.NET and Java Only) | 218 |
| Adding Trusted Signers to Your Distribution | 219 |
| Verifying that Trusted Signers Are Active (Optional) | 221 |
| Rotating CloudFront Key Pairs | 221 |
| Choosing Between Signed URLs and Signed Cookies | 223 |
| Using Both Signed URLs and Signed Cookies | 223 |
| Using Signed URLs | 224 |
| Choosing Between Canned and Custom Policies for Signed URLs | 224 |
| How Signed URLs Work | 225 |
| Choosing How Long Signed URLs Are Valid | 225 |
| When Does CloudFront Check the Expiration Date and Time in a Signed URL? | 226 |
| Sample Code and Third-Party Tools | 226 |
| Creating a Signed URL Using a Canned Policy | 227 |
| Creating a Signed URL Using a Custom Policy | 234 |
| Using Signed Cookies | 244 |
| Choosing Between Canned and Custom Policies for Signed Cookies | 244 |
| How Signed Cookies Work | 244 |
| Preventing Misuse of Signed Cookies | 245 |
| When Does CloudFront Check the Expiration Date and Time in a Signed Cookie? | 246 |
| Sample Code and Third-Party Tools | 246 |
| Setting Signed Cookies Using a Canned Policy | 246 |
| Setting Signed Cookies Using a Custom Policy | 251 |
| Using a Linux Command and OpenSSL for Base64-Encoding and Encryption | 259 |
| Code Examples for Creating a Signature for a Signed URL | 260 |
| Create a URL Signature Using Perl | 260 |
| Create a URL Signature Using PHP | 268 |
| Create a URL Signature Using C# and the .NET Framework | 270 |
| Create a URL Signature Using Java | 276 |
| Using HTTPS with CloudFront | 278 |
| Requiring HTTPS for Communication Between Viewers and CloudFront | 278 |
| Requiring HTTPS for Communication Between CloudFront and Your Custom Origin | 280 |
| Changing CloudFront Settings | 280 |
| Installing an SSL/TLS Certificate on Your Custom Origin Server | 281 |
| Requiring HTTPS for Communication Between CloudFront and Your Amazon S3 Origin | 282 |
| Using Alternate Domain Names and HTTPS | 283 |
| Choosing How CloudFront Serves HTTPS Requests | 283 |
| Requirements for Using SSL/TLS Certificates with CloudFront | 285 |
| Limits on Using SSL/TLS Certificates with CloudFront (HTTPS Between Viewers and CloudFront Only) | 288 |
| Configuring Alternate Domain Names and HTTPS | 289 |
| Determining the Size of the Public Key in an SSL/TLS Certificate | 292 |
| Increasing the Limit for SSL/TLS Certificates | 293 |
| Rotating SSL/TLS Certificates | 294 |
| Reverting from a Custom SSL/TLS Certificate to the Default CloudFront Certificate | 295 |
| Switching from a Custom SSL/TLS Certificate with Dedicated IP Addresses to SNI | 295 |
| Supported Protocols and Ciphers | 296 |
| Supported SSL/TLS Protocols and Ciphers for Communication Between Viewers and CloudFront | 296 |

| | |
|--|-----|
| Supported SSL/TLS Protocols and Ciphers for Communication Between CloudFront and Your Origin | 298 |
| Charges for HTTPS Connections | 299 |
| Authentication and Access Control | 300 |
| Authentication | 300 |
| Access Control | 301 |
| Overview of Managing Access | 301 |
| ARNs for CloudFront Resources | 302 |
| Understanding Resource Ownership | 302 |
| Managing Access to Resources | 302 |
| Specifying Policy Elements: Resources, Actions, Effects, and Principals | 303 |
| Specifying Conditions in a Policy | 304 |
| Using IAM Policies for CloudFront | 304 |
| Permissions Required to Use the CloudFront Console | 305 |
| AWS Managed (Predefined) Policies for CloudFront | 306 |
| Customer Managed Policy Examples | 307 |
| CloudFront API Permissions Reference | 309 |
| Required Permissions for Actions on Web Distributions | 309 |
| Required Permissions for Actions on RTMP Distributions | 311 |
| Required Permissions for Actions on Invalidations | 312 |
| Required Permissions for Actions on Origin Access Identities | 312 |
| Required Permissions for CloudFront Actions Related to Lambda@Edge | 313 |
| Required Permissions for Actions on Tags | 314 |
| Access Logs | 315 |
| How Logging Works | 315 |
| Choosing an Amazon S3 Bucket for Your Access Logs | 316 |
| Permissions Required to Configure Logging and to Access Your Log Files | 316 |
| File Name Format | 317 |
| Timing of Log File Delivery | 317 |
| Analyzing Access Logs | 318 |
| Editing Your Logging Settings | 318 |
| Deleting Log Files from an Amazon S3 Bucket | 318 |
| Log File Format | 318 |
| Web Distribution Log File Format | 319 |
| RTMP Distribution Log File Format | 327 |
| Charges for Access Logs | 329 |
| Monitoring CloudFront Activity Using CloudWatch | 330 |
| Viewing Metrics for a Distribution | 330 |
| Receiving Notifications | 330 |
| Downloading Data in CSV Format | 331 |
| Information About the Report | 331 |
| Data in the CloudWatch Metrics Report | 332 |
| Amazon CloudFront Metrics | 332 |
| Dimensions for CloudFront Metrics | 333 |
| Capturing API Requests with CloudTrail | 335 |
| CloudFront Information in CloudTrail Log Files | 335 |
| Understanding CloudFront Log File Entries | 336 |
| Tagging Amazon CloudFront Distributions | 341 |
| Tag Restrictions | 341 |
| Adding, Editing, and Deleting Tags for Distributions | 342 |
| Troubleshooting | 343 |
| Troubleshooting Distribution Issues | 343 |
| I Can't View the Files in My Web Distribution | 343 |
| I Can't View the Files in My RTMP Distribution | 344 |
| Error Message: Certificate: <certificate-id> Is Being Used by CloudFront | 344 |
| Troubleshooting Error Responses from Your Origin | 345 |
| HTTP 502 Status Code (Bad Gateway) | 345 |

| | |
|--|-----|
| HTTP 503 Status Code (Service Unavailable) | 347 |
| Troubleshooting Lambda@Edge | 348 |
| Load Testing CloudFront | 350 |
| Streaming Tutorials | 351 |
| Live Streaming | 351 |
| On-Demand Streaming | 351 |
| RTMP Streaming | 351 |
| Live HTTP Streaming Using CloudFront and Any HTTP Origin | 352 |
| Creating a New CloudFront Distribution for Live Streaming | 352 |
| Configuring Web Players to Play the Live Stream | 353 |
| (Optional) Deleting an AWS CloudFormation Stack for Live Streaming | 353 |
| On-Demand Video Streaming Using CloudFront and JW Player | 354 |
| Uploading Media and JW Player Files to an Amazon S3 Bucket | 354 |
| Creating CloudFront Web and RTMP Distributions | 355 |
| Embedding Video in a Web Page | 356 |
| Uploading the HTML File and Playing the Video | 358 |
| Limits | 359 |
| General Limits | 359 |
| General Limits on Web Distributions | 359 |
| Limits on Whitelisted Cookies (Web Distributions Only) | 360 |
| Limits on Whitelisted Query Strings (Web Distributions Only) | 360 |
| Limits on Custom Headers (Web Distributions Only) | 361 |
| Limits on SSL Certificates (Web Distributions Only) | 361 |
| Limits on Invalidations | 361 |
| Limits on Field-Level Encryption | 362 |
| Limits on Lambda@Edge | 362 |
| Limits on RTMP Distributions | 363 |
| Resources | 364 |
| Additional Amazon CloudFront Documentation | 364 |
| Getting Support | 364 |
| CloudFront Developer Tools and SDKs | 365 |
| Using CloudFront Logging | 365 |
| Additional Tips from the Amazon Web Services Blog | 365 |
| Invalidating Objects | 365 |
| Distributing Streaming Media | 366 |
| Tools and Code Examples for Configuring Private Content | 366 |
| Using CloudFront with a Content Management System | 366 |
| Document History | 368 |
| AWS Glossary | 381 |

What Is Amazon CloudFront?

Amazon CloudFront is a web service that speeds up distribution of your static and dynamic web content, such as .html, .css, .js, and image files, to your users. CloudFront delivers your content through a worldwide network of data centers called edge locations. When a user requests content that you're serving with CloudFront, the user is routed to the edge location that provides the lowest latency (time delay), so that content is delivered with the best possible performance. If the content is already in the edge location with the lowest latency, CloudFront delivers it immediately. If the content is not in that edge location, CloudFront retrieves it from an Amazon S3 bucket or an HTTP server (for example, a web server) that you have identified as the source for the definitive version of your content.

This concept is best illustrated by an example. Suppose you're serving the following image from a traditional web server, not from CloudFront.



(The image is owned by NASA and comes from the Visible Earth website at <http://visibleearth.nasa.gov/>.)

You're serving the image using the URL `http://example.com/globe_west_540.png`. Your users can easily navigate to this URL and see the image, but they probably don't know that their request was routed from one network to another—through the complex collection of interconnected networks that comprise the internet—until the image was found.

Further suppose that the web server that you're serving the image from is in Seattle, Washington, USA, and that a user in Austin, Texas, USA requests the image. The following traceroute list (courtesy of www.WatchMouse.com) shows one way that this request could be routed.

| | | |
|----|--|-----------|
| 1 | vrid-225.core-sw.aus.us.siteprotect.com (216.139.225.1) | 0.627 ms |
| 2 | xe-3-4.brdr-rtr-02.aus.us.siteprotect.com (216.139.253.53) | 0.219 ms |
| 3 | 66.113.197.121 | 0.452 ms |
| 4 | xe-5-2-0.edge3.Dallas1.Level3.net (4.59.112.37) | 4.978 ms |
| 5 | ae-73-70.ebr3.Dallas1.Level3.net (4.69.145.116) | 9.817 ms |
| 6 | ae-7-7.ebr3.Atlanta2.Level3.net (4.69.134.22) | 30.570 ms |
| 7 | ae-2-2.ebr1.Washington1.Level3.net (4.69.132.86) | 38.801 ms |
| 8 | ae-81-81.csw3.Washington1.Level3.net (4.69.134.138) | 41.795 ms |
| 9 | ae-3-89.edge2.Washington1.Level3.net (4.68.17.145) | 39.193 ms |
| 10 | 72.21.222.139 | 35.767 ms |



Map courtesy of the University of Texas Libraries, The University of Texas at Austin

In this example, the request was routed 10 times within the United States before the image was retrieved, which is not an unusually large number of hops (the network paths between the source and the final destination). If your user were in Europe, the request would be routed through even more networks to reach your server in Seattle. The number of networks and the distance that the request and the image must travel have a significant impact on the performance, reliability, and availability of the image.

CloudFront speeds up the distribution of your content by routing each user request to the edge location that can best serve your content. Typically, this is the CloudFront edge location that provides the lowest latency. This dramatically reduces the number of networks that your users' requests must pass through, which improves performance. Users get lower latency—the time it takes to load the first byte of the file—and higher data transfer rates. You also get increased reliability and availability because copies of your files (also known as *objects*) are now held in multiple edge locations around the world.

For a list of the locations of CloudFront edge servers, see the [Amazon CloudFront Product Details page](#).

Topics

- [How CloudFront Delivers Content \(p. 4\)](#)
- [How CloudFront Works with Regional Edge Caches \(p. 6\)](#)
- [Locations and IP Address Ranges of CloudFront Edge Servers \(p. 7\)](#)
- [CloudFront Compliance \(p. 8\)](#)

How CloudFront Delivers Content

After some initial setup, CloudFront works invisibly to speed up delivery of your content. This overview includes both the steps you perform before your first user accesses your application or website and how CloudFront serves your content when configuration is complete.

Setting up CloudFront involves a few simple steps:

How You Configure CloudFront to Deliver Your Content

1. You configure your *origin servers*, from which CloudFront gets your files for distribution from CloudFront edge locations all over the world.

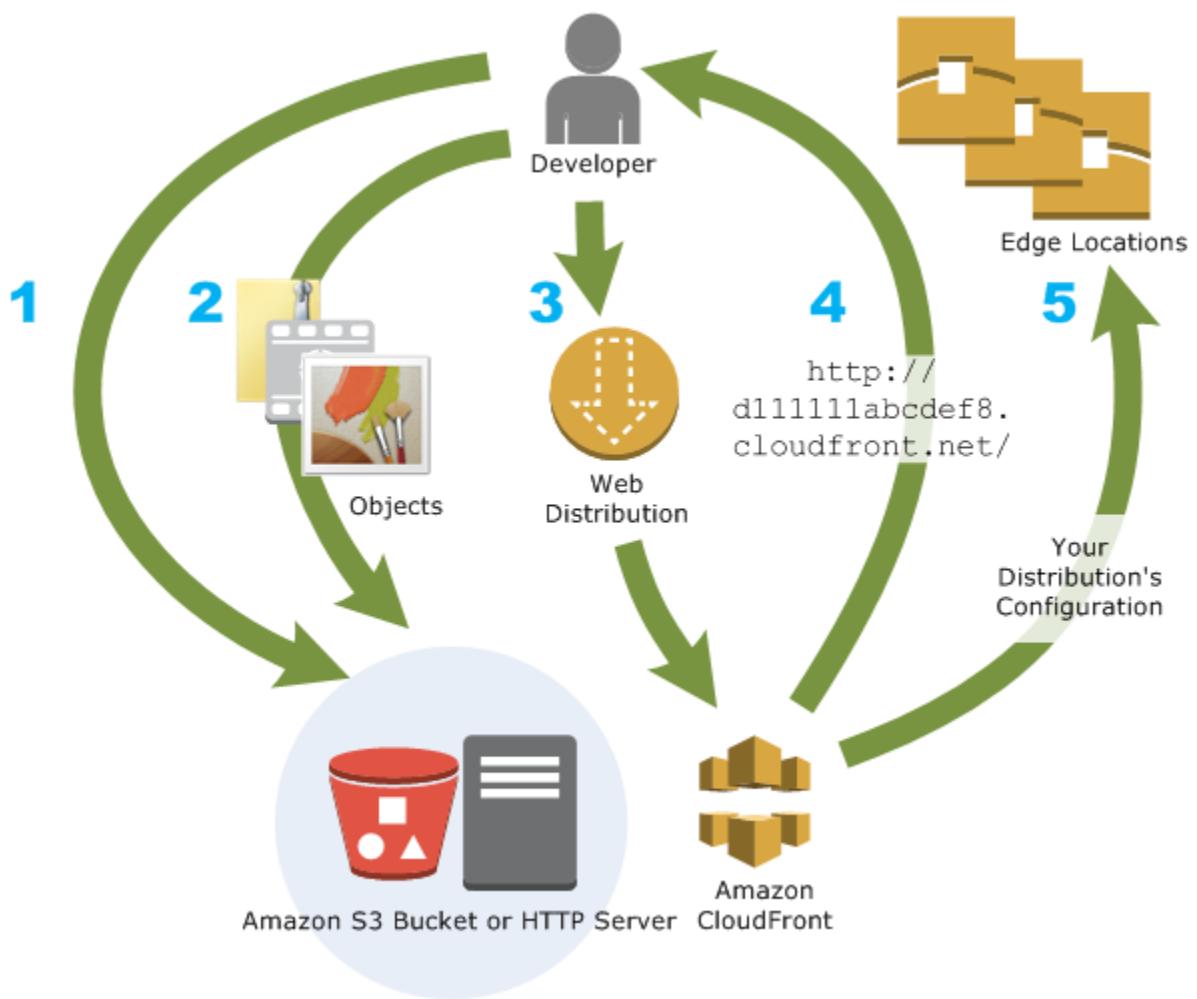
An origin server stores the original, definitive version of your objects. If you're serving content over HTTP, your origin server is either an Amazon S3 bucket or an HTTP server, such as a web server. Your HTTP server can run on an Amazon Elastic Compute Cloud (Amazon EC2) instance or on a server that you manage; these servers are also known as *custom origins*.

If you distribute media files on demand using the Adobe Media Server RTMP protocol, your origin server is always an Amazon S3 bucket.

2. You upload your files to your origin servers. Your files, also known as *objects*, typically include web pages, images, and media files, but can be anything that can be served over HTTP or a supported version of Adobe RTMP, the protocol used by Adobe Flash Media Server.

If you're using an Amazon S3 bucket as an origin server, you can make the objects in your bucket publicly readable, so that anyone who knows the CloudFront URLs for your objects can access them. You also have the option of keeping objects private and controlling who accesses them. See [Serving Private Content through CloudFront \(p. 207\)](#).

3. You create a CloudFront *distribution*, which tells CloudFront which origin servers to get your files from when users request the files through your web site or application. At the same time, you specify details such as whether you want CloudFront to log all requests and whether you want the distribution to be enabled as soon as it's created.
4. CloudFront assigns a domain name to your new distribution and displays it in the CloudFront console or returns it in the response to a programmatic request, for example, an API request.
5. CloudFront sends your distribution's configuration (but not your content) to all of its **edge locations**—collections of servers in geographically dispersed data centers where CloudFront caches copies of your objects.



As you develop your website or application, you use the domain name that CloudFront provides for your URLs. For example, if CloudFront returns `d111111abcdef8.cloudfront.net` as the domain name for your distribution, the URL for `logo.jpg` in your Amazon S3 bucket (or in the root directory on an HTTP server) will be `http://d111111abcdef8.cloudfront.net/logo.jpg`.

You can also configure your CloudFront distribution so you can use your own domain name. In that case, the URL might be `http://www.example.com/logo.jpg`.

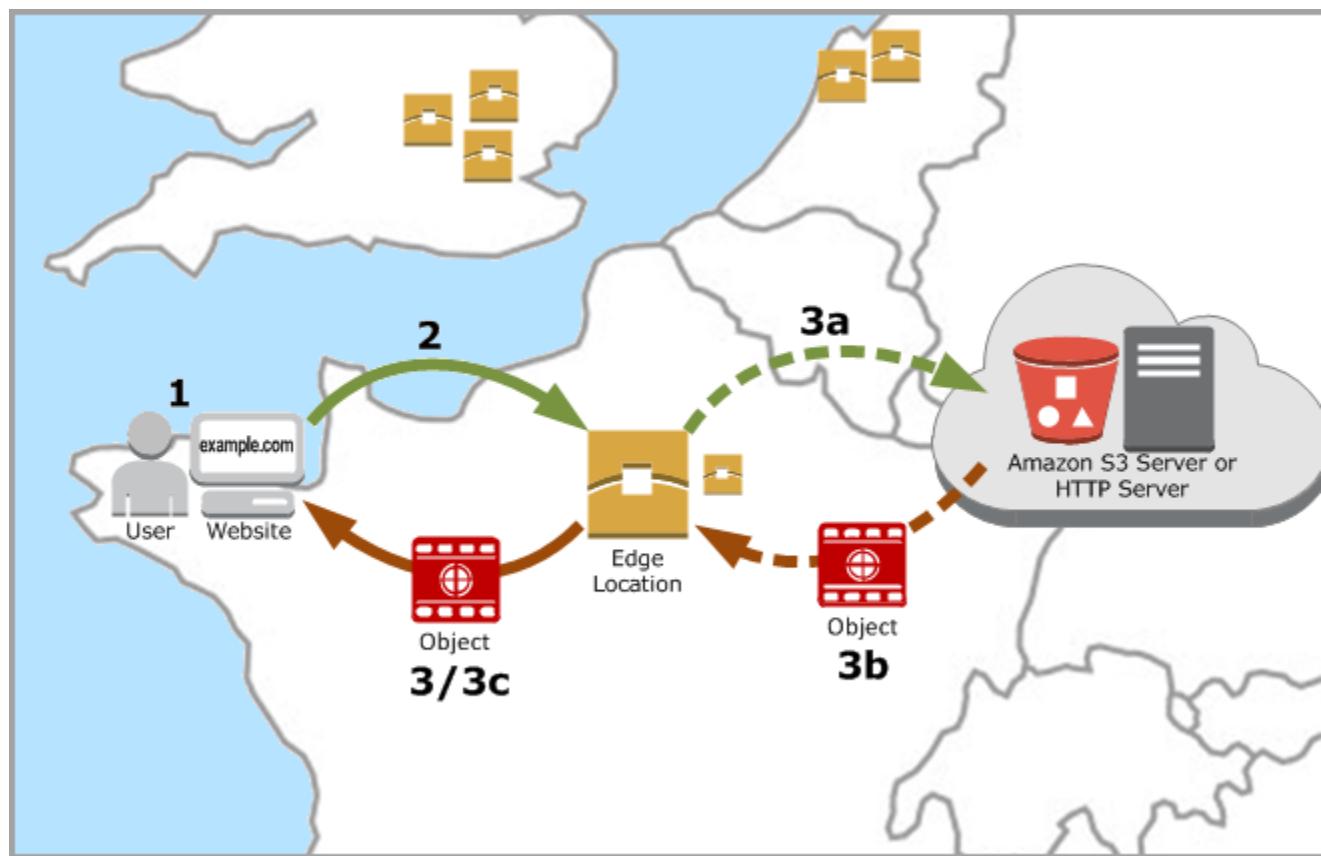
Optionally, you can configure your origin server to add headers to the files; the headers indicate how long you want the files to stay in the cache in CloudFront edge locations. By default, each object stays in an edge location for 24 hours before it expires. The minimum expiration time is 0 seconds; there isn't a maximum expiration time limit. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

How CloudFront Delivers Content to Your Users

Once you configure CloudFront to deliver your content, here's what happens when users request your objects:

1. A user accesses your website or application and requests one or more objects, such as an image file and an HTML file.

2. DNS routes the request to the CloudFront edge location that can best serve the user's request, typically the nearest CloudFront edge location in terms of latency, and routes the request to that edge location.
3. In the edge location, CloudFront checks its cache for the requested files. If the files are in the cache, CloudFront returns them to the user. If the files are *not* in the cache, it does the following:
 - a. CloudFront compares the request with the specifications in your distribution and forwards the request for the files to the applicable origin server for the corresponding file type—for example, to your Amazon S3 bucket for image files and to your HTTP server for the HTML files.
 - b. The origin servers send the files back to the CloudFront edge location.
 - c. As soon as the first byte arrives from the origin, CloudFront begins to forward the files to the user. CloudFront also adds the files to the cache in the edge location for the next time someone requests those files.



How CloudFront Works with Regional Edge Caches

CloudFront Regional Edge Caches bring more of your content closer to your viewers, even when the content is not popular enough to stay at a CloudFront edge location. This helps improve performance for your viewers, while lowering the operational burden and cost of scaling your origin resources.

This feature helps with all types of content, particularly content that tends to become less popular over time. Examples include user-generated content, such as video, photos, or artwork; e-commerce assets such as product photos and videos; and news and event-related content that might suddenly find new popularity.

How Regional Caches Work

Regional Edge Caches are CloudFront locations that are deployed globally, at close proximity to your viewers. These locations sit between your origin server and the global edge locations that serve traffic directly to your viewers. As the popularity of your objects reduce, individual edge locations may evict those objects to make room for more popular content. Regional Edge Caches have larger cache width than any individual edge location, so your objects remain in cache longer at the nearest regional edge cache location. This helps keep more of your content closer to your viewers, reducing the need for CloudFront to go back to your origin server, and improving overall performance for viewers.

When a viewer makes a request by your website or application, DNS routes the request to the CloudFront edge location that can best serve the user's request. This location is typically the nearest CloudFront edge location in terms of latency. In the edge location, CloudFront checks its cache for the requested files. If the files are in the cache, CloudFront returns them to the user. If the files are not in the cache, the edge servers go to the nearest Regional Edge Cache to fetch the object. In the Regional Edge Cache location, CloudFront again checks its cache for the requested files. If the files are in the cache, CloudFront forwards the files to the requested edge location. As soon as the first byte arrives from Regional Edge Cache location, CloudFront begins to forward the files to the user. CloudFront also adds the files to the cache in the requested edge location for the next time someone requests those files.

For files not cached at both the edge location and the Regional Edge Cache location, CloudFront compares the request with the specifications in your distributions and forwards the request for your files to the applicable origin server. Once your origin server sends the files back to the Regional Edge Cache location, they are forwarded to the requested edge location, and CloudFront forwards the files to the user. In this case, CloudFront also adds the files to the cache in the Regional Edge Cache location in addition to the edge location for the next time a viewer requests those files. Now, all edge locations in a region now share a local cache, eliminating multiple requests to your origin server. Additionally, CloudFront keeps persistent connections with your origin servers so files are fetched from the origin servers as quickly as possible.

Note

- You do not need to make any changes to your CloudFront distributions. Regional edge caches are enabled by default for all CloudFront distributions.
- There is no additional cost for using this feature.
- Regional Edge Caches have feature parity with edge locations. For example, a cache invalidation request removes an object from both edge caches and Regional Edge Caches before it expires. The next time a viewer requests the object, CloudFront returns to the origin to fetch the latest version of the object.
- Regional Edge Caches are available for custom origins. Amazon S3 origins are not supported.
- Proxy methods `PUT`/`POST`/`PATCH`/`OPTIONS`/`DELETE` flow directly to the origin from the edge locations and do not proxy through the Regional Edge Caches.
- Dynamic content as determined at request time (cache-behavior configured to forward all headers) does not flow through the Regional Edge Caches, but goes directly to the origin.
- You can measure the performance improvements from this feature by using cache-hit ratio metrics available on the console. To learn more about cache-hit ratio metrics, see [CloudFront Cache Statistics Reports \(p. 16\)](#).

Locations and IP Address Ranges of CloudFront Edge Servers

For a list of the locations of CloudFront edge servers, see the [Amazon CloudFront Product Details page](#).

Amazon Web Services (AWS) publishes its current IP address ranges in JSON format. To view the current ranges, download [ip-ranges.json](#). For more information, see [AWS IP Address Ranges](#) in the *Amazon Web Services General Reference*.

To find the IP address ranges that are associated with CloudFront edge servers, search `ip-ranges.json` for the following string:

```
"service": "CLOUDFRONT"
```

Alternatively, you can view only the CloudFront IP ranges [here](#).

CloudFront Compliance

CloudFront is compliant with the PCI DSS and HIPAA standards.

Topics

- [PCI DSS \(p. 8\)](#)
- [HIPAA \(p. 8\)](#)
- [Logging CloudFront Usage Data for Auditing \(p. 8\)](#)

PCI DSS

CloudFront supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS). For more information about PCI DSS, including how to request a copy of the AWS PCI Compliance Package, see [PCI DSS Level 1](#).

As a security best practice, we recommend that you don't cache credit card information in CloudFront edge caches. For example, you can configure your origin to include a `Cache-Control:no-cache="field-name"` header in responses that contain credit card information, such as the last four digits of a credit card number and the card owner's contact information.

HIPAA

AWS has expanded its HIPAA compliance program to include CloudFront as a HIPAA Eligible Service. If you have an executed Business Associate Agreement (BAA) with AWS, you can use CloudFront to deliver content containing protected health information (PHI). For more information, see [HIPAA Compliance](#).

Logging CloudFront Usage Data for Auditing

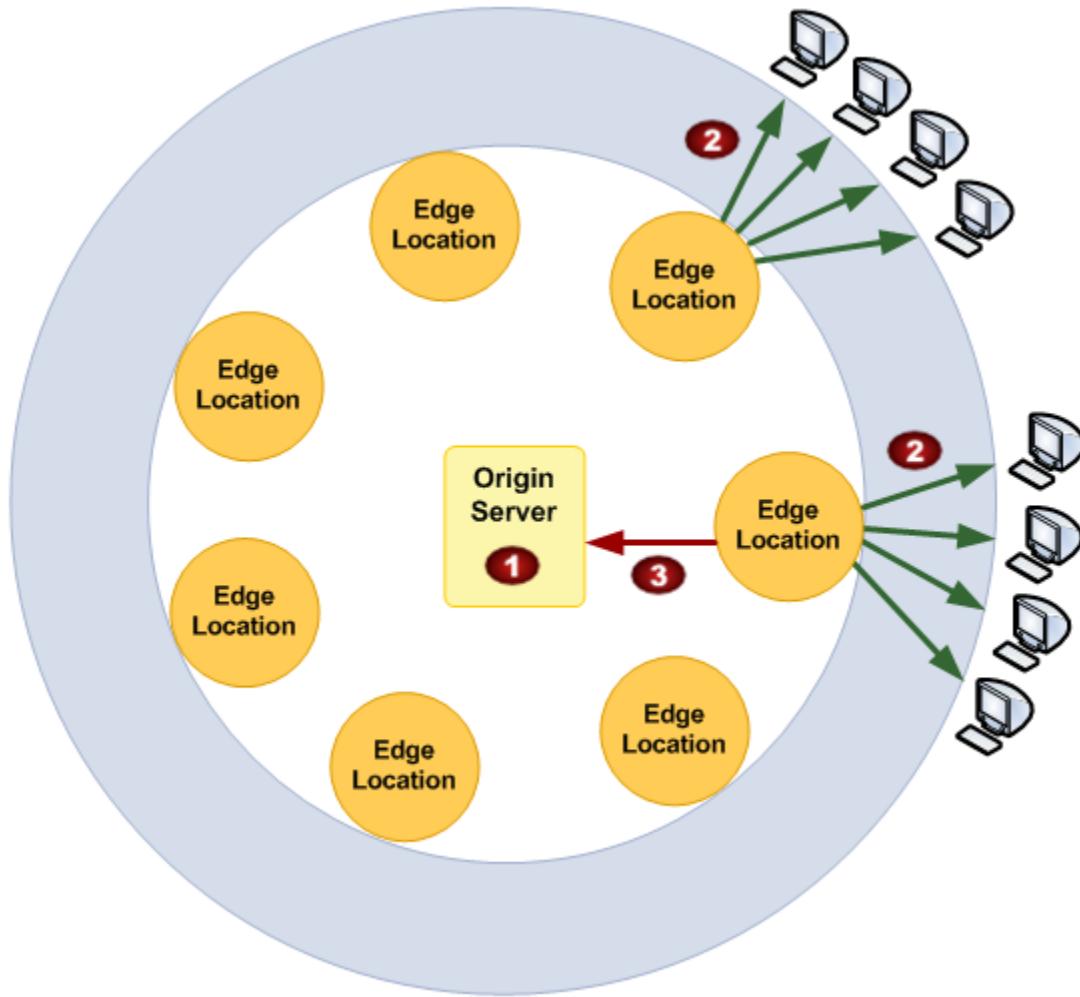
If you run PCI or HIPAA-compliant workloads, based on the [AWS Shared Responsibility Model](#), we recommend that you log your CloudFront usage data for the last 365 days for future auditing purposes. To log usage data, you can do the following:

- Enable CloudFront access logs. For more information, see [Access Logs \(p. 315\)](#).
- Capture requests that are sent to the CloudFront API. For more information, see [Using AWS CloudTrail to Capture Requests Sent to the CloudFront API \(p. 335\)](#).

AWS Billing and Usage Reports for CloudFront

Amazon CloudFront is designed so you don't have to pay any up-front fees or commit to how much content you'll have. As with the other AWS services, you pay as you go and pay only for what you use.

The following diagram and table summarize the charges to use CloudFront.



Your monthly bill from AWS allocates your usage and dollar amounts by AWS service and function. The following explains the charges that are illustrated in the previous graphic.

1. **Charge for storage in an Amazon S3 bucket.** You pay normal Amazon S3 storage charges to store objects in your bucket. The charges appear in the Amazon S3 portion of your AWS statement.
2. **Charge for serving objects from edge locations.** You incur CloudFront charges when CloudFront responds to requests for your objects. These charges are lower than the corresponding Amazon S3 charges. The CloudFront charges appear in the CloudFront portion of your AWS statement. For more information, see [Amazon CloudFront Pricing](#).

- 3. Charge for submitting data to your origin.** You incur CloudFront charges when users transfer data to your origin, which includes `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests. The CloudFront charges appear in the CloudFront portion of your AWS statement. For more information, see [Amazon CloudFront Pricing](#).

Note

You also incur a surcharge for HTTPS requests, and an additional surcharge for requests that also have field-level encryption enabled. For more information, see [Amazon CloudFront Pricing](#).

AWS provides two usage reports for CloudFront:

- The billing report is a high-level view of all of the activity for the AWS services that you're using, including CloudFront. For more information, see [AWS Billing Report for CloudFront \(p. 10\)](#).
- The usage report is a summary of activity for a specific service, aggregated by hour, day, or month. It also includes usage charts that provide a graphical representation of your CloudFront usage. For more information, see [AWS Usage Report for CloudFront \(p. 11\)](#).

AWS Billing Report for CloudFront

You can view a summary of your AWS usage and charges, listed by service, on the Bills page in the AWS Management Console.

You can also download a more detailed version of the report in CSV format. The detailed billing report includes the following values that are applicable to CloudFront:

- **ProductCode** — `AmazonCloudFront`
- **UsageType** — One of the following values
 - A code that identifies the type of data transfer
 - `Invalidations`
 - `SSL-Cert-Custom`

For more information, see [Interpreting Your AWS Bill and the AWS Usage Report for CloudFront \(p. 12\)](#).

- **ItemDescription** — A description of the billing rate for the **UsageType**.
- **Usage Start Date/Usage End Date** — The day that the usage applies to, in Coordinated Universal Time (UTC).
- **Usage Quantity** — One of the following values:
 - The number of requests during the specified time period
 - The amount of data transferred in gigabytes
 - The number of objects invalidated
 - The sum of the prorated months that you had SSL certificates associated with enabled CloudFront distributions. For example, if you have one certificate associated with an enabled distribution for an entire month and another certificate associated with an enabled distribution for half of the month, this value will be 1.5.

To display summary billing information and download the detailed billing report

1. Sign in to the AWS Management Console at <https://console.aws.amazon.com/console/home>.
2. In the title bar, click your IAM user name, and click **Billing & Cost Management**.
3. In the navigation pane, click **Bills**.
4. To view summary information for CloudFront, under **Details**, click **CloudFront**.

5. To download a detailed billing report in CSV format, click **Download CSV**, and follow the on-screen prompts to save the report.

AWS Usage Report for CloudFront

AWS provides a CloudFront usage report that is more detailed than the billing report but less detailed than CloudFront access logs. The usage report provides aggregate usage data by hour, day, or month; and it lists operations by region and usage type, such as data transferred out of the Australia region.

The CloudFront usage report includes the following values:

- **Service** — AmazonCloudFront
- **Operation** — HTTP method. Values include `DELETE`, `GET`, `HEAD`, `OPTIONS`, `PATCH`, `POST`, and `PUT`.
- **UsageType** — One of the following values
 - A code that identifies the type of data transfer
 - Invalidations
 - SSL-Cert-Custom

For more information, see [Interpreting Your AWS Bill and the AWS Usage Report for CloudFront \(p. 12\)](#).

- **Resource** — Either the ID of the CloudFront distribution associated with the usage or the certificate ID of an SSL certificate that you have associated with a CloudFront distribution.
- **StartTime/EndTime** — The day that the usage applies to, in Coordinated Universal Time (UTC).
- **UsageValue** — (1) The number of requests during the specified time period or (2) the amount of data transferred in bytes.

If you're using Amazon S3 as the origin for CloudFront, consider running the usage report for Amazon S3, too. However, if you use Amazon S3 for purposes other than as an origin for your CloudFront distributions, it might not be clear what portion applies to your CloudFront usage.

Tip

For detailed information about every request that CloudFront receives for your objects, turn on CloudFront access logs for your distribution. For more information, see [Access Logs \(p. 315\)](#).

To download the usage report for CloudFront or Amazon S3

1. Sign in to the AWS Management Console at <https://console.aws.amazon.com/console/home>.
2. In the title bar, click your IAM user name, and click **Billing & Cost Management**.
3. In the navigation pane, click **Reports**.
4. Under **AWS Usage Report**, click **AWS Usage Report**.
5. In the **Service** list, click **CloudFront** or **Amazon Simple Storage Service**.
6. Select the applicable settings:
 - **Usage Types** — For a detailed explanation of CloudFront usage types, see [the section called "Interpreting Your AWS Bill and the AWS Usage Report for CloudFront" \(p. 12\)](#).
 - For Amazon S3, select **All Usage Types**.
 - **Operation** — Select **All Operations**.
 - **Time Period** — Select the time period that you want the report to cover.
 - **Report Granularity** — Select whether you want the report to include subtotals by the hour, by the day, or by the month.

7. Click the download button for the desired format.
8. Follow the on-screen prompts to view or save the report.

Interpreting Your AWS Bill and the AWS Usage Report for CloudFront

Your AWS bill for CloudFront service includes codes and abbreviations that might not be immediately obvious. The first column in the following table lists items that appear in your bill and explains what each means.

In addition, you can get an AWS usage report for CloudFront that contains more detail than the AWS bill for CloudFront. The second column in the table lists items that appear in the usage report and shows the correlation between bill items and usage report items.

Most codes in both columns include a two-letter abbreviation that indicates the location of the activity. In the following table, *region* in a code is replaced in your AWS bill and in the usage report by one of the following two-letter abbreviations :

- **AP:** Hong Kong, Philippines, South Korea, Singapore, and Taiwan (Asia Pacific)
- **AU:** Australia
- **CA:** Canada
- **EU:** Europe
- **IN:** India
- **JP:** Japan
- **SA:** South America
- **US:** United States

For more information about pricing by region, see [Amazon CloudFront Pricing](#).

Note

This table doesn't include charges for transferring your objects from an Amazon S3 bucket to CloudFront edge locations. These charges, if any, appear in the **AWS Data Transfer** portion of your AWS bill.

| Items in Your CloudFront Bill | Values in the Usage Type Column in the CloudFront Usage Report |
|--|---|
| <p><i>region</i>-DataTransfer-Out-Bytes</p> <p>Sum of bytes that CloudFront served for web and RTMP distributions:</p> <ul style="list-style-type: none">• Web distributions: Total bytes served from CloudFront edge locations in <i>region</i> in response to user GET and HEAD requests• RTMP distributions: Total bytes transferred from CloudFront edge locations in <i>region</i> to end users | <p>Web distributions:</p> <ul style="list-style-type: none">• <i>region</i>-Out-Bytes-HTTP-Static: Bytes served via HTTP for objects with TTL ≥ 3600 seconds• <i>region</i>-Out-Bytes-HTTPS-Static: Bytes served via HTTPS for objects with TTL ≥ 3600 seconds• <i>region</i>-Out-Bytes-HTTP-Dynamic: Bytes served via HTTP for objects with TTL < 3600 seconds• <i>region</i>-Out-Bytes-HTTPS-Dynamic: Bytes served via HTTPS for objects with TTL < 3600 seconds• <i>region</i>-Out-Bytes-HTTP-Proxy: Bytes returned from CloudFront to viewers via HTTP in response to DELETE, OPTIONS, PATCH, POST, and PUT requests. |

| Items in Your CloudFront Bill | Values in the Usage Type Column in the CloudFront Usage Report |
|--|--|
| | <ul style="list-style-type: none"> • <i>region</i>-Out-Bytes-HTTPS-Proxy: Bytes returned from CloudFront to viewers via HTTPS in response to <code>DELETE</code>, <code>OPTIONS</code>, <code>PATCH</code>, <code>POST</code>, and <code>PUT</code> requests. <p>RTMP distributions:</p> <ul style="list-style-type: none"> • <i>region</i>-FMS-Out-Bytes |
| <i>region</i>-DataTransfer-Out-OBytes Web distributions only: Total bytes transferred from CloudFront edge locations to your origin in response to <code>DELETE</code> , <code>OPTIONS</code> , <code>PATCH</code> , <code>POST</code> , and <code>PUT</code> requests. | <i>region</i>-Out-OBytes-HTTP-Proxy Total bytes transferred via HTTP from CloudFront edge locations to your origin in response to <code>DELETE</code> , <code>OPTIONS</code> , <code>PATCH</code> , <code>POST</code> , and <code>PUT</code> requests. <i>region</i>-Out-OBytes-HTTPS-Proxy Total bytes transferred via HTTPS from CloudFront edge locations to your origin in response to <code>DELETE</code> , <code>OPTIONS</code> , <code>PATCH</code> , <code>POST</code> , and <code>PUT</code> requests. |
| <i>region</i>-Requests-Tier1 Web distributions only: Number of HTTP <code>GET</code> and <code>HEAD</code> requests | <i>region</i>-Requests-HTTP-Static Number of HTTP <code>GET</code> and <code>HEAD</code> requests served for objects with <code>TTL ≥ 3600</code> seconds <i>region</i>-Requests-HTTP-Dynamic Number of HTTP <code>GET</code> and <code>HEAD</code> requests served for objects with <code>TTL < 3600</code> seconds |
| <i>region</i>-Requests-Tier2-HTTPS Web distributions only: Number of HTTPS <code>GET</code> and <code>HEAD</code> requests | <i>region</i>-Requests-HTTPS-Static Number of HTTPS <code>GET</code> and <code>HEAD</code> requests served for objects with <code>TTL ≥ 3600</code> seconds <i>region</i>-Requests-HTTPS-Dynamic Number of HTTPS <code>GET</code> and <code>HEAD</code> requests served for objects with <code>TTL < 3600</code> seconds |
| <i>region</i>-Requests-HTTP-Proxy Web distributions only: Number of HTTP <code>DELETE</code> , <code>OPTIONS</code> , <code>PATCH</code> , <code>POST</code> , and <code>PUT</code> requests that CloudFront forwards to your origin | <i>region</i>-Requests-HTTP-Proxy Same as the corresponding item in your CloudFront bill |
| <i>region</i>-Requests-HTTPS-Proxy Web distributions only: Number of HTTPS <code>DELETE</code> , <code>OPTIONS</code> , <code>PATCH</code> , <code>POST</code> , and <code>PUT</code> requests that CloudFront forwards to your origin | <i>region</i>-Requests-HTTPS-Proxy Same as the corresponding item in your CloudFront bill |

| Items in Your CloudFront Bill | Values in the Usage Type Column in the CloudFront Usage Report |
|---|---|
| <i>region</i>-Requests-HTTPS-Proxy-FLE Web distributions only: Number of HTTPS DELETE, OPTIONS, PATCH, and POST requests that CloudFront forwards to your origin which were processed with field-level encryption . | <i>region</i>-Requests-HTTPS-Proxy-FLE Same as the corresponding item in your CloudFront bill |
| Invalidations Web distributions only: The charge for invalidating objects (removing the objects from CloudFront edge locations); for more information, see Paying for Object Invalidation (p. 171) . | Invalidations Same as the corresponding item in your CloudFront bill |
| SSL-Cert-Custom Web distributions only: The charge for using an SSL certificate with a CloudFront alternate domain name such as example.com instead of using the default CloudFront SSL certificate and the domain name that CloudFront assigned to your distribution. | SSL-Cert-Custom Same as the corresponding item in your CloudFront bill |

CloudFront Reports

The CloudFront console includes a variety of reports:

- [CloudFront Cache Statistics Reports \(p. 15\)](#)
- [CloudFront Popular Objects Report \(p. 15\)](#)
- [CloudFront Top Referrers Report \(p. 15\)](#)
- [CloudFront Usage Reports \(p. 15\)](#)
- [CloudFront Viewers Reports \(p. 16\)](#)

Most of these reports are based on the data in CloudFront access logs, which contain detailed information about every user request that CloudFront receives. You don't need to enable access logs to view the reports. For more information, see [Access Logs \(p. 315\)](#). The CloudFront usage report is based on the AWS usage report for CloudFront, which also doesn't require any special configuration. For more information, see [AWS Usage Report for CloudFront \(p. 11\)](#).

CloudFront Cache Statistics Reports

The CloudFront cache statistics report includes the following information:

- **Total Requests** – Shows the total number of requests for all HTTP status codes (for example, 200 or 404) and all methods (for example, GET, HEAD, or POST)
- **Percentage of Viewer Requests by Result Type** – Shows hits, misses, and errors as a percentage of total viewer requests for the selected CloudFront distribution
- **Bytes Transferred to Viewers** – Shows total bytes and bytes from misses
- **HTTP Status Codes** – Shows viewer requests by HTTP status code
- **Percentage of GET Requests that Didn't Finish Downloading** – Shows viewer GET requests that didn't finish downloading the requested object as a percentage of total requests

For more information, see [CloudFront Cache Statistics Reports \(p. 16\)](#).

CloudFront Popular Objects Report

The CloudFront popular objects report lists the 50 most popular objects and statistics about those objects, including the number of requests for the object, the number of hits and misses, the hit ratio, the number of bytes served for misses, the total bytes served, the number of incomplete downloads, and the number of requests by HTTP status code (2xx, 3xx, 4xx, and 5xx).

For more information, see [CloudFront Popular Objects Report \(p. 20\)](#).

CloudFront Top Referrers Report

The CloudFront top referrers report includes the top 25 referrers, the number of requests from a referrer, and the number of requests from a referrer as a percentage of the total number of requests during the specified period.

For more information, see [CloudFront Top Referrers Report \(p. 23\)](#).

CloudFront Usage Reports

The CloudFront usage reports include the following information:

- **Number of Requests** – Shows the number of HTTP and HTTPS requests that CloudFront responds to from edge locations in the selected region during each time interval for the specified CloudFront distribution

- **Data Transferred by Protocol** – Shows the total amount of data transferred over HTTP and HTTPS from CloudFront edge locations in the selected region during each time interval for the specified CloudFront distribution
- **Data Transferred by Destination** – Shows the total amount of data transferred over HTTP and HTTPS from CloudFront edge locations in the selected region during each time interval for the specified CloudFront distribution

For more information, see [CloudFront Usage Reports \(p. 26\)](#).

CloudFront Viewers Reports

The CloudFront viewers reports include the following information:

- **Devices** – Shows the types of devices (for example, Desktop or Mobile) that your users use to access your content
- **Browsers** – Shows the name (or the name and version) of the browsers that your users use most frequently to access your content, for example, Chrome or Firefox
- **Operating Systems** – Shows the name (or the name and version) of the operating system that viewers run on most frequently when accessing your content, for example, Linux, Mac OS X, or Windows
- **Locations** – Shows the locations, by country or by U.S. state/territory, of the viewers that access your content most frequently

For more information, see [CloudFront Viewers Reports \(p. 30\)](#).

CloudFront Cache Statistics Reports

You can use the Amazon CloudFront console to display a graphical representation of statistics related to CloudFront edge locations. Data for these statistics are drawn from the same source as CloudFront access logs. You can display charts for a specified date range in the last 60 days, with data points every hour or every day. You can usually view data about requests that CloudFront received as recently as an hour ago, but data can occasionally be delayed by as much as 24 hours.

Note

You don't need to enable access logging to view cache statistics.

To display CloudFront cache statistics

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the navigation pane, click **Cache Statistics**.
3. In the **CloudFront Cache Statistics Reports** pane, for **Start Date** and **End Date**, select the date range for which you want to display cache statistics charts. Available ranges depend on the value that you select for **Granularity**:
 - **Daily** – To display charts with one data point per day, select any date range in the previous 60 days.
 - **Hourly** – To display charts with one data point every hour, select any date range of up to 14 days within the previous 60 days.

Dates and times are in Coordinated Universal Time (UTC).

4. For **Granularity**, specify whether to display one data point per day or one data point per hour in the charts. If you specify a date range greater than 14 days, the option to specify one data point per hour is not available.

5. For **Viewer Location**, choose the continent from which viewer requests originated, or choose **All Locations**. Cache statistics charts include data for requests that CloudFront received from the specified location.
6. In the **Distribution** list, select the distributions for which you want to display data in the usage charts:
 - **An individual web distribution** – The charts display data for the selected CloudFront web distribution. The **Distribution** list displays the distribution ID and alternate domain names (CNAMEs) for the distribution, if any. If a distribution has no alternate domain names, the list includes origin domain names for the distribution.
 - **All Web Distributions** – The charts display summed data for all web distributions that are associated with the current AWS account, excluding web distributions that you have deleted.
7. Click **Update**.
8. To view data for a daily or hourly data point within a chart, move your mouse pointer over the data point.
9. For charts that show data transferred, note that you can change the vertical scale to gigabytes, megabytes, or kilobytes for each chart.

Topics

- [Downloading Data in CSV Format \(p. 17\)](#)
- [How Cache Statistics Charts Are Related to Data in the CloudFront Access Logs \(p. 19\)](#)

Downloading Data in CSV Format

You can download the Cache Statistics report in CSV format. This section explains how to download the report and describes the values in the report.

To download the Cache Statistics report in CSV format

1. While viewing the Cache Statistics report, click **CSV**.
2. In the **Opening file name** dialog box, choose whether to open or save the file.

Information About the Report

The first few rows of the report include the following information:

Version

The version of the format for this CSV file.

Report

The name of the report.

DistributionID

The ID of the distribution that you ran the report for, or **ALL** if you ran the report for all distributions.

StartDateUTC

The beginning of the date range for which you ran the report, in Coordinated Universal Time (UTC).

EndDateUTC

The end of the date range for which you ran the report, in Coordinated Universal Time (UTC).

GeneratedTimeUTC

The date and time on which you ran the report, in Coordinated Universal Time (UTC).

Granularity

Whether each row in the report represents one hour or one day.

ViewerLocation

The continent that viewer requests originated from, or `ALL`, if you chose to download the report for all locations.

Data in the Cache Statistics Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

ViewerLocation

The continent that viewer requests originated from, or `ALL`, if you chose to download the report for all locations.

TimeBucket

The hour or the day that data applies to, in Coordinated Universal Time (UTC).

RequestCount

The total number of requests for all HTTP status codes (for example, 200 or 404) and all methods (for example, GET, HEAD, or POST).

HitCount

The number of viewer requests for which the object is served from a CloudFront edge cache.

MissCount

The number of viewer requests for which the object isn't currently in an edge cache, so CloudFront must get the object from your origin.

ErrorCount

The number of viewer requests that resulted in an error, so CloudFront didn't serve the object.

IncompleteDownloadCount

The number of viewer requests for which the viewer started but didn't finish downloading the object.

HTTP2xx

The number of viewer requests for which the HTTP status code was a 2xx value (succeeded).

HTTP3xx

The number of viewer requests for which the HTTP status code was a 3xx value (additional action is required).

HTTP4xx

The number of viewer requests for which the HTTP status code was a 4xx value (client error).

HTTP5xx

The number of viewer requests for which the HTTP status code was a 5xx value (server error).

TotalBytes

The total number of bytes served to viewers by CloudFront in response to all requests for all HTTP methods.

BytesFromMisses

The number of bytes served to viewers for objects that were not in the applicable edge cache at the time of the request. This value is a good approximation of bytes transferred from your origin to CloudFront edge caches. However, it excludes requests for objects that are already in the edge cache but that have expired.

How Cache Statistics Charts Are Related to Data in the CloudFront Access Logs

The following table shows how cache statistics charts in the CloudFront console correspond with values in CloudFront access logs. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Total Requests

This chart shows the total number of requests for all HTTP status codes (for example, 200 or 404) and all methods (for example, GET, HEAD, or POST). Total requests shown in this chart equal the total number of requests in the access log files for the same time period.

Percentage of Viewer Requests by Result Type

This chart shows hits, misses, and errors as a percentage of total viewer requests for the selected CloudFront distribution:

- **Hit** – A viewer request for which the object is served from a CloudFront edge cache. In access logs, these are requests for which the value of `x-edge-response-result-type` is `Hit`.
- **Miss** – A viewer request for which the object isn't currently in an edge cache, so CloudFront must get the object from your origin. In access logs, these are requests for which the value of `x-edge-response-result-type` is `Miss`.
- **Error** – A viewer request that resulted in an error, so CloudFront didn't serve the object. In access logs, these are requests for which the value of `x-edge-response-result-type` is `Error`, `LimitExceeded`, or `CapacityExceeded`.

The chart does not include refresh hits—requests for objects that are in the edge cache but that have expired. In access logs, refresh hits are requests for which the value of `x-edge-response-result-type` is `RefreshHit`.

Bytes Transferred to Viewers

This chart shows two values:

- **Total Bytes** – The total number of bytes served to viewers by CloudFront in response to all requests for all HTTP methods. In CloudFront access logs, **Total Bytes** is the sum of the values in the `sc-bytes` column for all of the requests during the same time period.
- **Bytes from Misses** – The number of bytes served to viewers for objects that were not in the applicable edge cache at the time of the request. In CloudFront access logs, **Bytes from Misses** is the sum of the values in the `sc-bytes` column for requests for which the value of `x-edge-response-result-type` is `Miss`. This value is a good approximation of bytes transferred from your origin.

to CloudFront edge caches. However, it excludes requests for objects that are already in the edge cache but that have expired.

HTTP Status Codes

This chart shows viewer requests by HTTP status code. In CloudFront access logs, status codes appear in the `sc-status` column:

- **2xx** – The request succeeded.
- **3xx** – Additional action is required. For example, 301 (Moved Permanently) means that the requested object has moved to a different location.
- **4xx** – The client apparently made an error. For example, 404 (Not Found) means that the client requested an object that could not be found.
- **5xx** – The origin server didn't fill the request. For example, 503 (Service Unavailable) means that the origin server is currently unavailable.

Percentage of GET Requests that Didn't Finish Downloading

This chart shows viewer GET requests that didn't finish downloading the requested object as a percentage of total requests. Typically, downloading an object doesn't complete because the viewer canceled the download, for example, by clicking a different link or by closing the browser. In CloudFront access logs, these requests have a value of 200 in the `sc-status` column and a value of `Error` in the `x-edge-result-type` column.

CloudFront Popular Objects Report

The Amazon CloudFront console can display a list of the 50 most popular objects for a distribution during a specified date range in the previous 60 days.

Data for the Popular Objects report is drawn from the same source as CloudFront access logs. To get an accurate count of the top 50 objects, CloudFront counts the requests for all of your objects in 10-minute intervals beginning at midnight and keeps a running total of the top 150 objects for the next 24 hours. (CloudFront also retains daily totals for the top 150 objects for 60 days.) Near the bottom of the list, objects constantly rise onto or drop off of the list, so the totals for those objects are approximations. The fifty objects at the top of the list of 150 objects may rise and fall within the list, but they rarely drop off of the list altogether, so the totals for those objects typically are more reliable.

When an object drops off of the list of the top 150 objects and then rises onto the list again over the course of a day, CloudFront adds an estimated number of requests for the period that the object was missing from the list. The estimate is based on the number of requests received by whichever object was at the bottom of the list during that time period. If the object rises into the top 50 objects later in the day, the estimates of the number of requests that CloudFront received while the object was out of the top 150 objects usually causes the number of requests in the Popular Objects report to exceed the number of requests that appear in the access logs for that object.

Note

You don't need to enable access logging to view a list of popular objects.

To display popular objects for a distribution

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the navigation pane, click **Popular Objects**.
3. In the **CloudFront Popular Objects Report** pane, for **Start Date** and **End Date**, select the date range for which you want to display a list of popular objects. You can choose any date range in the previous 60 days.

Dates and times are in Coordinated Universal Time (UTC).

4. In the **Distribution** list, select the distribution for which you want to display a list of popular objects.
5. Click **Update**.

Topics

- [Downloading Data in CSV Format \(p. 21\)](#)
- [How Data in the Popular Objects Report Is Related to Data in the CloudFront Access Logs \(p. 22\)](#)

Downloading Data in CSV Format

You can download the Popular Objects report in CSV format. This section explains how to download the report and describes the values in the report.

To download the Popular Objects report in CSV format

1. While viewing the Popular Objects report, click **CSV**.
2. In the **Opening file name** dialog box, choose whether to open or save the file.

Information About the Report

The first few rows of the report include the following information:

Version

The version of the format for this CSV file.

Report

The name of the report.

DistributionID

The ID of the distribution that you ran the report for.

StartDateUTC

The beginning of the date range for which you ran the report, in Coordinated Universal Time (UTC).

EndDateUTC

The end of the date range for which you ran the report, in Coordinated Universal Time (UTC).

GeneratedTimeUTC

The date and time on which you ran the report, in Coordinated Universal Time (UTC).

Data in the Popular Objects Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

Object

The last 500 characters of the URL for the object.

RequestCount

The total number of requests for this object.

HitCount

The number of viewer requests for which the object is served from a CloudFront edge cache.

MissCount

The number of viewer requests for which the object isn't currently in an edge cache, so CloudFront must get the object from your origin.

HitCountPct

The value of `HitCount` as a percentage of the value of `RequestCount`.

BytesFromMisses

The number of bytes served to viewers for this object when the object was not in the applicable edge cache at the time of the request.

TotalBytes

The total number of bytes served to viewers by CloudFront for this object in response to all requests for all HTTP methods.

IncompleteDownloadCount

The number of viewer requests for this object for which the viewer started but didn't finish downloading the object.

HTTP2xx

The number of viewer requests for which the HTTP status code was a 2xx value (succeeded).

HTTP3xx

The number of viewer requests for which the HTTP status code was a 3xx value (additional action is required).

HTTP4xx

The number of viewer requests for which the HTTP status code was a 4xx value (client error).

HTTP5xx

The number of viewer requests for which the HTTP status code was a 5xx value (server error).

How Data in the Popular Objects Report Is Related to Data in the CloudFront Access Logs

The following list shows how values in the Popular Objects report in the CloudFront console correspond with values in CloudFront access logs. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

URL

The last 500 characters of the URL that viewers use to access the object.

Requests

The total number of requests for the object. This value generally corresponds closely with the number of GET requests for the object in CloudFront access logs.

Hits

The number of viewer requests for which the object was served from a CloudFront edge cache. In access logs, these are requests for which the value of `x-edge-response-result-type` is `Hit`.

Misses

The number of viewer requests for which the object wasn't in an edge cache, so CloudFront retrieved the object from your origin. In access logs, these are requests for which the value of `x-edge-response-result-type` is `Miss`.

Hit Ratio

The value of the **Hits** column as a percentage of the value of the **Requests** column.

Bytes from Misses

The number of bytes served to viewers for objects that were not in the applicable edge cache at the time of the request. In CloudFront access logs, **Bytes from Misses** is the sum of the values in the `sc-bytes` column for requests for which the value of `x-edge-result-type` is `Miss`.

Total Bytes

The total number of bytes that CloudFront served to viewers in response to all requests for the object for all HTTP methods. In CloudFront access logs, **Total Bytes** is the sum of the values in the `sc-bytes` column for all of the requests during the same time period.

Incomplete Downloads

The number of viewer requests that did not finish downloading the requested object. Typically, the reason that a download doesn't complete is that the viewer canceled it, for example, by clicking a different link or by closing the browser. In CloudFront access logs, these requests have a value of `200` in the `sc-status` column and a value of `Error` in the `x-edge-result-type` column.

2xx

The number of requests for which the HTTP status code is `2xx`, `Successful`. In CloudFront access logs, status codes appear in the `sc-status` column.

3xx

The number of requests for which the HTTP status code is `3xx`, `Redirection`. `3xx` status codes indicate that additional action is required. For example, `301` (`Moved Permanently`) means that the requested object has moved to a different location.

4xx

The number of requests for which the HTTP status code is `4xx`, `Client Error`. `4xx` status codes indicate that the client apparently made an error. For example, `404` (`Not Found`) means that the client requested an object that could not be found.

5xx

The number of requests for which the HTTP status code is `5xx`, `Server Error`. `5xx` status codes indicate that the origin server didn't fill the request. For example, `503` (`Service Unavailable`) means that the origin server is currently unavailable.

CloudFront Top Referrers Report

The CloudFront console can display a list of the 25 domains of the websites that originated the most HTTP and HTTPS requests for objects that CloudFront is distributing for a specified distribution. These top referrers can be search engines, other websites that link directly to your objects, or your own website. For example, if `http://example.com/index.html` links to 10 graphics, `example.com` is the referer for all 10 graphics. You can display the Top Referrers report for any date range in the previous 60 days.

Note

If a user enters a URL directly into the address line of a browser, there is no referrer for the requested object.

Data for the Top Referrers report is drawn from the same source as CloudFront access logs. To get an accurate count of the top 25 referrers, CloudFront counts the requests for all of your objects in 10-minute intervals and keeps a running total of the top 75 referrers. Near the bottom of the list, referrers constantly rise onto or drop off of the list, so the totals for those referrers are approximations. The 25 referrers at the top of the list of 75 referrers may rise and fall within the list, but they rarely drop off of the list altogether, so the totals for those referrers typically are more reliable.

Note

You don't need to enable access logging to view a list of top referrers.

To display top referrers for a distribution

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the navigation pane, click **Top Referrers**.
3. In the **CloudFront Top Referrers Report** pane, for **Start Date** and **End Date**, select the date range for which you want to display a list of top referrers.

Dates and times are in Coordinated Universal Time (UTC).
4. In the **Distribution** list, select the distribution for which you want to display a list of top referrers.
5. Click **Update**.

Topics

- [Downloading Data in CSV Format \(p. 24\)](#)
- [How Data in the Top Referrers Report Is Related to Data in the CloudFront Access Logs \(p. 25\)](#)

Downloading Data in CSV Format

You can download the Top Referrers report in CSV format. This section explains how to download the report and describes the values in the report.

To download the Top Referrers report in CSV format

1. While viewing the Top Referrers report, click **CSV**.
2. In the **Opening file name** dialog box, choose whether to open or save the file.

Information About the Report

The first few rows of the report include the following information:

Version

The version of the format for this CSV file.

Report

The name of the report.

DistributionID

The ID of the distribution that you ran the report for, or **ALL** if you ran the report for all distributions.

StartDateUTC

The beginning of the date range for which you ran the report, in Coordinated Universal Time (UTC).

EndDateUTC

The end of the date range for which you ran the report, in Coordinated Universal Time (UTC).

GeneratedTimeUTC

The date and time on which you ran the report, in Coordinated Universal Time (UTC).

Data in the Top Referrers Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

Referrer

The domain name of the referrer.

The total number of requests from the domain name in the `Referrer` column.

RequestsPct

The number of requests submitted by the referrer as a percentage of the total number of requests during the specified period.

How Data in the Top Referrers Report Is Related to Data in the CloudFront Access Logs

The following list shows how values in the Top Referrers report in the CloudFront console correspond with values in CloudFront access logs. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Referrer

The domain name of the referrer. In access logs, referrers are listed in the `cs(Referer)` column.

Request Count

The total number of requests from the domain name in the `Referrer` column. This value generally corresponds closely with the number of `GET` requests from the referrer in CloudFront access logs.

Request %

The number of requests submitted by the referrer as a percentage of the total number of requests during the specified period. If you have more than 25 referrers, then you can't calculate `Request %` based on the data in this table because the `Request Count` column doesn't include all of the requests during the specified period.

CloudFront Usage Reports

The Amazon CloudFront console can display a graphical representation of your CloudFront usage that is based on a subset of the usage report data. You can display charts for a specified date range in the last 60 days, with data points every hour or every day. You can usually view data about requests that CloudFront received as recently as four hours ago, but data can occasionally be delayed by as much as 24 hours.

For more information, see [How the Usage Charts Are Related to Data in the CloudFront Usage Report \(p. 29\)](#).

To display CloudFront usage charts

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In navigation pane, click **Usage Reports**.
3. In the **CloudFront Usage Reports** pane, for **Start Date** and **End Date**, select the date range for which you want to display usage charts. Available ranges depend on the value that you select for **Granularity**:
 - **Daily** — To display charts with one data point per day, select any date range in the previous 60 days.
 - **Hourly** — To display charts with one data point every hour, select any date range of up to 14 days within the previous 60 days.

Dates and times are in Coordinated Universal Time (UTC).

4. For **Granularity**, specify whether to display one data point per day or one data point per hour in the charts. If you specify a date range greater than 14 days, the option to specify one data point per hour is not available.
5. For **Billing Region**, choose the CloudFront billing region that has the data you want to view, or choose **All Regions**. Usage charts include data for requests that CloudFront processes in edge locations in the specified region. The region where CloudFront processes requests might or might not correspond with the location of your users.

Select only regions that are included in the price class for your distribution; otherwise, the usage charts probably won't contain any data. For example, if you chose Price Class 200 for your distribution, the South America and Australia billing regions are not included, so CloudFront generally won't process your requests from those regions. For more information about price classes, see [Choosing the Price Class for a CloudFront Distribution \(p. 55\)](#).

6. In the **Distribution** list, select the distributions for which you want to display data in the usage charts:
 - **An individual web distribution** — The charts display data for the selected CloudFront distribution. The **Distribution** list displays the distribution ID and alternate domain names (CNAMEs) for the distribution, if any. If a distribution has no alternate domain names, the list includes origin domain names for the distribution.
 - **All Web Distributions (excludes deleted)** — The charts display summed data for all web distributions that are associated with the current AWS account, excluding web distributions that you have deleted.
 - **All Deleted Distributions** — The charts display summed data for all web distributions that are associated with the current AWS account and that were deleted in the last 60 days.
7. Click **Update Graphs**.
8. To view data for a daily or hourly data point within a chart, move your mouse pointer over the data point.

9. For charts that show data transferred, note that you can change the vertical scale to gigabytes, megabytes, or kilobytes for each chart.

Topics

- [Downloading Data in CSV Format \(p. 27\)](#)
- [How the Usage Charts Are Related to Data in the CloudFront Usage Report \(p. 29\)](#)

Downloading Data in CSV Format

You can download the Usage report in CSV format. This section explains how to download the report and describes the values in the report.

To download the Usage report in CSV format

1. While viewing the Usage report, click **CSV**.
2. In the **Opening file name** dialog box, choose whether to open or save the file.

Information About the Report

The first few rows of the report include the following information:

Version

The version of the format for this CSV file.

Report

The name of the report.

DistributionID

The ID of the distribution that you ran the report for, `ALL` if you ran the report for all distributions, or `ALL_DELETED` if you ran the report for all deleted distributions.

StartDateUTC

The beginning of the date range for which you ran the report, in Coordinated Universal Time (UTC).

EndDateUTC

The end of the date range for which you ran the report, in Coordinated Universal Time (UTC).

GeneratedTimeUTC

The date and time on which you ran the report, in Coordinated Universal Time (UTC).

Granularity

Whether each row in the report represents one hour or one day.

BillingRegion

The continent that viewer requests originated from, or `ALL`, if you chose to download the report for all billing regions.

Data in the Usage Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, `ALL` if you ran the report for all distributions, or `ALL_DELETED` if you ran the report for all deleted distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

BillingRegion

The CloudFront billing region that you ran the report for, or `ALL`.

TimeBucket

The hour or the day that data applies to, in Coordinated Universal Time (UTC).

HTTP

The number of HTTP requests that CloudFront responded to from edge locations in the selected region during each time interval for the specified CloudFront distribution. Values include:

- The number of `GET` and `HEAD` requests, which cause CloudFront to transfer data to your users
- The number of `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests, which cause CloudFront to transfer data to your origin

HTTPS

The number of HTTPS requests that CloudFront responded to from edge locations in the selected region during each time interval for the specified CloudFront distribution. Values include:

- The number of `GET` and `HEAD` requests, which cause CloudFront to transfer data to your users
- The number of `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests, which cause CloudFront to transfer data to your origin

HTTPBytes

The total amount of data transferred over HTTP from CloudFront edge locations in the selected billing region during the time period for the specified CloudFront distribution. Values include:

- Data transferred from CloudFront to your users in response to `GET` and `HEAD` requests
- Data transferred from CloudFront to your origin for `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests
- Data transferred from CloudFront to your users in response to `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests

HTTPSSBytes

The total amount of data transferred over HTTPS from CloudFront edge locations in the selected billing region during the time period for the specified CloudFront distribution. Values include:

- Data transferred from CloudFront to your users in response to `GET` and `HEAD` requests
- Data transferred from CloudFront to your origin for `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests
- Data transferred from CloudFront to your users in response to `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests

BytesIn

The total amount of data transferred from CloudFront to your origin for `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests in the selected region during each time interval for the specified CloudFront distribution.

BytesOut

The total amount of data transferred over HTTP and HTTPS from CloudFront to your users in the selected region during each time interval for the specified CloudFront distribution. Values include:

- Data transferred from CloudFront to your users in response to `GET` and `HEAD` requests
- Data transferred from CloudFront to your users in response to `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests

How the Usage Charts Are Related to Data in the CloudFront Usage Report

The following list shows how the usage charts in the CloudFront console correspond with values in the **Usage Type** column in the CloudFront usage report.

Topics

- [Number of Requests \(p. 29\)](#)
- [Data Transferred by Protocol \(p. 29\)](#)
- [Data Transferred by Destination \(p. 30\)](#)

Number of Requests

This chart shows the number of HTTP and HTTPS requests that CloudFront responds to from edge locations in the selected region during each time interval for the specified CloudFront distribution.

Number of HTTP Requests

- ***region*-Requests-HTTP-Static:** Number of HTTP `GET` and `HEAD` requests served for objects with $\text{TTL} \geq 3600$ seconds
- ***region*-Requests-HTTP-Dynamic:** Number of HTTP `GET` and `HEAD` requests served for objects with $\text{TTL} < 3600$ seconds
- ***region*-Requests-HTTP-Proxy:** Number of HTTP `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests that CloudFront forwards to your origin

Number of HTTPS Requests

- ***region*-Requests-HTTPS-Static:** Number of HTTPS `GET` and `HEAD` requests served for objects with $\text{TTL} \geq 3600$ seconds
- ***region*-Requests-HTTPS-Dynamic:** Number of HTTPS `GET` and `HEAD` requests served for objects with $\text{TTL} < 3600$ seconds
- ***region*-Requests-HTTPS-Proxy:** Number of HTTPS `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests that CloudFront forwards to your origin

Data Transferred by Protocol

This chart shows the total amount of data transferred over HTTP and HTTPS from CloudFront edge locations in the selected region during each time interval for the specified CloudFront distribution.

Data Transferred over HTTP

- ***region*-Out-Bytes-HTTP-Static:** Bytes served via HTTP for objects with $\text{TTL} \geq 3600$ seconds
- ***region*-Out-Bytes-HTTP-Dynamic:** Bytes served via HTTP for objects with $\text{TTL} < 3600$ seconds
- ***region*-Out-Bytes-HTTP-Proxy:** Bytes returned from CloudFront to viewers via HTTP in response to `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests
- ***region*-Out-OBBytes-HTTP-Proxy:** Total bytes transferred via HTTP from CloudFront edge locations to your origin in response to `DELETE`, `OPTIONS`, `PATCH`, `POST`, and `PUT` requests

Data Transferred over HTTPS

- ***region*-Out-Bytes-HTTPS-Static:** Bytes served via HTTPS for objects with $\text{TTL} \geq 3600$ seconds

- ***region*-Out-Bytes-HTTPS-Dynamic:** Bytes served via HTTPS for objects with TTL < 3600 seconds
- ***region*-Out-Bytes-HTTPS-Proxy:** Bytes returned from CloudFront to viewers via HTTPS in response to DELETE, OPTIONS, PATCH, POST, and PUT requests
- ***region*-Out-OBytes-HTTPS-Proxy:** Total bytes transferred via HTTPS from CloudFront edge locations to your origin in response to DELETE, OPTIONS, PATCH, POST, and PUT requests

Data Transferred by Destination

This chart shows the total amount of data transferred over HTTP and HTTPS from CloudFront edge locations in the selected region during each time interval for the specified CloudFront distribution.

Data Transferred from CloudFront to Your Users

- ***region*-Out-Bytes-HTTP-Static:** Bytes served via HTTP for objects with TTL ≥ 3600 seconds
- ***region*-Out-Bytes-HTTPS-Static:** Bytes served via HTTPS for objects with TTL ≥ 3600 seconds
- ***region*-Out-Bytes-HTTP-Dynamic:** Bytes served via HTTP for objects with TTL < 3600 seconds
- ***region*-Out-Bytes-HTTPS-Dynamic:** Bytes served via HTTPS for objects with TTL < 3600 seconds
- ***region*-Out-Bytes-HTTP-Proxy:** Bytes returned from CloudFront to viewers via HTTP in response to DELETE, OPTIONS, PATCH, POST, and PUT requests
- ***region*-Out-Bytes-HTTPS-Proxy:** Bytes returned from CloudFront to viewers via HTTPS in response to DELETE, OPTIONS, PATCH, POST, and PUT requests

Data Transferred from CloudFront to Your Origin

- ***region*-Out-OBytes-HTTP-Proxy:** Total bytes transferred via HTTP from CloudFront edge locations to your origin in response to DELETE, OPTIONS, PATCH, POST, and PUT requests
- ***region*-Out-OBytes-HTTPS-Proxy:** Total bytes transferred via HTTPS from CloudFront edge locations to your origin in response to DELETE, OPTIONS, PATCH, POST, and PUT requests

CloudFront Viewers Reports

The CloudFront console can display four reports about the physical devices (desktop computers, mobile devices) and about the viewers (typically web browsers) that are accessing your content:

- **Devices** – The type of the devices that your users use most frequently to access your content, for example, Desktop or Mobile.
- **Browsers** – The name (or the name and version) of the browsers that your users use most frequently to access your content, for example, Chrome or Firefox. The report lists the top 10 browsers.
- **Operating Systems** – The name (or the name and version) of the operating system that viewers run on most frequently when accessing your content, for example, Linux, Mac OS X, or Windows. The report lists the top 10 operating systems.
- **Locations** – The locations, by country or by U.S. state/territory, of the viewers that access your content most frequently. The report lists the top 50 countries or U.S. states/territories.

You can display all four Viewers reports for any date range in the previous 60 days. For the Locations report, you can also display the report with data points every hour for any date range of up to 14 days in the previous 60 days.

Note

You don't need to enable access logging to view Viewers charts and reports.

Topics

- [Displaying Viewers Charts and Reports \(p. 31\)](#)

- [Downloading Data in CSV Format \(p. 31\)](#)
- [How Data in the Locations Report Is Related to Data in the CloudFront Access Logs \(p. 37\)](#)

Displaying Viewers Charts and Reports

To display CloudFront Viewers charts and reports, perform the following procedure.

To display CloudFront Viewers charts and reports

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the navigation pane, click **Viewers**.
3. In the **CloudFront Viewers** pane, for **Start Date** and **End Date**, select the date range for which you want to display viewer charts and reports.

For the Locations chart, available ranges depend on the value that you select for **Granularity**:

- **Daily** – To display charts with one data point per day, select any date range in the previous 60 days.
- **Hourly** – To display charts with one data point every hour, select any date range of up to 14 days within the previous 60 days.

Dates and times are in Coordinated Universal Time (UTC).

4. (Browsers and Operating Systems charts only) For **Grouping**, specify whether you want to group browsers and operating systems by name (Chrome, Firefox) or by name and version (Chrome 40.0, Firefox 35.0).
5. (Locations chart only) For **Granularity**, specify whether to display one data point per day or one data point per hour in the charts. If you specify a date range greater than 14 days, the option to specify one data point per hour is not available.
6. (Locations chart only) For **Details**, specify whether to display the top locations by countries or by U.S. states.
7. In the **Distribution** list, select the distribution for which you want to display data in the usage charts:
 - **An individual web distribution** – The charts display data for the selected CloudFront web distribution. The **Distribution** list displays the distribution ID and an alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.
 - **All Web Distributions (excludes deleted)** – The charts display summed data for all web distributions that are associated with the current AWS account, excluding web distributions that you have deleted.
8. Click **Update**.
9. To view data for a daily or hourly data point within a chart, move your mouse pointer over the data point.

Downloading Data in CSV Format

You can download each of the Viewer reports in CSV format. This section explains how to download the reports and describes the values in the report.

To download the Viewer reports in CSV format

1. While viewing the applicable Viewer report, click **CSV**.

2. Choose the data that you want to download, for example, **Devices** or **Devices Trends**.
3. In the **Opening file name** dialog box, choose whether to open or save the file.

Topics

- [Information About the Reports \(p. 32\)](#)
- [Devices Report \(p. 32\)](#)
- [Device Trends Report \(p. 33\)](#)
- [Browsers Report \(p. 34\)](#)
- [Browser Trends Report \(p. 34\)](#)
- [Operating Systems Report \(p. 35\)](#)
- [Operating System Trends Report \(p. 35\)](#)
- [Locations Report \(p. 36\)](#)
- [Location Trends Report \(p. 36\)](#)

Information About the Reports

The first few rows of each report includes the following information:

Version

The version of the format for this CSV file.

Report

The name of the report.

DistributionID

The ID of the distribution that you ran the report for, or **ALL** if you ran the report for all web distributions.

StartDateUTC

The beginning of the date range for which you ran the report, in Coordinated Universal Time (UTC).

EndDateUTC

The end of the date range for which you ran the report, in Coordinated Universal Time (UTC).

GeneratedTimeUTC

The date and time on which you ran the report, in Coordinated Universal Time (UTC).

Grouping (Browsers and Operating Systems Reports Only)

Whether the data is grouped by the name or by the name and version of the browser or operating system.

Granularity

Whether each row in the report represents one hour or one day.

Details (Locations Report Only)

Whether requests are listed by country or by U.S. state.

Devices Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

Requests

The number of requests that CloudFront received from each type of device.

RequestsPct

The number of requests that CloudFront received from each type of device as a percentage of the total number of requests that CloudFront received from all devices.

Device Trends Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

TimeBucket

The hour or the day that the data applies to, in Coordinated Universal Time (UTC).

Desktop

The number of requests that CloudFront received from desktop computers during the period.

Mobile

The number of requests that CloudFront received from mobile devices during the period. Mobile devices can include both tablets and mobile phones. If CloudFront can't determine whether a request originated from a mobile device or a tablet, it's counted in the `Mobile` column.

Smart-TV

The number of requests that CloudFront received from smart TVs during the period.

Tablet

The number of requests that CloudFront received from tablets during the period. If CloudFront can't determine whether a request originated from a mobile device or a tablet, it's counted in the `Mobile` column.

Unknown

Requests for which the value of the `User-Agent` HTTP header was not associated with one of the standard device types, for example, `Desktop` or `Mobile`.

Empty

The number of requests that CloudFront received that didn't include a value in the HTTP `User-Agent` header during the period.

Browsers Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

Group

The browser or the browser and version that CloudFront received requests from, depending on the value of `Grouping`. In addition to browser names, possible values include the following:

- **Bot/Crawler** – primarily requests from search engines that are indexing your content.
- **Empty** – requests for which the value of the `User-Agent` HTTP header was empty.
- **Other** – browsers that CloudFront identified but that aren't among the most popular. If `Bot/Crawler`, `Empty`, and/or `Unknown` don't appear among the first nine values, then they're also included in `Other`.
- **Unknown** – requests for which the value of the `User-Agent` HTTP header was not associated with a standard browser. Most requests in this category come from custom applications or scripts.

Requests

The number of requests that CloudFront received from each type of browser.

RequestsPct

The number of requests that CloudFront received from each type of browser as a percentage of the total number of requests that CloudFront received during the time period.

Browser Trends Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

TimeBucket

The hour or the day that the data applies to, in Coordinated Universal Time (UTC).

(Browsers)

The remaining columns in the report list the browsers or the browsers and their versions, depending on the value of `Grouping`. In addition to browser names, possible values include the following:

- **Bot/Crawler** – primarily requests from search engines that are indexing your content.
- **Empty** – requests for which the value of the `User-Agent` HTTP header was empty.
- **Other** – browsers that CloudFront identified but that aren't among the most popular. If `Bot/Crawler`, `Empty`, and/or `Unknown` don't appear among the first nine values, then they're also included in `Other`.

- **Unknown** – requests for which the value of the User-Agent HTTP header was not associated with a standard browser. Most requests in this category come from custom applications or scripts.

Operating Systems Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

Group

The operating system or the operating system and version that CloudFront received requests from, depending on the value of `Grouping`. In addition to operating system names, possible values include the following:

- **Bot/Crawler** – primarily requests from search engines that are indexing your content.
- **Empty** – requests for which the value of the User-Agent HTTP header was empty.
- **Other** – operating systems that CloudFront identified but that aren't among the most popular. If `Bot/Crawler`, `Empty`, and/or `Unknown` don't appear among the first nine values, then they're also included in `Other`.
- **Unknown** – requests for which the value of the User-Agent HTTP header was not associated with a standard browser. Most requests in this category come from custom applications or scripts.

Requests

The number of requests that CloudFront received from each type of operating system.

RequestsPct

The number of requests that CloudFront received from each type of operating system as a percentage of the total number of requests that CloudFront received during the time period.

Operating System Trends Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

TimeBucket

The hour or the day that the data applies to, in Coordinated Universal Time (UTC).

(Operating systems)

The remaining columns in the report list the operating systems or the operating systems and their versions, depending on the value of `Grouping`. In addition to operating system names, possible values include the following:

- **Bot/Crawler** – primarily requests from search engines that are indexing your content.
- **Empty** – requests for which the value of the User-Agent HTTP header was empty.
- **Other** – operating systems that CloudFront identified but that aren't among the most popular. If Bot/Crawler, Empty, and/or Unknown don't appear among the first nine values, then they're also included in Other.
- **Unknown** – requests for which the operating system isn't specified in the User-Agent HTTP header.

Locations Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

LocationCode

The abbreviation for the location that CloudFront received requests from. For more information about possible values, see the description of Location in [How Data in the Locations Report Is Related to Data in the CloudFront Access Logs \(p. 37\)](#).

LocationName

The name of the location that CloudFront received requests from.

Requests

The number of requests that CloudFront received from each location.

RequestsPct

The number of requests that CloudFront received from each location as a percentage of the total number of requests that CloudFront received from all locations during the time period.

TotalBytes

The number of bytes that CloudFront served to viewers in this country or state, for the specified distribution and period.

Location Trends Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for, or `ALL` if you ran the report for all distributions.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

TimeBucket

The hour or the day that the data applies to, in Coordinated Universal Time (UTC).

(Locations)

The remaining columns in the report list the locations that CloudFront received requests from. For more information about possible values, see the description of Location in [How Data in the Locations Report Is Related to Data in the CloudFront Access Logs \(p. 37\)](#).

How Data in the Locations Report Is Related to Data in the CloudFront Access Logs

The following list shows how data in the Locations report in the CloudFront console corresponds with values in CloudFront access logs. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Location

The country or U.S. state that the viewer is in. In access logs, the `c-ip` column contains the IP address of the device that the viewer is running on. We use geolocation data to identify the geographic location of the device based on the IP address.

If you're displaying the **Locations** report by country, note that the country list is based on [ISO 3166-2, Codes for the representation of names of countries and their subdivisions – Part 2: Country subdivision code](#). The country list includes the following additional values:

- **Anonymous Proxy** – The request originated from an anonymous proxy.
- **Satellite Provider** – The request originated from a satellite provider that provides internet service to multiple countries. Users might be in countries with a high risk of fraud.
- **Europe (Unknown)** – The request originated from an IP in a block that is used by multiple European countries. The country that the request originated from cannot be determined. CloudFront uses **Europe (Unknown)** as the default.
- **Asia/Pacific (Unknown)** – The request originated from an IP in a block that is used by multiple countries in the Asia/Pacific region. The country that the request originated from cannot be determined. CloudFront uses **Asia/Pacific (Unknown)** as the default.

If you're displaying the **Locations** report by U.S. state, note that the report can include U.S. territories and U.S. Armed Forces regions.

Request Count

The total number of requests from the country or U.S. state that the viewer is in, for the specified distribution and period. This value generally corresponds closely with the number of GET requests from IP addresses in that country or state in CloudFront access logs.

Request %

One of the following, depending on the value that you selected for **Details**:

- **Countries** – The requests from this country as a percentage of the total number of requests.
- **U.S. States** – The requests from this state as a percentage of the total number of requests from the United States.

If requests came from more than 50 countries, then you can't calculate **Request %** based on the data in this table because the **Request Count** column doesn't include all of the requests during the specified period.

Bytes

The number of bytes that CloudFront served to viewers in this country or state, for the specified distribution and period. To change the display of data in this column to KB, MB, or GB, click the link in the column heading.

Getting Started with CloudFront

The example in this topic gives you a quick overview of how to use CloudFront to:

- Store the original versions of your objects in one Amazon Simple Storage Service (Amazon S3) bucket.
- Distribute download content such as text or graphics.
- Make your objects accessible to everyone.
- Use the CloudFront domain name in URLs for your objects (for example, `http://d11111abcdef8.cloudfront.net/image.jpg`) instead of your own domain name (for example, `http://www.example.com/image.jpg`).
- Keep your objects in CloudFront edge locations for the default duration of 24 hours. (The minimum duration is 0 seconds.)

For information about how to use CloudFront when you want to use other options, see [Task List for Creating a Web Distribution \(p. 59\)](#) or [Task List for Streaming Media Files Using RTMP \(p. 132\)](#).

You only need to perform a few basic steps to start delivering your content using CloudFront. The first step is signing up. After that, you create a CloudFront distribution, and then use the CloudFront domain name to reference content in your web pages or applications.

Topics

- [Step 1: Sign up for Amazon Web Services \(p. 38\)](#)
- [Step 2: Upload your content to Amazon S3 and grant object permissions \(p. 38\)](#)
- [Step 3: Create a CloudFront Web Distribution \(p. 40\)](#)
- [Step 4: Test your links \(p. 46\)](#)

Step 1: Sign up for Amazon Web Services

If you haven't already done so, sign up for Amazon Web Services at <http://aws.amazon.com>. Just choose **Sign Up Now** and enter any required information.

Step 2: Upload your content to Amazon S3 and grant object permissions

An Amazon S3 bucket is a container that can contain objects or folders. CloudFront can distribute almost any type of object for you using an Amazon S3 bucket as the source, for example, text, images, and videos. You can create multiple buckets, and there is no limit to the amount of data that you can store on Amazon S3.

By default, your Amazon S3 bucket and all of the objects in it are private—only the AWS account that created the bucket has permission to read or write the objects in it. If you want to allow anyone to access the objects in your Amazon S3 bucket using CloudFront URLs, you must grant public read permissions to

the objects. (This is one of the most common mistakes when working with CloudFront and Amazon S3. You must explicitly grant privileges to each object in an Amazon S3 bucket.)

Note

If you want to restrict who can download your content, you can use the CloudFront private content feature. For more information about distributing private content, see [Serving Private Content through CloudFront \(p. 207\)](#).

To upload your content to Amazon S3 and grant read permission to everyone

1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. In the Amazon S3 console, choose **Create Bucket**.
3. In the **Create Bucket** dialog, enter a bucket name.

Important

For your bucket to work with CloudFront, the name must conform to DNS naming requirements. For more information, go to [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

4. Select a region for your bucket. By default, Amazon S3 creates buckets in the US East (N. Virginia) region. We recommend that you choose a region close to you to optimize latency, minimize costs, or to address regulatory requirements.
5. Choose **Create**.
6. Select your bucket in the **Buckets** pane, and choose **Upload**.
7. On the **Upload - Select Files** page, choose **Add Files**, and choose the files that you want to upload.



8. Enable public read privileges for each object that you upload to your Amazon S3 bucket.

- a. Choose **Set Details**.

- b. On the **Set Details** page, choose **Set Permissions**.
- c. On the **Set Permissions** page, choose **Make everything public**.
9. Choose **Start Upload**.

After the upload completes, you can navigate to this item by its URL. In the case of the previous example, the URL would be:

`http://s3.amazonaws.com/example-myawsbucket/filename`

Use your Amazon S3 URL to verify that your content is publicly accessible, but remember that this is not the URL you will use when you are ready to distribute your content.

Step 3: Create a CloudFront Web Distribution

To create a CloudFront web distribution

1. Open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose **Create Distribution**.
3. On the **Select a delivery method for your content** page, in the **Web** section, choose **Get Started**.

The screenshot shows the 'Select a delivery method for your content' page. It has two main sections: 'Web' and 'RTMP'. The 'Web' section is active, indicated by a blue bar at the top. It contains a 'Get Started' button, which is circled in red. The 'RTMP' section is also visible below it.

Step 1: Select delivery method

Step 2: Create distribution

Select a delivery method for your content

Web

Create a web distribution if you want to:

- Speed up distribution of static and dynamic content, for example, .html, .css, and .js files.
- Distribute media files using HTTP or HTTPS.
- Add, update, or delete objects, and submit data from web forms.
- Use live streaming to stream an event in real time.

You store your files in an origin — either an Amazon S3 bucket or a web server. If you have multiple origins in your distribution, you can add more origins to the distribution.

Get Started

RTMP

Create an RTMP distribution to speed up distribution of your streaming media files. An RTMP distribution uses the Real-time Messaging Protocol (RTMP) to deliver video and audio content to end users. An RTMP distribution uses the same infrastructure as a CloudFront edge location to speed up distribution of your streaming media files. You can use an RTMP distribution to deliver content to a CloudFront edge location using the RTMP protocol. An RTMP distribution allows an end user to begin playing a video file before it has finished downloading from a CloudFront edge location. Note the following:

- To create an RTMP distribution, you must store the media files in an Amazon S3 bucket.
- To use CloudFront live streaming, create a web distribution.

Get Started

4. On the **Create Distribution** page, under **Origin Settings**, choose the Amazon S3 bucket that you created earlier. For **Origin ID**, **Origin Path**, **Restrict Bucket Access**, and **Origin Custom Headers**, accept the default values.

Create Distribution

Origin Settings

| | | |
|------------------------|--|-------------------------------|
| Origin Domain Name | myawsbucket100.com.s3.amazonaws.co | i |
| Origin Path | | i |
| Origin ID | S3-myawsbucket100.com | i |
| Restrict Bucket Access | <input type="radio"/> Yes <input checked="" type="radio"/> No | i |
| Origin Custom Headers | Header Name <input type="text"/> | Value <input type="text"/> |

5. Under **Default Cache Behavior Settings**, accept the default values, and CloudFront will:

- Forward all requests that use the CloudFront URL for your distribution (for example, `http://d111111abcdef8.cloudfront.net/image.jpg`) to the Amazon S3 bucket that you specified in Step 4.
- Allow end users to use either HTTP or HTTPS to access your objects.
- Respond to requests for your objects.
- Cache your objects at CloudFront edge locations for 24 hours.
- Forward only the default request headers to your origin and not cache your objects based on the values in the headers.
- Exclude cookies and query string parameters, if any, when forwarding requests for objects to your origin. (Amazon S3 doesn't process cookies and processes only a limited set of query string parameters.)
- Not be configured to distribute media files in the Microsoft Smooth Streaming format.
- Allow everyone to view your content.
- Not automatically compress your content.

For more information about cache behavior options, see [Cache Behavior Settings \(p. 70\)](#).

Default Cache Behavior Settings

| | |
|---|--|
| Path Pattern | Default (*) |
| Viewer Protocol Policy | <input checked="" type="radio"/> HTTP and HTTPS <input type="radio"/> Redirect HTTP to HTTPS <input type="radio"/> HTTPS Only |
| Allowed HTTP Methods | <input checked="" type="radio"/> GET, HEAD <input type="radio"/> GET, HEAD, OPTIONS <input type="radio"/> GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE |
| Cached HTTP Methods | GET, HEAD (Cached by default) |
| Forward Headers | None (Improves Caching) ▼ |
| Object Caching | <input checked="" type="radio"/> Use Origin Cache Headers <input type="radio"/> Customize Learn More |
| Minimum TTL | 0 |
| Maximum TTL | 31536000 |
| Default TTL | 86400 |
| Forward Cookies | None (Improves Caching) ▼ |
| Forward Query Strings | <input type="radio"/> Yes <input checked="" type="radio"/> No (Improves Caching) |
| Smooth Streaming | <input type="radio"/> Yes <input checked="" type="radio"/> No |
| Restrict Viewer Access (Use Signed URLs or Signed Cookies) | <input type="radio"/> Yes <input checked="" type="radio"/> No |
| Compress Objects Automatically | <input type="radio"/> Yes <input checked="" type="radio"/> No Learn More |

6. Under **Distribution Settings**, enter the applicable values:

Price Class

Select the price class that corresponds with the maximum price that you want to pay for CloudFront service. By default, CloudFront serves your objects from edge locations in all CloudFront regions.

For more information about price classes and about how your choice of price class affects CloudFront performance for your distribution, go to [Choosing the Price Class for a CloudFront Distribution \(p. 55\)](#). For information about CloudFront pricing, including how price classes map to CloudFront regions, go to [Amazon CloudFront Pricing](#).

AWS WAF Web ACL

If you want to use AWS WAF to allow or block HTTP and HTTPS requests based on criteria that you specify, choose the web ACL to associate with this distribution. For more information about AWS WAF, see the [AWS WAF Developer Guide](#).

Alternate Domain Names (CNAMEs) (Optional)

Specify one or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution. For example, if you want the URL for the object:

/images/image.jpg

to look like this:

http://www.example.com/images/image.jpg

instead of like this:

http://d111111abcdef8.cloudfront.net/images/image.jpg

you would create a CNAME for www.example.com.

Important

If you add a CNAME for www.example.com to your distribution, you also need to create (or update) a CNAME record with your DNS service to route queries for www.example.com to d111111abcdef8.cloudfront.net. You must have permission to create a CNAME record with the DNS service provider for the domain. Typically, this means that you own the domain, but you may also be developing an application for the domain owner. For more information about CNAMEs, see [Using Alternate Domain Names \(CNAMEs\) \(p. 50\)](#).

For the current limit on the number of alternate domain names that you can add to a distribution or request a higher limit, see [General Limits on Web Distributions \(p. 359\)](#).

SSL Certificate

Accept the default value, **Default CloudFront Certificate**.

Default Root Object (Optional)

The object that you want CloudFront to request from your origin (for example, index.html) when a viewer requests the root URL of your distribution (<http://www.example.com/>) instead of an object in your distribution (<http://www.example.com/product-description.html>). Specifying a default root object avoids exposing the contents of your distribution.

Logging (Optional)

If you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket, select **On**, and specify the bucket and an optional prefix for the names of the log files. There is no extra charge to enable logging, but you accrue the usual Amazon S3 charges for storing and accessing the files. CloudFront doesn't delete the logs automatically, but you can delete them at any time.

Cookie Logging

In this example, we're using Amazon S3 as the origin for your objects, and Amazon S3 doesn't process cookies, so we recommend that you select **Off** for the value of **Cookie Logging**.

Comment (Optional)

Enter any comments that you want to save with the distribution.

Distribution State

Select **Enabled** if you want CloudFront to begin processing requests as soon as the distribution is created, or select **Disabled** if you do not want CloudFront to begin processing requests after the distribution is created.

Distribution Settings

Price Class

Use All Edge Locations (Best Performance) 

AWS WAF Web ACL

None 

Alternate Domain Names (CNAMEs)**SSL Certificate**

Default CloudFront Certificate (*.cloudfront.net)

Choose this option if you want your users to use HTTPS or HTTP with the CloudFront domain name (such as https://d1111111abcd1/logo.jpg).

Important: If you choose this option, CloudFront requires that browsers support TLSv1 or later to access your content.

Custom SSL Certificate (stored in AWS IAM): 

No certificate

Choose this option if you want your users to use HTTPS to access alternate domain name (such as https://www.example.com/logo) using dedicated CloudFront IP addresses or SNI.

To choose this option, you first need to upload your certificate to the store (the -path parameter must start with /cloudfront/).

[Learn More](#)

Default Root Object**Logging**

On
 Off

Bucket for Logs**Log Prefix****Cookie Logging**

On
 Off

Comment**Distribution Status**

API Version 2016-09-29 

45

Enabled
 Disabled

[Cancel](#)

[Back](#)

[Create](#)

7. Choose **Create Distribution**.
8. After CloudFront has created your distribution, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. If you chose to enable the distribution, it will then be ready to process requests. This should take less than 15 minutes.

The domain name that CloudFront assigns to your distribution appears in the list of distributions. (It also appears on the **General** tab for a selected distribution.)

Step 4: Test your links

After you've created your distribution, CloudFront knows where your Amazon S3 origin server is, and you know the domain name associated with the distribution. You can create a link to your Amazon S3 bucket content with that domain name, and have CloudFront serve it.

Note

You must wait until the status of your distribution changes to **Deployed** before testing your links.

To link to your objects

1. Copy the following HTML into a new file:
 - Replace <domain name> with the domain name that CloudFront assigned to your distribution.
 - Replace <object name> with the name of a file in your Amazon S3 bucket.

```
<html>
<head>My CloudFront Test</head>
<body>
<p>My text content goes here.</p>
<p>
</body>
</html>
```

For example, if your domain name was d111111abcdef8.cloudfront.net and your object was image.jpg, the URL for the link would be:

<http://d111111abcdef8.cloudfront.net/image.jpg>.

If your object is in a folder within your bucket, include the folder in the URL. For example, if image.jpg is located in an images folder, then the URL would be:

<http://d111111abcdef8.cloudfront.net/images/image.jpg>

2. Save the text in a file that has a .html filename extension.
3. Open your web page in a browser to ensure that you can see your content. If you cannot see the content, confirm that you have performed all of the steps correctly. You can also see the tips in [Troubleshooting \(p. 343\)](#).

The browser returns your page with the embedded image file, served from the edge location that CloudFront determined was appropriate to serve the object.

For more information on using CloudFront, go to [Amazon CloudFront Resources \(p. 364\)](#).

Working with Distributions

Topics

- [Overview of Web and RTMP Distributions \(p. 48\)](#)
- [Creating Web and RTMP Distributions \(p. 49\)](#)
- [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#)
- [Deleting a Distribution \(p. 50\)](#)
- [Using Alternate Domain Names \(CNAMEs\) \(p. 50\)](#)
- [Choosing the Price Class for a CloudFront Distribution \(p. 55\)](#)
- [Using CloudFront with Amazon S3 \(p. 55\)](#)
- [Changes to the CloudFront API \(p. 57\)](#)

The following table lists the actions you can perform on a distribution and provides links to the corresponding documentation on how to perform the actions using the CloudFront console and the CloudFront API.

| Action | Using the CloudFront Console | Using the CloudFront API: Web Distributions | Using the CloudFront API: RTMP Distributions |
|--|---|---|---|
| Create a distribution | Web Distributions: See Task List for Creating a Web Distribution (p. 59) RTMP Distributions: See Task List for Streaming Media Files Using RTMP (p. 132) | Go to POST Distribution | Go to POST Streaming Distribution |
| List your distributions | See Listing, Viewing, and Updating CloudFront Distributions (p. 49) | Go to GET Distribution List | Go to GET Streaming Distribution List |
| Get all information about a distribution | See Listing, Viewing, and Updating CloudFront Distributions (p. 49) | Go to GET Distribution | Go to GET Streaming Distribution |
| Get the distribution configuration | See Listing, Viewing, and Updating CloudFront Distributions (p. 49) | Go to GET Distribution Config | Go to GET Streaming Distribution Config |
| Update a distribution | See Listing, Viewing, and Updating CloudFront Distributions (p. 49) | Go to PUT Distribution Config | Go to PUT Streaming Distribution Config |

| Action | Using the CloudFront Console | Using the CloudFront API: Web Distributions | Using the CloudFront API: RTMP Distributions |
|-----------------------|---|---|---|
| Delete a distribution | See Deleting a Distribution (p. 50) | Go to DELETE Distribution | Go to DELETE Streaming Distribution |

Overview of Web and RTMP Distributions

When you want to use CloudFront to distribute your content, you create a distribution and specify configuration settings such as:

- Your origin, which is the Amazon S3 bucket or HTTP server from which CloudFront gets the files that it distributes. You can specify any combination of up to 25 Amazon S3 buckets and/or HTTP servers as your origins.
- Whether you want the files to be available to everyone or you want to restrict access to selected users.
- Whether you want CloudFront to require users to use HTTPS to access your content.
- Whether you want CloudFront to forward cookies and/or query strings to your origin.
- Whether you want CloudFront to prevent users in selected countries from accessing your content.
- Whether you want CloudFront to create access logs.

For the current limit on the number of distributions that you can create for each AWS account, see [General Limits on Web Distributions \(p. 359\)](#) and [Limits on RTMP Distributions \(p. 363\)](#).

The number of files that you can serve per distribution is unlimited.

Web Distributions

You can use web distributions to serve the following content over HTTP or HTTPS:

- Static and dynamic download content, for example, .html, .css, .js, and image files, using HTTP or HTTPS.
- Multimedia content on demand using progressive download and Apple HTTP Live Streaming (HLS). For more information, see the applicable topic in [Working with Web Distributions \(p. 59\)](#).

You can't serve Adobe Flash multimedia content over HTTP or HTTPS, but you can serve it using a CloudFront RTMP distribution. See [RTMP Distributions \(p. 48\)](#) below.

- A live event, such as a meeting, conference, or concert, in real time. For live streaming, you create the distribution automatically by using an AWS CloudFormation stack. For more information, see the applicable live-streaming tutorial in [CloudFront Streaming Tutorials \(p. 351\)](#).

For web distributions, your origin can be either an Amazon S3 bucket or an HTTP server, for example, a web server. For more information about how web distributions work, including the values that you specify when you create a web distribution, see [Working with Web Distributions \(p. 59\)](#). For information about creating a web distribution, see [Task List for Creating a Web Distribution \(p. 59\)](#).

RTMP Distributions

RTMP distributions stream media files using Adobe Media Server and the Adobe Real-Time Messaging Protocol (RTMP). An RTMP distribution must use an Amazon S3 bucket as the origin.

For information about the values you specify when you create an RTMP distribution, see [Working with RTMP Distributions \(p. 131\)](#). For information about creating an RTMP distribution, see [Task List for Streaming Media Files Using RTMP \(p. 132\)](#).

Creating Web and RTMP Distributions

Web distributions: For information about creating web distributions using the CloudFront console, see [Task List for Creating a Web Distribution \(p. 59\)](#). For information about creating web distributions using the CloudFront API, see [POST Distribution](#) in the *Amazon CloudFront API Reference*.

RTMP distributions: For information about creating RTMP distributions using the CloudFront console, see [Task List for Streaming Media Files Using RTMP \(p. 132\)](#). For information about creating RTMP distributions using the CloudFront API, see [POST Streaming Distribution](#) in the *Amazon CloudFront API Reference*.

List, Viewing, and Updating CloudFront Distributions

You can use the CloudFront console to list the CloudFront distributions that are associated with your AWS account, view the settings for a distribution, and update most settings.

When you save changes to your distribution configuration, CloudFront starts to propagate the changes to all edge locations. Until your configuration is updated in an edge location, CloudFront continues to serve your content from that location based on the previous configuration. After your configuration is updated in an edge location, CloudFront immediately starts to serve your content from that location based on the new configuration.

Your changes don't propagate to every edge location instantaneously. When propagation is complete, the status of your distribution changes from **InProgress** to **Deployed**. While CloudFront is propagating your changes to edge locations, we cannot determine whether a given edge location is serving your content based on the previous configuration or the new configuration.

To List, View, and Update CloudFront Distributions Using the CloudFront Console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the ID of the distribution that you want to view or update.

Note

The top pane lists all of the distributions that are associated with the AWS account that you used when you signed in to the CloudFront console.

3. To view or edit RTMP distribution settings, skip to Step 4.

To view or edit settings for a web distribution, perform the following steps.

- a. In the **Distribution Settings** pane, click the tab for the settings that you want to change: **General**, **Origins**, or **Behaviors**.
- b. For general settings, click **Edit**.
For origins or cache behaviors, click the origin or cache behavior, and click **Edit**.
- c. Enter or update the applicable values. For information about the fields, see the following topics:
 - **General settings:** [Distribution Details \(p. 77\)](#)
 - **Origin settings:** [Origin Settings \(p. 65\)](#)

- **Cache behavior settings:** [Cache Behavior Settings \(p. 70\)](#)
- d. Click **Yes, Edit**.
- 4. To edit or view settings for an RTMP distribution:
 - a. In the **Distribution Details** pane, click **Edit**.
 - b. Enter or update the applicable values. For information about the fields, see [Values that You Specify When You Create or Update an RTMP Distribution \(p. 134\)](#).
 - c. Click **Yes, Edit**.

Deleting a Distribution

If you no longer want to use a distribution, use the following procedure to delete it using the CloudFront console.

You can also delete a distribution using the CloudFront API:

- To delete a web distribution, use the `DELETE Distribution` API action. For more information, go to [DELETE Distribution](#) in the *Amazon CloudFront API Reference*.
- To delete an RTMP distribution, use the `DELETE Streaming Distribution` API action. For more information, go to [DELETE Streaming Distribution](#) in the *Amazon CloudFront API Reference*.

Note

CloudFront lets you create a total of up to 200 web distributions and 100 RTMP distributions for an AWS account.

To Delete a CloudFront Distribution Using the CloudFront Console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the right pane of the CloudFront console, find the distribution that you want to delete.
3. If the value of the **State** column is **Disabled**, skip to Step 7.

If the value of **State** is **Enabled** and the value of **Status** is **Deployed**, continue with Step 4 to disable the distribution before deleting it.

If the value of **State** is **Enabled** and the value of **Status** is **InProgress**, wait until **Status** changes to **Deployed**. Then continue with Step 4 to disable the distribution before deleting it.

4. In the right pane of the CloudFront console, check the check box for the distribution that you want to delete.
5. Click **Disabled** to disable the distribution, and click **Yes, Disable** to confirm. Then click **Close**.
6. The value of the **State** column immediately changes to **Disabled**. Wait until the value of the **Status** column changes to **Deployed**.
7. Check the check box for the distribution that you want to delete.
8. Click **Delete**, and click **Yes, Delete** to confirm. Then click **Close**.

Using Alternate Domain Names (CNAMEs)

In CloudFront, an alternate domain name, also known as a CNAME, lets you use your own domain name (for example, `www.example.com`) for links to your objects instead of using the domain name that CloudFront assigns to your distribution. Both web and RTMP distributions support alternate domain names.

When you create a distribution, CloudFront returns a domain name for the distribution, for example:

d111111abcdef8.cloudfront.net

When you use the CloudFront domain name for your objects, the URL for an object called /images/image.jpg is:

http://d111111abcdef8.cloudfront.net/images/image.jpg

If you want to use your own domain name, such as www.example.com, instead of the cloudfront.net domain name that CloudFront assigned to your distribution, you can add an alternate domain name to your distribution for www.example.com. You can then use the following URL for /images/image.jpg:

http://www.example.com/images/image.jpg

Topics

- [Using the * Wildcard in Alternate Domain Names \(p. 51\)](#)
- [Restrictions on Using Alternate Domain Names \(p. 51\)](#)
- [Adding an Alternate Domain Name \(p. 52\)](#)

Using the * Wildcard in Alternate Domain Names

When you add alternate domain names, you can use the * wildcard at the beginning of a domain name instead of specifying subdomains individually. For example, with an alternate domain name of *.example.com, you can use any domain name that ends with example.com in your object URLs, such as www.example.com, product-name.example.com, and marketing.product-name.example.com. The name of an object is the same regardless of the domain name, for example:

www.example.com/images/image.jpg

product-name.example.com/images/image.jpg

marketing.product-name.example.com/images/image.jpg

The alternate domain name must begin with an asterisk and a dot (*.). You *cannot* use a wildcard to replace part of a subdomain name, like this: *domain.example.com, and you cannot replace a subdomain in the middle of a domain name, like this: subdomain.*.example.com.

A wildcard alternate domain name, such as *.example.com, can include another alternate domain name, such as example.com, as long as they're both in the same CloudFront distribution or they're in distributions that were created by using the same AWS account.

Restrictions on Using Alternate Domain Names

Note the following restrictions on using alternate domain names:

CNAMEs Must be Lowercase

All CNAMEs must be lowercase to be valid.

Maximum Number of Alternate Domain Names

For the current limit on the number of alternate domain names that you can add to a distribution or to request a higher limit, see [General Limits on Web Distributions \(p. 359\)](#).

Permission to Change DNS Configuration

If you're adding alternate domain names to your distribution, you need to create CNAME records to route DNS queries for the domain names to your CloudFront distribution. You must have permission

to create CNAME records with the DNS service provider for the corresponding domains. Typically, this means that you own the domains, but you may also be developing an application for the domain owner.

Duplicate and Overlapping Alternate Domain Names

You cannot add an alternate domain name to a CloudFront distribution if the alternate domain name already exists in another CloudFront distribution, even if your AWS account owns the other distribution.

However, you can add a wildcard alternate domain name, such as `*.example.com`, that includes (that overlaps with) a non-wildcard alternate domain name, such as `www.example.com`. Overlapping domain names can be in the same distribution or in separate distributions as long as both distributions were created by using the same AWS account.

Alternate Domain Names at the Zone Apex for a Domain

When you add an alternate domain name to a distribution, you need to create a CNAME record in your DNS configuration to route DNS queries for the domain name to your CloudFront distribution. However, you can't create a CNAME record for the top node of a DNS namespace, also known as the zone apex; the DNS protocol doesn't allow it. For example, if you register the DNS name `example.com`, the zone apex is `example.com`. You can't create a CNAME record for `example.com`, but you can create CNAME records for `www.example.com`, `newproduct.example.com`, and so on.

If you're using Route 53 as your DNS service, you can create an alias resource record set, which has two advantages over CNAME records. You can create an alias resource record set for a domain name at the zone apex (`example.com`). In addition, with an alias resource record set, you don't pay for Route 53 queries.

Note

If you enable IPv6, you need to create two alias resource record sets, one to route IPv4 traffic (an A record) and one to route IPv6 traffic (an AAAA record). For more information, see [Enable IPv6 \(p. 81\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

For more information, see [Routing Queries to an Amazon CloudFront Distribution](#) in the *Amazon Route 53 Developer Guide*.

Alternate Domain Names and HTTPS

If you want viewers to use HTTPS with an alternate domain names, additional configuration is required. For more information, see [Using Alternate Domain Names and HTTPS \(p. 283\)](#).

Adding an Alternate Domain Name

The following task list describes the process for using the CloudFront console to add an alternate domain name to your distribution so you can use your own domain name in your links instead of the CloudFront domain name that is associated with your distribution.

Note

If you want viewers to use HTTPS with your alternate domain name, see [Using Alternate Domain Names and HTTPS \(p. 283\)](#).

For information about updating your distribution using the CloudFront API, see [Working with Distributions \(p. 47\)](#).

Adding an Alternate Domain Name Using the CloudFront Console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.

2. Choose the ID for the distribution that you want to update.
3. On the **General** tab, choose **Edit**.
4. Update the following values:

Alternate Domain Names (CNAMEs)

Add the applicable alternate domain names. Separate domain names with commas, or type each domain name on a new line.

SSL Certificate (Web Distributions Only)

Choose the applicable option, depending on whether you want viewers to use HTTPS to access your content:

- **Don't use HTTPS** – Choose **Default CloudFront Certificate**.
- **Use HTTPS** – Choose **Custom SSL Certificate**, and choose a certificate from the list. The list can include certificates provisioned by AWS Certificate Manager (ACM), certificates that you purchased from another certificate authority and uploaded to ACM, and certificates that you purchased from another certificate authority and uploaded to the IAM certificate store.

If you uploaded a certificate to the IAM certificate store but it doesn't appear in the list, review the procedure [Importing an SSL/TLS Certificate \(p. 290\)](#) to confirm that you correctly uploaded the certificate.

If you choose this setting, we recommend that you use only an alternate domain name in your object URLs (`https://example.com/logo.jpg`). If you use your CloudFront distribution domain name (`https://d111111abcdef8.cloudfront.net/logo.jpg`) and the viewer supports SNI, then CloudFront behaves normally. However, a viewer that does not support SNI exhibits one of the following behaviors, depending on the value of **Clients Supported**:

- **All Clients**: If the viewer doesn't support SNI, it displays a warning because the CloudFront domain name doesn't match the domain name in your SSL/TLS certificate.
- **Only Clients that Support Server Name Indication (SNI)**: CloudFront drops the connection with the viewer without returning the object.

Clients Supported (Web Distributions Only)

Choose the applicable options:

- **All Clients**: CloudFront serves your HTTPS content using dedicated IP addresses. If you select this option, you incur additional charges when you associate your SSL/TLS certificate with a distribution that is enabled. For more information, see <http://aws.amazon.com/cloudfront/pricing>.
- **Only Clients that Support Server Name Indication (SNI)**: Older browsers or other clients that don't support SNI must use another method to access your content.

For more information, see [Choosing How CloudFront Serves HTTPS Requests \(p. 283\)](#).

5. Choose **Yes, Edit**.
6. On the **General** tab for the distribution, confirm that **Distribution Status** has changed to **Deployed**. If you try to use an alternate domain name before the updates to your distribution have been deployed, the links that you create in the following steps might not work.
7. Configure the DNS service for the domain to route traffic for the domain, such as `example.com`, to the CloudFront domain name for your distribution, such as `d111111abcdef8.cloudfront.net`. The method that you use depends on whether you're using Route 53 as the DNS service provider for the domain:

Route 53

Create an alias resource record set. With an alias resource record set, you don't pay for Route 53 queries. In addition, you can create an alias resource record set for the root domain name

(example.com), which DNS doesn't allow for CNAMEs. For more information, see [Routing Queries to an Amazon CloudFront Distribution](#) in the *Amazon Route 53 Developer Guide*.

Another DNS service provider

Use the method provided by your DNS service provider to add a CNAME resource record set to the hosted zone for your domain. This new CNAME resource record set will redirect DNS queries from your domain (for example, www.example.com) to the CloudFront domain name for your distribution (for example, d111111abcdef8.cloudfront.net). For more information, see the documentation provided by your DNS service provider.

Important

If you already have an existing CNAME record for your domain name, update that resource record set or replace it with a new one that points to the CloudFront domain name for your distribution.

In addition, confirm that your CNAME resource record set points to your distribution's domain name and not to one of your origin servers.

8. Using dig or a similar tool, confirm that the resource record set that you created in step 7 points to the domain name for your distribution. For more information about dig, go to <http://www.kloth.net/services/dig.php>.

The following example shows a dig request on the images.example.com domain, as well as the relevant part of the response.

```
[prompt]--> dig images.example.com

; <>> DIG 9.3.3rc2 <>> images.example.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15917
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;images.example.com.      IN      A

;; ANSWER SECTION:
. 10800 IN CNAME d111111abcdef8.cloudfront.net.
...
...
```

The line in the Answer Section shows a CNAME resource record set that routes queries for images.example.com to the CloudFront distribution domain name d111111abcdef8.cloudfront.net. The CNAME resource record set is configured correctly if the name on the right side of CNAME is the domain name for your CloudFront distribution. If that is any other value, for example, the domain name for your Amazon S3 bucket, then the CNAME resource record set is configured incorrectly. In that case, go back to Step 4 and correct the CNAME record to point to the domain name for your distribution.

9. Test the alternate domain name by creating some test links that use your domain name in the URL instead of the CloudFront domain name for your distribution.
10. In your application, change the links for your objects to use your alternate domain name instead of the domain name of your CloudFront distribution.

Choosing the Price Class for a CloudFront Distribution

CloudFront has edge locations all over the world. Our cost for each edge location varies and, as a result, the price that we charge you varies depending on the edge location from which CloudFront serves your requests.

CloudFront edge locations are grouped into geographic regions, and we've grouped regions into price classes. The default price class includes all regions. Another price class includes most regions (the United States; Europe; Hong Kong, Korea, and Singapore; Japan; and India regions) but excludes the most-expensive regions. A third price class includes only the least-expensive regions (the United States and Europe regions).

By default, CloudFront responds to requests for your objects based only on performance: objects are served from the edge location for which latency is lowest for that viewer. If you're willing to accept higher latency for your viewers in some geographic regions in return for lower cost, you can choose a price class that doesn't include all CloudFront regions. Although CloudFront will serve your objects only from the edge locations in that price class, it still serves content from the edge location that has the lowest latency among the edge locations in your selected price class. However, some of your viewers, especially those in geographic regions that are not in your price class, may see higher latency than if your content were being served from all CloudFront edge locations. For example, if you choose the price class that includes only the United States and Europe, viewers in Australia and in Asia may experience higher latency than if you choose the price class that includes Australia and Asia.

If you choose a price class that does not include all edge locations, CloudFront may still occasionally serve requests for your content from an edge location in a region that is not included in your price class. When this happens, you are not charged the rate for the more expensive region from which your objects were served. Instead, you're charged the rate for the least-expensive region in your selected price class.

You can choose a price class when you create or update a CloudFront web distribution or RTMP distribution. To find the applicable topic about creating or updating a web or an RTMP distribution using the CloudFront console or API, see [Working with Distributions \(p. 47\)](#).

If you're creating or updating a distribution by using the CloudFront API, one of the AWS SDKs, or AWS CloudFormation, see the applicable topic for a list of valid values (search for `PriceClass`):

- **Web distributions** – [DistributionConfig Complex Type](#)
- **RTMP distributions** – [StreamingDistributionConfig Complex Type](#)

For more information about CloudFront pricing and price classes, go to [Amazon CloudFront Pricing](#).

Using CloudFront with Amazon S3

You can store your content in an Amazon S3 bucket and use CloudFront to distribute the content. This topic explains how to use CloudFront with your S3 bucket, and how to update your CloudFront distribution if you move the S3 bucket to a different region.

Topics

- [Adding CloudFront When You're Distributing Content from Amazon S3 \(p. 56\)](#)
- [Moving an Amazon S3 Bucket to a Different Region \(p. 57\)](#)

Adding CloudFront When You're Distributing Content from Amazon S3

If you store your objects in an Amazon S3 bucket, you can either have your users get your objects directly from S3, or you can configure CloudFront to get your objects from S3 and distribute them to your users.

Using CloudFront can be more cost effective if your users access your objects frequently because, at higher usage, the price for CloudFront data transfer is lower than the price for Amazon S3 data transfer. In addition, downloads are faster with CloudFront than with Amazon S3 alone because your objects are stored closer to your users.

Note

If you want CloudFront to respect Amazon S3 cross-origin resource sharing settings, configure CloudFront to forward the Origin header to Amazon S3. For more information, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

If you currently distribute content directly from your Amazon S3 bucket using your own domain name (such as example.com) instead of the domain name of your Amazon S3 bucket (such as MyAWSBucket.s3.amazonaws.com), you can add CloudFront with no disruption by using the following procedure.

To add CloudFront when you're already distributing your content from Amazon S3

1. Create a CloudFront distribution using the procedure described in the applicable topic:
 - [Task List for Creating a Web Distribution \(p. 59\)](#)
 - [Task List for Streaming Media Files Using RTMP \(p. 132\)](#)

When you create the distribution, specify the name of your Amazon S3 bucket as the origin server.

Important

For your bucket to work with CloudFront, the name must conform to DNS naming requirements. For more information, see [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

If you're using a CNAME with Amazon S3, specify the CNAME for your distribution, too.

2. Create a test web page that contains links to publicly readable objects in your Amazon S3 bucket, and test the links. For this initial test, use the CloudFront domain name of your distribution in the object URLs, for example, `http://d111111abcdef8.cloudfront.net/images/image.jpg`.

For more information about the format of CloudFront URLs, see [Format of URLs for Objects \(p. 143\)](#).

3. If you're using Amazon S3 CNAMEs, your application uses your domain name (for example, example.com) to reference the objects in your Amazon S3 bucket instead of using the name of your bucket (for example, myawsbucket.s3.amazonaws.com). To continue using your domain name to reference objects instead of using the CloudFront domain name for your distribution (for example, d111111abcdef8.cloudfront.net), you need to update your settings with your DNS service provider.

For Amazon S3 CNAMEs to work, your DNS service provider must have a CNAME resource record set for your domain that currently routes queries for the domain to your Amazon S3 bucket. For example, if a user requests this object:

`http://example.com/images/image.jpg`

the request is automatically rerouted, and the user sees this object:

`http://myawsbucket.s3.amazonaws.com/images/image.jpg`

To route queries to your CloudFront distribution instead of your Amazon S3 bucket, you need to use the method provided by your DNS service provider to update the CNAME resource record set for your domain. This updated CNAME record will start to redirect DNS queries from your domain to the CloudFront domain name for your distribution. For more information, see the documentation provided by your DNS service provider.

Note

If you're using Route 53 as your DNS service, you can use either a CNAME resource record set or an alias resource record set. For information about editing resource record sets, see [Editing Resource Record Sets](#). For information about alias resource record sets, see [Choosing Between Alias and Non-Alias Resource Record Sets](#). Both topics are in the *Amazon Route 53 Developer Guide*.

For more information about using CNAMEs with CloudFront, see [Using Alternate Domain Names \(CNAMEs\) \(p. 50\)](#).

After you update the CNAME resource record set, it can take up to 72 hours for the change to propagate throughout the DNS system, although it usually happens faster. During this time, some requests for your content will continue to be routed to your Amazon S3 bucket, and others will be routed to CloudFront.

Moving an Amazon S3 Bucket to a Different Region

If you're using Amazon S3 as the origin for a CloudFront distribution and you move the bucket to a different region, CloudFront can take up to an hour to update its records to include the change of region when both of the following are true:

- You're using a CloudFront origin access identity (OAI) to restrict access to the bucket
- You move the bucket to an Amazon S3 region that requires Signature Version 4 for authentication

When you're using OAIs, CloudFront uses the region (among other values) to calculate the signature that it uses to request objects from your bucket. For more information about OAIs, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#). For a list of Amazon S3 regions and the signature versions that they support, see [Amazon Simple Storage Service \(Amazon S3\)](#) in the "Regions and Endpoints" chapter of the *Amazon Web Services General Reference*.

To force a faster update to CloudFront's records, you can update your CloudFront distribution, for example, by updating the **Comment** field on the **General** tab in the CloudFront console. When you update a distribution, CloudFront immediately checks on the region that your bucket is in; propagation of the change to all edge locations should take less than 15 minutes.

Changes to the CloudFront API

Beginning with the 2012-05-05 version of the CloudFront API, we made substantial changes to the format of the XML document that you include in the request body when you create or update a web distribution or an RTMP distribution, and when you invalidate objects. With previous versions of the API, we discovered that it was too easy to accidentally delete one or more values for an element that accepts multiple values, for example, CNAMEs and trusted signers. Our changes for the 2012-05-05 release are intended to prevent these accidental deletions and to notify you when there's a mismatch between the number of values you say you're specifying in the `Quantity` element and the number of values you're actually specifying.

Note the following about using the 2012-05-05 API version or later with web and RTMP distributions that were created using earlier API versions:

- You cannot use versions of the API earlier than 2012-05-05 to update a web distribution that was created or updated using the 2012-05-05 or later CloudFront API.
- You can use the new API version to get a list of distributions, get information about a distribution, or get distribution configuration. CloudFront returns an XML document in the new XML format.
- To update a distribution that was created using an earlier API version, use the 2012-05-05 or later version of GET Distribution or GET Streaming Distribution to get an XML document in the new XML format, change the data as applicable, and use the 2012-05-05 or later version of PUT Distribution Config or PUT Streaming Distribution Config to submit the changes to CloudFront.
- You can use the new API to delete a distribution that was created using an earlier API version. The distribution must already be disabled.

Working with Web Distributions

This section describes how you configure and manage CloudFront web distributions. For a basic explanation of distributions, see [Working with Distributions \(p. 47\)](#). For information about CloudFront RTMP distributions, see [Working with RTMP Distributions \(p. 131\)](#).

Topics

- [Task List for Creating a Web Distribution \(p. 59\)](#)
- [Creating or Updating a Web Distribution Using the CloudFront Console \(p. 60\)](#)
- [Testing Your Web Distribution \(p. 61\)](#)
- [Using Amazon S3 Origins and Custom Origins for Web Distributions \(p. 62\)](#)
- [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#)
- [Values that CloudFront Displays in the Console When You Create or Update a Web Distribution \(p. 84\)](#)
- [Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins \(p. 85\)](#)
- [Using Field-Level Encryption to Help Protect Sensitive Data \(p. 85\)](#)
- [Using AWS WAF to Control Access to Your Content \(p. 93\)](#)
- [Restricting the Geographic Distribution of Your Content \(p. 94\)](#)
- [Configuring On-Demand Smooth Streaming \(p. 96\)](#)
- [Configuring On-Demand Progressive Downloads \(p. 97\)](#)
- [Configuring On-Demand Apple HTTP Live Streaming \(HLS\) \(p. 98\)](#)

Task List for Creating a Web Distribution

The following task list summarizes the process for creating a web distribution.

To Create a Web Distribution

1. Create one or more Amazon S3 buckets or configure HTTP servers as your origin servers. An origin is the location where you store the original version of your web content. When CloudFront gets a request for your files, it goes to the origin to get the files that it distributes at edge locations. You can use any combination of Amazon S3 buckets and HTTP servers as your origin servers.

If you're using Amazon S3, note that the name of your bucket must be all lowercase and cannot contain spaces.

If you're using an Amazon EC2 server or another custom origin, review [Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins \(p. 85\)](#).

For the current limit on the number of origins that you can create for a distribution or to request a higher limit, see [General Limits on Web Distributions \(p. 359\)](#).

2. Upload your content to your origin servers. If you don't want to restrict access to your content using CloudFront signed URLs, make the objects publicly readable.

Important

You are responsible for ensuring the security of your origin server. You must ensure that CloudFront has permission to access the server and that the security settings are appropriate to safeguard your content.

3. Create your CloudFront web distribution:
 - For more information about creating a web distribution using the CloudFront console, see [Creating or Updating a Web Distribution Using the CloudFront Console \(p. 60\)](#).
 - For information about creating a web distribution using the CloudFront API, go to [POST Distribution](#) in the [Amazon CloudFront API Reference](#).
4. Optional: If you created your distribution using the CloudFront console, create more cache behaviors or origins for your distribution. For more information, see [To List, View, and Update CloudFront Distributions Using the CloudFront Console \(p. 49\)](#).
5. Test your web distribution. For more information, see [Testing Your Web Distribution \(p. 61\)](#).
6. Develop your website or application to access your content using the domain name that CloudFront returned after you created your distribution in Step 3. For example, if CloudFront returns d111111abcdef8.cloudfront.net as the domain name for your distribution, the URL for the file image.jpg in an Amazon S3 bucket or in the root directory on an HTTP server will be `http://d111111abcdef8.cloudfront.net/image.jpg`.

If you specified one or more alternate domain names (CNAMEs) when you created your distribution, you can use your own domain name. In that case, the URL for image.jpg might be `http://www.example.com/image.jpg`.

Note the following:

- If you want to use signed URLs to restrict access to your content, see [Serving Private Content through CloudFront \(p. 207\)](#).
- If you want to serve compressed content, see [Serving Compressed Files \(p. 179\)](#).
- For information about CloudFront request and response behavior for Amazon S3 and custom origins, see [Request and Response Behavior \(p. 184\)](#).

Creating or Updating a Web Distribution Using the CloudFront Console

The procedures in this topic explain how to create or update a web distribution using the CloudFront console. If you want to create a web distribution using the CloudFront API, go to [POST Distribution](#) in the [Amazon CloudFront API Reference](#). If you want to update a web distribution using the CloudFront API, go to [PUT DistributionConfig](#) in the [Amazon CloudFront API Reference](#).

For the current limit on the number of web distributions that you can create for each AWS account or to request a higher limit, [General Limits on Web Distributions \(p. 359\)](#).

- To create a CloudFront web distribution using the CloudFront console (p. 60)
- To update a CloudFront web distribution using the CloudFront console (p. 61)

To create a CloudFront web distribution using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose **Create Distribution**.
3. On the first page of the **Create Distribution Wizard**, in the **Web** section, choose **Get Started**.
4. Specify settings for the distribution. For more information, see [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).
5. Save changes.

6. After CloudFront creates your distribution, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. If you chose to enable the distribution, it will then be ready to process requests.

The domain name that CloudFront assigns to your distribution appears in the list of distributions. (It also appears on the **General** tab for a selected distribution.)
7. When your distribution is deployed, confirm that you can access your content using your new CloudFront URL or CNAME. For more information, see [Testing Your Web Distribution \(p. 61\)](#).

To update a CloudFront web distribution using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the ID for the distribution that you want to update.
3. Update the applicable values. For more information, see [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).
4. Choose **Create Distribution**.
5. After you change settings, the value of the **Status** column for your distribution changes to **InProgress** while CloudFront propagates the changes to edge locations. When **Status** changes to **Deployed**, the distribution is ready to process requests. (The value of the **State** column for the distribution must also be **Enabled**.)

Testing Your Web Distribution

After you've created your distribution, CloudFront knows where your origin server is, and you know the domain name associated with the distribution. You can create links to your objects using the CloudFront domain name, and CloudFront will serve the objects to your web page or application.

Note

You must wait until the status of the distribution changes to **Deployed** before you can test your links.

To create links to objects in a web distribution

1. Copy the following HTML code into a new file, replace `domain-name` with your distribution's domain name, and replace `object-name` with the name of your object.

```
<html>
<head>My CloudFront Test</head>
<body>
<p>My text content goes here.</p>
<p>
</body>
</html>
```

For example, if your domain name were `d111111abcdef8.cloudfront.net` and your object were `image.jpg`, the URL for the link would be:

`http://d111111abcdef8.cloudfront.net/image.jpg`.

If your object is in a folder on your origin server, then the folder must also be included in the URL. For example, if `image.jpg` were located in the `images` folder on your origin server, then the URL would be:

`http://d111111abcdef8.cloudfront.net/images/image.jpg`

2. Save the HTML code in a file that has a .html filename extension.
3. Open your web page in a browser to ensure that you can see your object.

The browser returns your page with the embedded image file, served from the edge location that CloudFront determined was appropriate to serve the object.

Using Amazon S3 Origins and Custom Origins for Web Distributions

When you create a web distribution, you specify where CloudFront sends requests for the files that it distributes to edge locations. CloudFront supports using Amazon S3 buckets and HTTP servers (for example, web servers) as origins.

Using Amazon S3 Buckets for Your Origin

When you use Amazon S3 as an origin for your distribution, you place any objects that you want CloudFront to deliver in an Amazon S3 bucket. You can use any method that is supported by Amazon S3 to get your objects into Amazon S3, for example, the Amazon S3 console or API, or a third-party tool. You can create a hierarchy in your bucket to store the objects, just as you would with any other Amazon S3 bucket.

Using an existing Amazon S3 bucket as your CloudFront origin server doesn't change the bucket in any way; you can still use it as you normally would to store and access Amazon S3 objects at the standard Amazon S3 price. You incur regular Amazon S3 charges for storing the objects in the bucket. For more information about the charges to use CloudFront, see [CloudFront Reports \(p. 15\)](#).

Important

For your bucket to work with CloudFront, the name must conform to DNS naming requirements. For more information, go to [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

When you specify the Amazon S3 bucket that you want CloudFront to get objects from, how you specify the bucket name depends on whether you have configured the bucket as a website endpoint:

The bucket is not configured as a website endpoint

In general, use the following format:

`bucket-name.s3.amazonaws.com`

If your bucket is in the US Standard region and you want Amazon S3 to route requests to a facility in Northern Virginia, use the following format:

`bucket-name.s3-external-1.amazonaws.com`

When you specify the bucket name in this format, you can use the following CloudFront features:

- Configure CloudFront to communicate with your Amazon S3 bucket using SSL. For more information, see [Using HTTPS with CloudFront \(p. 278\)](#).
- Use an origin access identity to require that your users access your content using CloudFront URLs, not by using Amazon S3 URLs. For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).
- Update the content of your bucket by submitting POST and PUT requests to CloudFront. For more information, see [HTTP Methods \(p. 186\)](#) in the topic [How CloudFront Processes and Forwards Requests to Your Amazon S3 Origin Server \(p. 184\)](#).

The bucket is configured as a website endpoint

Enter the Amazon S3 static website hosting endpoint for your bucket. This value appears in the Amazon S3 console, on the **Properties** page under **Static Website Hosting**.

When you specify the bucket name in this format, you can use Amazon S3 redirects and Amazon S3 custom error documents. (CloudFront also provides custom error pages. For more information, see [Customizing Error Responses \(p. 172\)](#).) For more information about Amazon S3 features, see the [Amazon S3 documentation](#).

Do not specify the bucket using the following formats:

- The Amazon S3 path style, `s3.amazonaws.com/bucket-name`
- The Amazon S3 CNAME, if any

Using Amazon EC2 or Other Custom Origins

A custom origin is an HTTP server, for example, a web server. The HTTP server can be an Amazon EC2 instance or an HTTP server that you manage privately. When you use a custom origin, you specify the DNS name of the server, along with the HTTP and HTTPS ports and the protocol that you want CloudFront to use when fetching objects from your origin.

Most CloudFront features are supported when you use a custom origin with the following exceptions:

- **RTMP distributions**—Not supported.
- **Private content**—Although you can use a signed URL to distribute content from a custom origin, for CloudFront to access the custom origin, the origin must remain publicly accessible. For more information, see [Serving Private Content through CloudFront \(p. 207\)](#).

For information about requirements and recommendations when using custom origins, see [Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins \(p. 85\)](#).

Values That You Specify When You Create or Update a Web Distribution

When you create a new web distribution or update an existing distribution, you specify the following values. For information about creating or updating a web distribution using the CloudFront console, see the applicable topic:

- [Working with Web Distributions \(p. 59\)](#)
- [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#)

Delivery Method (p. 65)

Origin Settings (p. 65)

- [Origin Domain Name \(p. 66\)](#)
- [Origin Path \(p. 67\)](#)
- [Origin ID \(p. 67\)](#)

- [Restrict Bucket Access \(Amazon S3 Only\) \(p. 67\)](#)
- [Origin Access Identity \(Amazon S3 Only\) \(p. 67\)](#)
- [Comment for New Identity \(Amazon S3 Only\) \(p. 68\)](#)
- [Your Identities \(Amazon S3 Only\) \(p. 68\)](#)
- [Grant Read Permissions on Bucket \(Amazon S3 Only\) \(p. 68\)](#)
- [Origin SSL Protocols \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 68\)](#)
- [Origin Protocol Policy \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 68\)](#)
- [Origin Response Timeout \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 69\)](#)
- [Origin Keep-alive Timeout \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 69\)](#)
- [HTTP Port \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 69\)](#)
- [HTTPS Port \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 70\)](#)
- [Origin Custom Headers \(p. 70\)](#)

Cache Behavior Settings (p. 70)

- [Path Pattern \(p. 71\)](#)
- [Origin \(Existing Distributions Only\) \(p. 72\)](#)
- [Viewer Protocol Policy \(p. 72\)](#)
- [Allowed HTTP Methods \(p. 73\)](#)
- [Field Level Encryption \(p. 72\)](#)
- [Cached HTTP Methods \(p. 73\)](#)
- [Cache Based on Selected Request Headers \(p. 73\)](#)
- [Whitelist Headers \(p. 74\)](#)
- [Object Caching \(p. 74\)](#)
- [Minimum TTL \(p. 74\)](#)
- [Maximum TTL \(p. 74\)](#)
- [Default TTL \(p. 75\)](#)
- [Forward Cookies \(Amazon EC2 and Other Custom Origins Only\) \(p. 75\)](#)
- [Whitelist Cookies \(Amazon EC2 and Other Custom Origins Only\) \(p. 75\)](#)
- [Query String Forwarding and Caching \(p. 76\)](#)
- [Query String Whitelist \(p. 76\)](#)
- [Smooth Streaming \(p. 76\)](#)
- [Restrict Viewer Access \(Use Signed URLs\) \(p. 76\)](#)
- [Trusted Signers \(p. 76\)](#)
- [AWS Account Numbers \(p. 77\)](#)
- [Compress Objects Automatically \(p. 77\)](#)
- [Event Type \(p. 77\)](#)
- [Lambda Function ARN \(p. 77\)](#)

Distribution Details (p. 77)

- [Price Class \(p. 78\)](#)

- [AWS WAF Web ACL \(p. 78\)](#)
- [Alternate Domain Names \(CNAMEs\) \(p. 78\)](#)
- [SSL Certificate \(p. 78\)](#)
- [Clients Supported \(p. 79\)](#)
- [Security Policy \(p. 79\)](#)
- Minimum SSL Security Protocol – See [Security Policy \(p. 79\)](#)
- [Supported HTTP Versions \(p. 80\)](#)
- [Default Root Object \(p. 80\)](#)
- [Logging \(p. 80\)](#)
- [Bucket for Logs \(p. 80\)](#)
- [Log Prefix \(p. 81\)](#)
- [Cookie Logging \(p. 81\)](#)
- [Enable IPv6 \(p. 81\)](#)
- [Comment \(p. 82\)](#)
- [Distribution State \(p. 82\)](#)

Custom Error Pages and Error Caching (p. 82)

- [Error Code \(p. 82\)](#)
- [Response Page Path \(p. 83\)](#)
- [Response Code \(p. 83\)](#)
- [Error Caching Minimum TTL \(p. 83\)](#)

Restrictions (p. 83)

- [Enable Geo Restriction \(p. 83\)](#)
- [Restriction Type \(p. 83\)](#)
- [Countries \(p. 83\)](#)

Delivery Method

You specify the delivery method when you create a distribution. For a web distribution, this value is always **Web**. You can't change the delivery method for an existing distribution.

Origin Settings

When you create or update a distribution, you provide information about one or more locations—known as origins—where you store the original versions of your web content. CloudFront gets your web content from your origins and serves it to viewers via a world-wide network of edge servers. Each origin is either an Amazon S3 bucket or an HTTP server, for example, a web server.

For the current limit on the number of origins that you can create for a distribution or to request a higher limit, see [General Limits on Web Distributions \(p. 359\)](#).

If you want to delete an origin, you must first edit or delete the cache behaviors that are associated with that origin.

Important

If you delete an origin, confirm that files that were previously served by that origin are available in another origin and that your cache behaviors are now routing requests for those files to the new origin.

When you create or update a distribution, you specify the following values for each origin.

Origin Domain Name

The DNS domain name of the Amazon S3 bucket or HTTP server from which you want CloudFront to get objects for this origin, for example:

- **Amazon S3 bucket** – `myawsbucket.s3.amazonaws.com`
- **Amazon EC2 instance** – `ec2-203-0-113-25.compute-1.amazonaws.com`
- **Elastic Load Balancing load balancer** – `my-load-balancer-1234567890.us-west-2.elb.amazonaws.com`

If your origin is an HTTP server, type the domain name of the resource. The files must be publicly readable.

If your origin is an Amazon S3 bucket, in the CloudFront console, choose in the **Origin Domain Name** field, and a list enumerates the Amazon S3 buckets that are associated with the current AWS account. Note the following:

- If the bucket is configured as a website, enter the Amazon S3 static website hosting endpoint for your bucket; do not select the bucket name from the list in the **Origin Domain Name** field. The static website hosting endpoint appears in the Amazon S3 console, on the **Properties** page under **Static Website Hosting**.
- If you configured Amazon S3 Transfer Acceleration for your bucket, do not specify the `s3-accelerate` endpoint for **Origin Domain Name**.
- If you're using a bucket from a different AWS account and if the bucket is not configured as a website, type the name in the following format:

`bucket-name.s3.amazonaws.com`

If your bucket is in the US Standard region and you want Amazon S3 to route requests to a facility in Northern Virginia, use the following format:

`bucket-name.s3-external-1.amazonaws.com`

If your bucket is in the EU (Frankfurt) region, you can also use the following format:

`bucket-name.s3.eu-central-1.amazonaws.com`

- The files must be publicly readable unless you secure your content in Amazon S3 by using a CloudFront origin access identity. For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

Important

If the origin is an Amazon S3 bucket, the bucket name must conform to DNS naming requirements. For more information, go to [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

When you change the value of **Origin Domain Name** for an origin, CloudFront immediately begins replicating the change to CloudFront edge locations. Until the distribution configuration is updated in a given edge location, CloudFront will continue to forward requests to the previous HTTP server or

Amazon S3 bucket. As soon as the distribution configuration is updated in that edge location, CloudFront begins to forward requests to the new HTTP server or Amazon S3 bucket.

Changing the origin does not require CloudFront to repopulate edge caches with objects from the new origin. As long as the viewer requests in your application have not changed, CloudFront will continue to serve objects that are already in an edge cache until the TTL on each object expires or until seldom-requested objects are evicted.

Origin Path

If you want CloudFront to request your content from a directory in your Amazon S3 bucket or your custom origin, enter the directory path, beginning with a /. CloudFront appends the directory path to the value of **Origin Domain Name**, for example, **cf-origin.example.com/production/images**. Do not add a / at the end of the path.

For example, suppose you've specified the following values for your distribution:

- **Origin Domain Name** – An Amazon S3 bucket named **myawsbucket**
- **Origin Path** – **/production**
- **Alternate Domain Names (CNAMEs)** – **example.com**

When a user enters **example.com/index.html** in a browser, CloudFront sends a request to Amazon S3 for **myawsbucket/production/index.html**.

When a user enters **example.com/acme/index.html** in a browser, CloudFront sends a request to Amazon S3 for **myawsbucket/production/acme/index.html**.

Origin ID

A string that uniquely distinguishes this origin from other origins in this distribution. If you create cache behaviors in addition to the default cache behavior, you use the origin ID that you specify here to identify the origin to which you want CloudFront to route a request when the request matches the path pattern for that cache behavior. For more information, see [Cache Behavior Settings \(p. 70\)](#).

Restrict Bucket Access (Amazon S3 Only)

Choose **Yes** if you want to require users to access objects in an Amazon S3 bucket by using only CloudFront URLs, not by using Amazon S3 URLs. Then specify the applicable values.

Choose **No** if you want users to be able to access objects using either CloudFront URLs or Amazon S3 URLs.

For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

For information about how to require users to access objects on a custom origin by using only CloudFront URLs, see [Using Custom Headers to Restrict Access to Your Content on a Custom Origin \(p. 158\)](#).

Origin Access Identity (Amazon S3 Only)

If you chose **Yes** for **Restrict Bucket Access**, choose whether to create a new origin access identity or use an existing one that is associated with your AWS account. If you already have an origin access identity, we recommend that you reuse it to simplify maintenance. For more information about origin access identities, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

Comment for New Identity (Amazon S3 Only)

If you chose **Create a New Identity for Origin Access Identity**, enter a comment that identifies the new origin access identity. CloudFront will create the origin access identity when you create this distribution.

Your Identities (Amazon S3 Only)

If you chose **Use an Existing Identity for Origin Access Identity**, choose the origin access identity that you want to use. You cannot use an origin access identity that is associated with another AWS account.

Grant Read Permissions on Bucket (Amazon S3 Only)

If you want CloudFront to automatically grant the origin access identity the permission to read objects in your Amazon S3 bucket, choose **Yes, Update Bucket Policy**.

Important

If you choose **Yes, Update Bucket Policy**, CloudFront updates the bucket policy to grant the specified origin access identity the permission to read objects in your bucket. However, CloudFront does not remove existing permissions in the bucket policy or permissions on individual objects. If users currently have permission to access the objects in your bucket using Amazon S3 URLs, they will still have that permission after CloudFront updates your bucket policy. To view or change the existing bucket policy and the existing permissions on the objects in your bucket, use a method provided by Amazon S3. For more information, see [Granting the Origin Access Identity Permission to Read Objects in Your Amazon S3 Bucket \(p. 213\)](#).

If you want to update permissions manually, for example, if you want to update ACLs on your objects instead of updating bucket permissions, choose **No, I will Update Permissions**.

Origin SSL Protocols (Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only)

Choose the SSL protocols that CloudFront can use when establishing an HTTPS connection with your origin. The SSLv3 protocol is less secure, so we recommend that you choose SSLv3 only if your origin doesn't support TLSv1 or later.

Note

If you select SSLv3, CloudFront does not attempt to make a connection to the Origin using TLS.

If the origin is an Amazon S3 bucket, CloudFront always uses TLSv1.2.

Origin Protocol Policy (Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only)

The protocol policy that you want CloudFront to use when fetching objects from your origin server.

Important

If your Amazon S3 bucket is configured as a website endpoint, you must specify HTTP Only. Amazon S3 doesn't support HTTPS connections in that configuration.

Choose the applicable value:

- **HTTP Only:** CloudFront uses only HTTP to access the origin.
- **HTTPS Only:** CloudFront uses only HTTPS to access the origin.
- **Match Viewer:** CloudFront communicates with your origin using HTTP or HTTPS, depending on the protocol of the viewer request. CloudFront caches the object only once even if viewers make requests using both HTTP and HTTPS protocols.

Important

For HTTPS viewer requests that CloudFront forwards to this origin, one of the domain names in the SSL certificate on your origin server must match the domain name that you specify for **Origin Domain Name**. Otherwise, CloudFront responds to the viewer requests with an HTTP status code 502 (Bad Gateway) instead of the requested object. For more information, see [Requirements for Using SSL/TLS Certificates with CloudFront \(p. 285\)](#).

Origin Response Timeout (Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only)

The origin response timeout , also known as the origin read timeout or origin request timeout, applies to both of the following values:

- The amount of time, in seconds, that CloudFront waits for a response after forwarding a request to a custom origin
- The amount of time, in seconds, that CloudFront waits after receiving a packet of a response from the origin and before receiving the next packet

The default timeout is 30 seconds. You can change the value to between 4 and 60 seconds. If you need a timeout value outside that range, [request a change to the limit](#).

CloudFront behavior depends on the HTTP method in the viewer request:

- **GET** and **HEAD** requests – If the origin doesn't respond before the read timeout elapses or if the origin stops responding for the configured timeout, CloudFront drops the connection and tries two more times to contact the origin. After the third try, if the origin doesn't respond before the read timeout elapses, CloudFront doesn't try again until it receives another request for content on the same origin.
- **DELETE**, **OPTIONS**, **PATCH**, **PUT**, and **POST** requests – If the origin doesn't respond before the read timeout elapses, CloudFront drops the connection and doesn't try again to contact the origin. The client can resubmit the request if necessary.

Origin Keep-alive Timeout (Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only)

The amount of time, in seconds, that CloudFront tries to maintain a connection to your custom origin after it gets the last packet of a response. Maintaining a persistent connection saves the time that is required to re-establish the TCP connection and perform another TLS handshake for subsequent requests. Increasing the keep-alive timeout helps improve the request-per-connection metric for distributions.

Note

For the **Origin Keep-alive Timeout** value to have an effect, your origin must be configured to allow persistent connections.

The default timeout is 5 seconds. You can change the value to between 1 and 60 seconds. If you need a keep-alive timeout longer than 60 seconds, [request a change to the limit](#).

HTTP Port (Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only)

Optional. The HTTP port that the custom origin listens on. Valid values include ports 80, 443, and 1024 to 65535. The default value is port 80.

HTTPS Port (Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only)

Optional. The HTTPS port that the custom origin listens on. Valid values include ports 80, 443, and 1024 to 65535. The default value is port 443.

Origin Custom Headers

If you want CloudFront to include custom headers whenever it forwards a request to your origin, specify the following values:

Header Name

The name of a header that you want CloudFront to forward to your origin.

Value

The value for the header that you specified in the **Custom Header** field.

For more information, see [Forwarding Custom Headers to Your Origin \(Web Distributions Only\) \(p. 156\)](#).

For the current limit on the maximum number of custom headers that you can forward to the origin, the maximum length of a custom header name and value, and the total length of all header names and values, see [Limits \(p. 359\)](#).

Cache Behavior Settings

A cache behavior lets you configure a variety of CloudFront functionality for a given URL path pattern for files on your website. For example, one cache behavior might apply to all .jpg files in the `images` directory on a web server that you're using as an origin server for CloudFront. The functionality you can configure for each cache behavior includes:

- The path pattern.
- If you have configured multiple origins for your CloudFront distribution, which origin you want CloudFront to forward your requests to.
- Whether to forward query strings to your origin.
- Whether accessing the specified files requires signed URLs.
- Whether to require users to use HTTPS to access those files.
- The minimum amount of time that those files stay in the CloudFront cache regardless of the value of any Cache-Control headers that your origin adds to the files.

When you create a new distribution, you specify settings for the default cache behavior, which automatically forwards all requests to the origin that you specify when you create the distribution. After you create a distribution, you can create additional cache behaviors that define how CloudFront responds when it receives a request for objects that match a path pattern, for example, *.jpg. If you create additional cache behaviors, the default cache behavior is always the last to be processed. Other cache behaviors are processed in the order in which they're listed in the CloudFront console or, if you're using the CloudFront API, the order in which they're listed in the `DistributionConfig` element for the distribution. For more information, see [Path Pattern \(p. 71\)](#).

When you create a cache behavior, you specify the one origin from which you want CloudFront to get objects. As a result, if you want CloudFront to distribute objects from all of your origins, you must have at least as many cache behaviors (including the default cache behavior) as you have origins. For

example, if you have two origins and only the default cache behavior, the default cache behavior will cause CloudFront to get objects from one of the origins, but the other origin will never be used.

For the current limit on the number of cache behaviors that you can add to a distribution or to request a higher limit, see [General Limits on Web Distributions \(p. 359\)](#).

Path Pattern

A path pattern (for example, `images/*.jpg`) specifies which requests you want this cache behavior to apply to. When CloudFront receives an end-user request, the requested path is compared with path patterns in the order in which cache behaviors are listed in the distribution. The first match determines which cache behavior is applied to that request. For example, suppose you have three cache behaviors with the following three path patterns, in this order:

- `images/*.jpg`
- `images/*`
- `*.gif`

Note

You can optionally include a slash (/) at the beginning of the path pattern, for example, `/images/*.jpg`. CloudFront behavior is the same with or without the leading `/`.

A request for the file `images/sample.gif` doesn't satisfy the first path pattern, so the associated cache behaviors are not applied to the request. The file does satisfy the second path pattern, so the cache behaviors associated with the second path pattern are applied even though the request also matches the third path pattern.

Note

When you create a new distribution, the value of **Path Pattern** for the default cache behavior is set to `*` (all files) and cannot be changed. This value causes CloudFront to forward all requests for your objects to the origin that you specified in the [Origin Domain Name \(p. 66\)](#) field. If the request for an object does not match the path pattern for any of the other cache behaviors, CloudFront applies the behavior that you specify in the default cache behavior.

Important

Define path patterns and their sequence carefully or you may give users undesired access to your content. For example, suppose a request matches the path pattern for two cache behaviors. The first cache behavior does not require signed URLs and the second cache behavior does require signed URLs. Users will be able to access the objects without using a signed URL because CloudFront processes the cache behavior associated with the first match.

The path you specify applies to requests for all files in the specified directory and in subdirectories below the specified directory. CloudFront does not consider query strings or cookies when evaluating the path pattern. For example, if an `images` directory contains `product1` and `product2` subdirectories, the path pattern `images/*.jpg` applies to requests for any `.jpg` file in the `images`, `images/product1`, and `images/product2` directories. If you want to apply a different cache behavior to the files in the `images/product1` directory than the files in the `images` and `images/product2` directories, create a separate cache behavior for `images/product1` and move that cache behavior to a position above (before) the cache behavior for the `images` directory.

You can use the following wildcard characters in your path pattern:

- `*` matches 0 or more characters.
- `?` matches exactly 1 character.

The following examples show how the wildcard characters work:

| Path pattern | Files that match the path pattern |
|--------------|---|
| *.jpg | All .jpg files |
| images/*.jpg | All .jpg files in the images directory and in subdirectories under the images directory |
| a*.jpg | <ul style="list-style-type: none"> All .jpg files for which the filename begins with a, for example, apple.jpg and appalachian_trail_2012_05_21.jpg All .jpg files for which the file path begins with a, for example, abra/cadabra/magic.jpg. |
| a???.jpg | All .jpg files for which the filename begins with a and is followed by exactly two other characters, for example, ant.jpg and abe.jpg |
| *.doc* | All files for which the filename extension begins with .doc, for example, .doc, .docx, and .docm files. You can't use the path pattern *.doc? in this case, because that path pattern wouldn't apply to requests for .doc files; the ? wildcard character replaces exactly one character. |

The maximum length of a path pattern is 255 characters. The value can contain any of the following characters:

- A-Z, a-z
Path patterns are case sensitive, so the path pattern *.jpg doesn't apply to the file LOGO.JPG.
- 0-9
- _ - . * \$ / ~ " ' @ : +
- &, passed and returned as & ;

Origin (Existing Distributions Only)

Enter the value of **Origin ID** for an existing origin. This identifies the origin that you want CloudFront to route requests to when a request (such as `http://example.com/logo.jpg`) matches the path pattern for a cache behavior (such as `*.jpg`) or for the default cache behavior (*).

Viewer Protocol Policy

Choose the protocol policy that you want viewers to use to access your content in CloudFront edge locations:

- **HTTP and HTTPS:** Viewers can use both protocols.
- **Redirect HTTP to HTTPS:** Viewers can use both protocols, but HTTP requests are automatically redirected to HTTPS requests.
- **HTTPS Only:** Viewers can only access your content if they're using HTTPS.

For more information, see [Requiring HTTPS for Communication Between Viewers and CloudFront \(p. 278\)](#).

Field Level Encryption

If you want to enforce field-level encryption on specific data fields, in the dropdown lists, choose a field-level encryption configuration.

For more information, see [Using Field-Level Encryption to Help Protect Sensitive Data \(p. 85\)](#).

Allowed HTTP Methods

Specify the HTTP methods that you want CloudFront to process and forward to your origin:

- **GET, HEAD:** You can use CloudFront only to get objects from your origin or to get object headers.
- **GET, HEAD, OPTIONS:** You can use CloudFront only to get objects from your origin, get object headers, or retrieve a list of the options that your origin server supports.
- **GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE:** You can use CloudFront to get, add, update, and delete objects, and to get object headers. In addition, you can perform other POST operations such as submitting data from a web form.

Note

CloudFront caches responses to `GET` and `HEAD` requests and, optionally, `OPTIONS` requests. CloudFront does not cache responses to requests that use the other methods.

If you use an Amazon S3 bucket as the origin for your distribution and if you use CloudFront origin access identities, `POST` requests aren't supported in some Amazon S3 regions and `PUT` requests in those regions require an additional header. For more information, see [Using an Origin Access Identity in Amazon S3 Regions that Support Only Signature Version 4 Authentication \(p. 215\)](#).

Important

If you choose **GET, HEAD, OPTIONS** or **GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE**, you might need to restrict access to your Amazon S3 bucket or to your custom origin to prevent users from performing operations that you don't want them to perform. The following examples explain how to restrict access:

- **If you're using Amazon S3 as an origin for your distribution:** Create a CloudFront origin access identity to restrict access to your Amazon S3 content, and grant the origin access identity the applicable permissions. For example, if you configure CloudFront to accept and forward these methods *only* because you want to use `PUT`, you must still configure Amazon S3 bucket policies or ACLs to handle `DELETE` requests appropriately. For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).
- **If you're using a custom origin:** Configure your origin server to handle all methods. For example, if you configure CloudFront to accept and forward these methods *only* because you want to use `POST`, you must still configure your origin server to handle `DELETE` requests appropriately.

Cached HTTP Methods

Specify whether you want CloudFront to cache the response from your origin when a viewer submits an `OPTIONS` request. CloudFront always caches the response to `GET` and `HEAD` requests.

Cache Based on Selected Request Headers

Specify whether you want CloudFront to cache objects based on the values of specified headers:

- **None (Improves Caching)** – CloudFront doesn't cache your objects based on header values.
- **Whitelist** – CloudFront caches your objects based only on the values of the specified headers. Use **Whitelist Headers** to choose the headers that you want CloudFront to base caching on.

If you're using an Amazon S3 bucket as your origin, you can configure CloudFront to cache based only on the following headers: `Access-Control-Request-Headers`, `Access-Control-Request-Method`, and `Origin`.

- **All** – CloudFront doesn't cache the objects that are associated with this cache behavior. Instead, CloudFront sends every request to the origin.

Regardless of which option you choose, CloudFront forwards headers to your origin based on whether the origin is an S3 bucket or a custom origin. See the following documentation:

- **S3 bucket** – See [HTTP Request Headers That CloudFront Removes or Updates \(p. 186\)](#)
- **Custom origin** – See [HTTP Request Headers and CloudFront Behavior \(p. 194\)](#)

For more information, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

Whitelist Headers

Specify the headers that you want CloudFront to consider when caching your objects. Select headers from the list of available headers and choose **Add**. To forward a custom header, enter the name of the header in the field, and choose **Add Custom**.

For the current limit on the number of headers that you can whitelist for each cache behavior or to request a higher limit, see [Limits on Custom Headers \(Web Distributions Only\) \(p. 361\)](#).

Object Caching

If your origin server is adding a Cache-Control header to your objects to control how long the objects stay in the CloudFront cache and if you don't want to change the Cache-Control value, choose **Use Origin Cache Headers**.

To specify a minimum and maximum time that your objects stay in the CloudFront cache regardless of Cache-Control headers, and a default time that your objects stay in the CloudFront cache when the Cache-Control header is missing from an object, choose **Customize**. Then, in the **Minimum TTL**, **Default TTL**, and **Maximum TTL** fields, specify the applicable value.

For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

Minimum TTL

Specify the minimum amount of time, in seconds, that you want objects to stay in CloudFront caches before CloudFront forwards another request to your origin to determine whether the object has been updated. The default value for **Minimum TTL** is 0 seconds.

Important

If you configure CloudFront to forward all headers to your origin for a cache behavior, CloudFront never caches the associated objects. Instead, CloudFront forwards all requests for those objects to the origin. In that configuration, the value of **Minimum TTL** must be 0.

To specify a value for **Minimum TTL**, you must choose the **Customize** option for the **Object Caching** setting.

For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

Maximum TTL

Specify the maximum amount of time, in seconds, that you want objects to stay in CloudFront caches before CloudFront queries your origin to see whether the object has been updated. The value that you specify for **Maximum TTL** applies only when your origin adds HTTP headers such as Cache-Control max-age, Cache-Control s-maxage, or Expires to objects. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

To specify a value for **Maximum TTL**, you must choose the **Customize** option for the **Object Caching** setting.

The default value for **Maximum TTL** is 31536000 seconds (one year). If you change the value of **Minimum TTL** or **Default TTL** to more than 31536000 seconds, then the default value of **Maximum TTL** changes to the value of **Default TTL**.

Default TTL

Specify the default amount of time, in seconds, that you want objects to stay in CloudFront caches before CloudFront forwards another request to your origin to determine whether the object has been updated. The value that you specify for **Default TTL** applies only when your origin does *not* add HTTP headers such as Cache-Control max-age, Cache-Control s-maxage, or Expires to objects. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

To specify a value for **Default TTL**, you must choose the **Customize** option for the **Object Caching** setting.

The default value for **Default TTL** is 86400 seconds (one day). If you change the value of **Minimum TTL** to more than 86400 seconds, then the default value of **Default TTL** changes to the value of **Minimum TTL**.

Forward Cookies (Amazon EC2 and Other Custom Origins Only)

Specify whether you want CloudFront to forward cookies to your origin server and, if so, which ones. If you choose to forward only selected cookies (a whitelist of cookies), enter the cookie names in the **Whitelist Cookies** field. If you choose **All**, CloudFront forwards all cookies regardless of how many your application uses.

Amazon S3 doesn't process cookies, and forwarding cookies to the origin reduces cacheability. For cache behaviors that are forwarding requests to an Amazon S3 origin, choose **None** for **Forward Cookies**.

For more information about forwarding cookies to the origin, go to [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).

Whitelist Cookies (Amazon EC2 and Other Custom Origins Only)

If you chose **Whitelist** in the **Forward Cookies** list, then in the **Whitelist Cookies** field, enter the names of cookies that you want CloudFront to forward to your origin server for this cache behavior. Enter each cookie name on a new line.

You can specify the following wildcards to specify cookie names:

- * matches 0 or more characters in the cookie name
- ? matches exactly one character in the cookie name

For example, suppose viewer requests for an object include a cookie named:

`userid_member-number`

where each of your users has a unique value for `member-number`. You want CloudFront to cache a separate version of the object for each member. You could accomplish this by forwarding all cookies to your origin, but viewer requests include some cookies that you don't want CloudFront to cache. Alternatively, you could specify the following value as a cookie name, which causes CloudFront to forward to the applicable origin all of the cookies that begin with `userid_`:

`userid_*`

For the current limit on the number of cookie names that you can whitelist for each cache behavior or to request a higher limit, see [Limits on Whitelisted Cookies \(Web Distributions Only\) \(p. 360\)](#).

Query String Forwarding and Caching

CloudFront can cache different versions of your content based on the values of query string parameters. Choose the applicable option:

None (Improves Caching)

Choose this option if your origin returns the same version of an object regardless of the values of query string parameters. This increases the likelihood that CloudFront can serve a request from the cache, which improves performance and reduces the load on your origin.

Forward all, cache based on whitelist

Choose this option if your origin server returns different versions of your objects based on one or more query string parameters. Then specify the parameters that you want CloudFront to use as a basis for caching in the [Query String Whitelist \(p. 76\)](#) field.

Forward all, cache based on all

Choose this option if your origin server returns different versions of your objects for all query string parameters.

For more information about caching based on query string parameters, including how to improve performance, see [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#).

Query String Whitelist

If you chose **Forward all, cache based on whitelist** for [Query String Forwarding and Caching \(p. 76\)](#), specify the query string parameters that you want CloudFront to use as a basis for caching.

Smooth Streaming

Choose **Yes** if you want to distribute media files in the Microsoft Smooth Streaming format using the origin that is associated with this cache behavior. Otherwise, choose **No**.

Note

If you specify **Yes**, you can still distribute other content using this cache behavior if the content matches the value of **Path Pattern**.

For more information, see [Configuring On-Demand Smooth Streaming \(p. 96\)](#).

Restrict Viewer Access (Use Signed URLs)

If you want requests for objects that match the **PathPattern** for this cache behavior to use public URLs, choose **No**.

If you want requests for objects that match the **PathPattern** for this cache behavior to use signed URLs, choose **Yes**. Then specify the AWS accounts that you want to use to create signed URLs; these accounts are known as trusted signers.

For more information about trusted signers, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#).

Trusted Signers

Choose which AWS accounts you want to use as trusted signers for this cache behavior:

- **Self:** Use the account with which you're currently signed into the AWS Management Console as a trusted signer. If you're currently signed in as an IAM user, the associated AWS account is added as a trusted signer.
- **Specify Accounts:** Enter account numbers for trusted signers in the **AWS Account Numbers** field.

To create signed URLs, an AWS account must have at least one active CloudFront key pair.

Important

If you're updating a distribution that you're already using to distribute content, add trusted signers only when you're ready to start generating signed URLs for your objects. After you add trusted signers to a distribution, users must use signed URLs to access the objects that match the `PathPattern` for this cache behavior.

AWS Account Numbers

If you want to create signed URLs using AWS accounts in addition to or instead of the current account, enter one AWS account number per line in this field. Note the following:

- The accounts that you specify must have at least one active CloudFront key pair. For more information, see [Creating CloudFront Key Pairs for Your Trusted Signers \(p. 217\)](#).
- You can't create CloudFront key pairs for IAM users, so you can't use IAM users as trusted signers.
- For information about how to get the AWS account number for an account, see [How Do I Get Security Credentials?](#) in the *Amazon Web Services General Reference*.
- If you enter the account number for the current account, CloudFront automatically checks the **Self** checkbox and removes the account number from the **AWS Account Numbers** list.

Compress Objects Automatically

If you want CloudFront to automatically compress files of certain types when viewer requests include `Accept-Encoding: gzip` in the request header, choose **Yes**. When CloudFront compresses your content, downloads are faster because the files are smaller, and your web pages render faster for your users. For more information, see [Serving Compressed Files \(p. 179\)](#).

Event Type

You can choose to run a Lambda function when one or more of the following CloudFront events occur:

- When CloudFront receives a request from a viewer (viewer request)
- Before CloudFront forwards a request to the origin (origin request)
- When CloudFront receives a response from the origin (origin response)
- Before CloudFront returns the response to the viewer (viewer response)

For more information, see [How You Decide Which CloudFront Event to Use to Trigger a Lambda Function \(p. 101\)](#).

Lambda Function ARN

Specify the Amazon Resource Name (ARN) of the Lambda function that you want to add a trigger for. To learn how to get the ARN for a function, see step 1 of the procedure [Adding Triggers for CloudFront Events \(CloudFront Console\) \(p. 107\)](#).

Distribution Details

The following values apply to the entire distribution.

Price Class

Choose the price class that corresponds with the maximum price that you want to pay for CloudFront service. By default, CloudFront serves your objects from edge locations in all CloudFront regions.

For more information about price classes and about how your choice of price class affects CloudFront performance for your distribution, see [Choosing the Price Class for a CloudFront Distribution \(p. 55\)](#). For information about CloudFront pricing, including how price classes map to CloudFront regions, go to [Amazon CloudFront Pricing](#).

AWS WAF Web ACL

If you want to use AWS WAF to allow or block requests based on criteria that you specify, choose the web ACL to associate with this distribution.

AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to CloudFront, and lets you control access to your content. Based on conditions that you specify, such as the IP addresses that requests originate from or the values of query strings, CloudFront responds to requests either with the requested content or with an HTTP 403 status code (Forbidden). You can also configure CloudFront to return a custom error page when a request is blocked. For more information about AWS WAF, see the [AWS WAF Developer Guide](#).

Alternate Domain Names (CNAMEs)

Optional. Specify one or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution. For example, if you want the URL for the object:

/images/image.jpg

to look like this:

http://www.example.com/images/image.jpg

instead of like this:

http://d111111abcdef8.cloudfront.net/images/image.jpg

add a CNAME for www.example.com.

Important

If you add a CNAME for www.example.com to your distribution, you also need to create (or update) a CNAME record with your DNS service to route queries for www.example.com to d111111abcdef8.cloudfront.net. You must have permission to create a CNAME record with the DNS service provider for the domain. Typically, this means that you own the domain, but you may also be developing an application for the domain owner.

For the current limit on the number of alternate domain names that you can add to a distribution or to request a higher limit, see [General Limits on Web Distributions \(p. 359\)](#).

For more information about alternate domain names, see [Using Alternate Domain Names \(CNAMEs\) \(p. 50\)](#). For more information about CloudFront URLs, see [Format of URLs for Objects \(p. 143\)](#).

SSL Certificate

If you want viewers to use HTTPS to access your objects, choose the applicable setting. In addition, if you choose **Custom SSL Certificate**, choose the certificate that you want to use:

- **Default CloudFront Certificate (*.cloudfront.net)** – If you want to use the CloudFront domain name in the URLs for your objects, such as `https://d111111abcdef8.cloudfront.net/image1.jpg`, choose this option. Also choose this option if you want viewers to use HTTP to access your objects.
- **Custom SSL Certificate** – If you want to use your own domain name in the URLs for your objects, such as `https://example.com/image1.jpg`, choose this option and then choose the applicable certificate. The list can include certificates provided by AWS Certificate Manager, and certificates that you purchased from a third-party certificate authority and uploaded to ACM or to the IAM certificate store. For more information, see [Using Alternate Domain Names and HTTPS \(p. 283\)](#).

If you choose this setting, we recommend that you use only an alternate domain name in your object URLs (`https://example.com/logo.jpg`). If you use your CloudFront distribution domain name (`https://d111111abcdef8.cloudfront.net/logo.jpg`) and the viewer supports SNI, then CloudFront behaves normally. However, a viewer that does not support SNI exhibits one of the following behaviors, depending on the value of **Clients Supported**:

- **All Clients:** If the viewer doesn't support SNI, it displays a warning because the CloudFront domain name doesn't match the domain name in your SSL certificate.
- **Only Clients that Support Server Name Indication (SNI):** CloudFront drops the connection with the viewer without returning the object.

Clients Supported

If you specified one or more alternate domain names and you specified an SSL certificate in the IAM certificate store, choose how you want CloudFront to serve HTTPS requests, either a method that works for all clients or one that works for most clients:

- **All Clients:** Any client can access your content. However, you must request permission to use this feature, and you incur additional monthly charges.
- **Only Clients that Support Server Name Indication (SNI):** All modern browsers can access your content because they all support SNI. However, some browsers still in use don't support SNI. Users with these browsers must access your content using some other method, for example, by getting your objects directly from the origin.

For more information, see [Using Alternate Domain Names and HTTPS \(p. 283\)](#).

Security Policy

Specify the security policy that you want CloudFront to use for HTTPS connections. A security policy determines two settings:

- The minimum SSL/TLS protocol that CloudFront uses to communicate with viewers
- The cipher that CloudFront uses to encrypt the content that it returns to viewers

The security policies that are available depend on the values that you specify for **SSL Certificate** and **Custom SSL Client Support**:

- When **SSL Certificate** is **Default CloudFront Certificate (*.cloudfront.net)**, CloudFront automatically sets the value of **Security Policy** to **TLSv1**.
- When **SSL Certificate** is **Custom SSL Certificate (example.com)** and **Custom SSL Client Support** is **Only Clients that Support Server Name Indication (SNI)**, you must use TLSv1 or later. We recommend that you choose **TLSv1.1_2016** unless your users are using browsers or devices that don't support TLSv1.1 or later.
- When **SSL Certificate** is **Custom SSL Certificate (example.com)** and **Custom SSL Client Support** is **All Clients**, we recommend that you choose **TLSv1**. In this configuration, the **TLSv1_2016**, **TLSv1.1_2016**, and **TLSv1.2_2018** security policies aren't available.

For information about the relationship between the security policy that you choose and the protocols and ciphers that CloudFront uses to communicate with viewers, see [Supported SSL/TLS Protocols and Ciphers for Communication Between Viewers and CloudFront \(p. 296\)](#).

Minimum SSL Protocol Version

See [Security Policy \(p. 79\)](#).

Supported HTTP Versions

Choose the HTTP versions that you want viewers to use to communicate with CloudFront. Viewers use the latest version that you configure CloudFront to use. Viewers that don't support HTTP/2 will automatically use an earlier version.

For viewers and CloudFront to use HTTP/2, viewers must support TLS 1.2 or later, and must support Server Name Identification (SNI).

In general, configuring CloudFront to communicate with viewers using HTTP/2 reduces latency. You can improve performance by optimizing for HTTP/2. For more information, do an internet search for "http/2 optimization."

Default Root Object

Optional. The object that you want CloudFront to request from your origin (for example, `index.html`) when a viewer requests the root URL of your distribution (`http://www.example.com/`) instead of an object in your distribution (`http://www.example.com/product-description.html`). Specifying a default root object avoids exposing the contents of your distribution.

The maximum length of the name is 255 characters. The name can contain any of the following characters:

- A-Z, a-z
- 0-9
- _ - . * \$ / ~ " '
- &, passed and returned as &

When you specify the default root object, enter only the object name, for example, `index.html`. Do not add a / before the object name.

For more information, see [Specifying a Default Root Object \(Web Distributions Only\) \(p. 177\)](#).

Logging

Whether you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket. You can enable or disable logging at any time. There is no extra charge if you enable logging, but you accrue the usual Amazon S3 charges for storing and accessing the files in an Amazon S3 bucket. You can delete the logs at any time. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Bucket for Logs

If you chose **On** for **Logging**, the Amazon S3 bucket that you want CloudFront to store access logs in, for example, `myawslogbucket.s3.amazonaws.com`. If you enable logging, CloudFront records information about each end-user request for an object and stores the files in the specified Amazon S3 bucket. You can enable or disable logging at any time. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Note

You must have the permissions required to get and update Amazon S3 bucket ACLs, and the S3 ACL for the bucket must grant you **FULL_CONTROL**. This allows CloudFront to give the awsdatafeeds account permission to save log files in the bucket. For more information, see [Permissions Required to Configure Logging and to Access Your Log Files \(p. 316\)](#).

Log Prefix

Optional. If you chose **On** for **Logging**, specify the string, if any, that you want CloudFront to prefix to the access log filenames for this distribution, for example, `exampleprefix/`. The trailing slash (/) is optional but recommended to simplify browsing your log files. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Cookie Logging

If you want CloudFront to include cookies in access logs, choose **On**. If you choose to include cookies in logs, CloudFront logs all cookies regardless of how you configure the cache behaviors for this distribution: forward all cookies, forward no cookies, or forward a specified list of cookies to the origin.

Amazon S3 doesn't process cookies, so unless your distribution also includes an Amazon EC2 or other custom origin, we recommend that you choose **Off** for the value of **Cookie Logging**.

For more information about cookies, go to [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).

Enable IPv6

IPv6 is a new version of the IP protocol. It's the eventual replacement for IPv4 and uses a larger address space. CloudFront always responds to IPv4 requests. If you want CloudFront to respond to requests from IPv4 IP addresses (such as 192.0.2.44) and requests from IPv6 addresses (such as 2001:0db8:85a3:0000:0000:8a2e:0370:7334), select **Enable IPv6**.

In general, you should enable IPv6 if you have users on IPv6 networks who want to access your content. However, if you're using signed URLs or signed cookies to restrict access to your content, and if you're using a custom policy that includes the `IpAddress` parameter to restrict the IP addresses that can access your content, do not enable IPv6. If you want to restrict access to some content by IP address and not restrict access to other content (or restrict access but not by IP address), you can create two distributions. For information about creating signed URLs by using a custom policy, see [Creating a Signed URL Using a Custom Policy \(p. 234\)](#). For information about creating signed cookies by using a custom policy, see [Setting Signed Cookies Using a Custom Policy \(p. 251\)](#).

If you're using an Route 53 alias resource record set to route traffic to your CloudFront distribution, you need to create a second alias resource record set when both of the following are true:

- You enable IPv6 for the distribution
- You're using alternate domain names in the URLs for your objects

For more information, see [Routing Traffic to an Amazon CloudFront Web Distribution by Using Your Domain Name](#) in the *Amazon Route 53 Developer Guide*.

If you created a CNAME resource record set, either with Route 53 or with another DNS service, you don't need to make any changes. A CNAME record will route traffic to your distribution regardless of the IP address format of the viewer request.

If you enable IPv6 and CloudFront access logs, the `c-ip` column will include values in IPv4 and IPv6 format. For more information, see [Access Logs \(p. 315\)](#).

Note

To maintain high customer availability, CloudFront will respond to viewer requests by using IPv4 if our data suggests that IPv4 will provide a better user experience. To find out what percentage of requests CloudFront is serving over IPv6, enable CloudFront logging for your distribution and parse the `c-ip` column, which contains the IP address of the viewer that made the request. This percentage should grow over time, but it will remain a minority of traffic as IPv6 is not yet supported by all viewer networks globally. Some viewer networks have excellent IPv6 support, but others don't support IPv6 at all. (A viewer network is analogous to your home internet or wireless carrier.)

For more information about our support for IPv6, see the [CloudFront FAQ](#). For information about enabling access logs, see the fields [Logging \(p. 80\)](#), [Bucket for Logs \(p. 80\)](#), and [Log Prefix \(p. 81\)](#).

Comment

Optional. When you create a distribution, you can include a comment of up to 128 characters. You can update the comment at any time.

Distribution State

Indicates whether you want the distribution to be enabled or disabled once it's deployed:

- *Enabled* means that as soon as the distribution is fully deployed you can deploy links that use the distribution's domain name and users can retrieve content. Whenever a distribution is enabled, CloudFront accepts and handles any end-user requests for content that use the domain name associated with that distribution.

When you create, modify, or delete a CloudFront distribution, it takes time for your changes to propagate to the CloudFront database. An immediate request for information about a distribution might not show the change. Propagation usually completes within minutes, but a high system load or network partition might increase this time.

- *Disabled* means that even though the distribution might be deployed and ready to use, users can't use it. Whenever a distribution is disabled, CloudFront doesn't accept any end-user requests that use the domain name associated with that distribution. Until you switch the distribution from disabled to enabled (by updating the distribution's configuration), no one can use it.

You can toggle a distribution between disabled and enabled as often as you want. Follow the process for updating a distribution's configuration. For more information, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#).

Custom Error Pages and Error Caching

You can have CloudFront return an object to the viewer (for example, an HTML file) when your Amazon S3 or custom origin returns an HTTP 4xx or 5xx status code to CloudFront. You can also specify how long an error response from your origin or a custom error page is cached in CloudFront edge caches. For more information, see [Customizing Error Responses \(p. 172\)](#).

Note

The following values aren't included in the Create Distribution wizard, so you can configure custom error pages only when you update a distribution.

Error Code

The HTTP status code for which you want CloudFront to return a custom error page. You can configure CloudFront to return custom error pages for none, some, or all of the HTTP status codes that CloudFront caches.

Response Page Path

The path to the custom error page (for example, `/4xx-errors/403-forbidden.html`) that you want CloudFront to return to a viewer when your origin returns the HTTP status code that you specified for **Error Code** (for example, 403). If you want to store your objects and your custom error pages in different locations, your distribution must include a cache behavior for which the following is true:

- The value of **Path Pattern** matches the path to your custom error messages. For example, suppose you saved custom error pages for 4xx errors in an Amazon S3 bucket in a directory named `/4xx-errors`. Your distribution must include a cache behavior for which the path pattern routes requests for your custom error pages to that location, for example, `/4xx-errors/*`.
- The value of **Origin** specifies the value of **Origin ID** for the origin that contains your custom error pages.

Response Code

The HTTP status code that you want CloudFront to return to the viewer along with the custom error page.

Error Caching Minimum TTL

The minimum amount of time that you want CloudFront to cache error responses from your origin server.

Restrictions

If you need to prevent users in selected countries from accessing your content, you can configure your CloudFront distribution either to allow users in a whitelist of specified countries to access your content or to not allow users in a blacklist of specified countries to access your content. For more information, see [Restricting the Geographic Distribution of Your Content \(p. 94\)](#).

Note

The following values aren't included in the Create Distribution wizard, so you can configure geo restrictions only when you update a distribution.

Enable Geo Restriction

Whether you want to prevent users in selected countries from accessing your content. There is no additional charge for configuring geo restriction.

Restriction Type

How you want to specify the countries from which your users can access your content:

- **Whitelist:** The **Countries** list includes all of the countries from which you *do* want your users to access your content.
- **Blacklist:** The **Countries** list includes all of the countries from which you *do not* want your users to access your content.

Countries

The countries that you want to add to your whitelist or blacklist. To add a country, select it in the list on the left and choose **Add**. Note the following:

- To add multiple consecutive countries, select the first country, press and hold the Shift key, select the last country, and choose **Add**.
- To add multiple non-consecutive countries, select the first country, press and hold the Ctrl key, select the remaining countries, and choose **Add**.
- To find a country in the left list, enter the first few characters of the country's full name.
- The two-letter code before the name of each country is the value that you enter if you want to create or update a distribution by using the CloudFront API. We use the International Organization for Standardization country codes. For an easy-to-use list, sortable by code and by country name, see the Wikipedia entry [ISO 3166-1 alpha-2](#).

Values that CloudFront Displays in the Console When You Create or Update a Web Distribution

When you create a new web distribution or update an existing distribution, CloudFront displays the following information in the CloudFront console.

Note

Active trusted signers, the AWS accounts that have an active CloudFront key pair and can be used to create valid signed URLs, are currently not visible in the CloudFront console.

Distribution ID (General Tab)

When you perform an action on a distribution using the CloudFront API, you use the distribution ID to specify which distribution you want to perform the action on, for example, EDFDVBD6EXAMPLE. You can't change the distribution ID.

Distribution Status (General Tab)

The possible status values for a distribution are listed in the following table.

| Value | Description |
|------------|--|
| InProgress | The distribution is still being created or updated. |
| Deployed | The distribution has been created or updated and the changes have been fully propagated through the CloudFront system. |

Note

In addition to ensuring that the status for a distribution is **Deployed**, you must enable the distribution before users can use CloudFront to access your content. For more information, see [Distribution State \(p. 82\)](#).

Last Modified (General Tab)

The date and time that the distribution was last modified, using ISO 8601 format, for example, 2012-05-19T19:37:58Z. For more information, go to <http://www.w3.org/TR/NODE-datetime>.

Domain Name (General Tab)

You use the distribution's domain name in the links to your objects. For example, if your distribution's domain name is d111111abcdef8.cloudfront.net, the link to /images/image.jpg would be

<http://d111111abcdef8.cloudfront.net/images/image.jpg>. You can't change the CloudFront domain name for your distribution. For more information about CloudFront URLs for links to your objects, see [Format of URLs for Objects \(p. 143\)](#).

If you specified one or more alternate domain names (CNAMEs), you can use your own domain names for links to your objects instead of using the CloudFront domain name. For more information about CNAMEs, see [Alternate Domain Names \(CNAMEs\) \(p. 78\)](#).

Note

CloudFront domain names are unique. Your distribution's domain name was never used for a previous distribution and will never be reused for another distribution in the future.

Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins

Follow these guidelines for using Amazon EC2 instances and other custom origins with CloudFront.

- Host and serve the same content on all servers.
- Log the `X-Amz-Cf-Id` header entries on all servers; CloudFront requires this information for debugging.
- Restrict access requests to the HTTP and HTTPS ports that your custom origin listens on.
- Synchronize the clocks of all servers in your implementation.
- Use redundant servers to handle failures.
- For information about using a custom origin to serve private content, see [Using an HTTP Server for Private Content \(p. 209\)](#).
- For information about request and response behavior and about supported HTTP status codes, see [Request and Response Behavior \(p. 184\)](#).

If you use Amazon Elastic Compute Cloud for your custom origins, we recommend that you do the following:

1. Use an Amazon Machine Image that automatically installs the software for a web server. For more information, see the [Amazon EC2 documentation](#).
2. Use an Elastic Load Balancing load balancer to handle traffic across multiple Amazon EC2 instances and to isolate your application from changes to Amazon EC2 instances. For example, if you use a load balancer, you can add and delete Amazon EC2 instances without changing your application. For more information, see the [Elastic Load Balancing documentation](#).
3. When you create your CloudFront distribution, specify the URL of the load balancer for the domain name of your origin server. For more information, see [Working with Web Distributions \(p. 59\)](#).

Using Field-Level Encryption to Help Protect Sensitive Data

You can already configure CloudFront to help enforce secure end-to-end connections to origin servers by using HTTPS. Field-level encryption adds an additional layer of security along with HTTPS that lets you protect specific data throughout system processing so that only certain applications can see it.

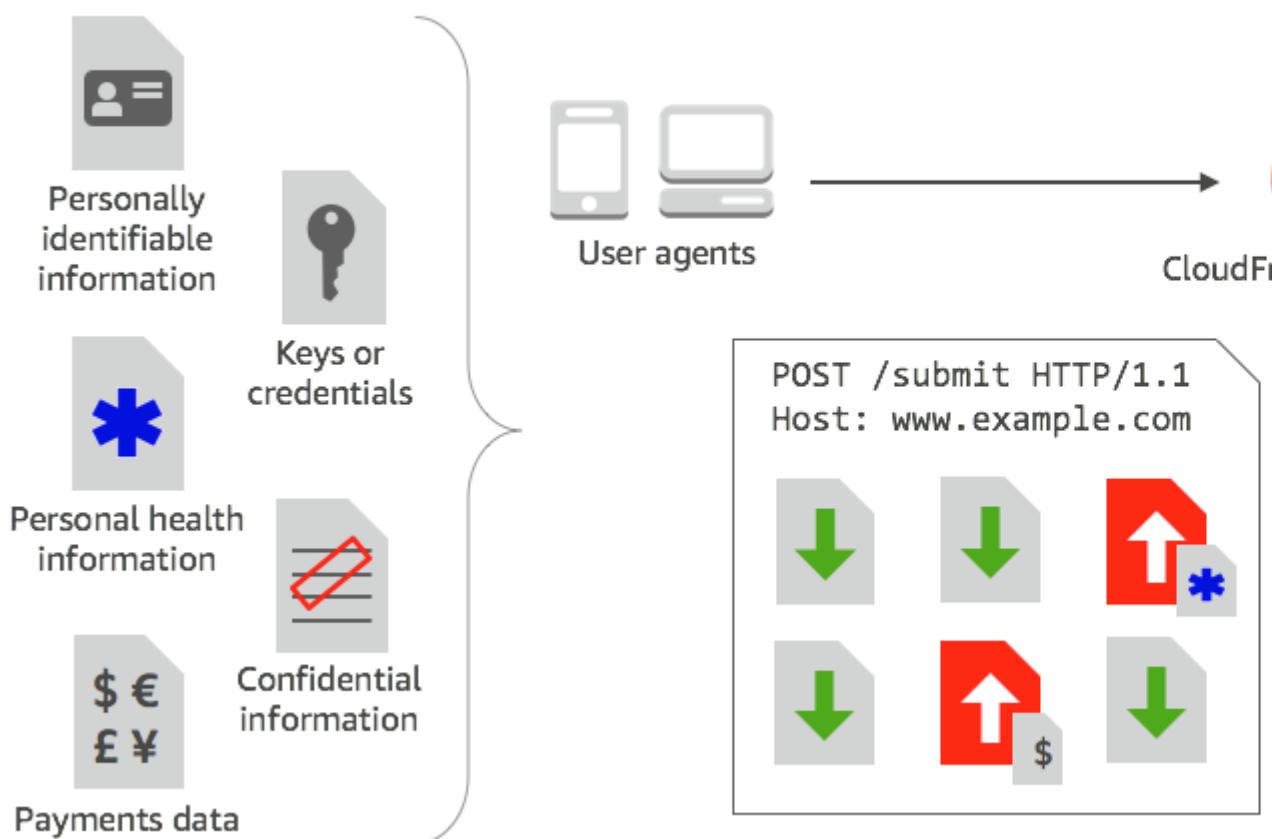
Field-level encryption allows you to securely upload user-submitted sensitive information to your web servers. The sensitive information provided by your clients is encrypted at the edge closer to the user and

remains encrypted throughout your entire application stack, ensuring that only applications that need the data—and have the credentials to decrypt it—are able to do so.

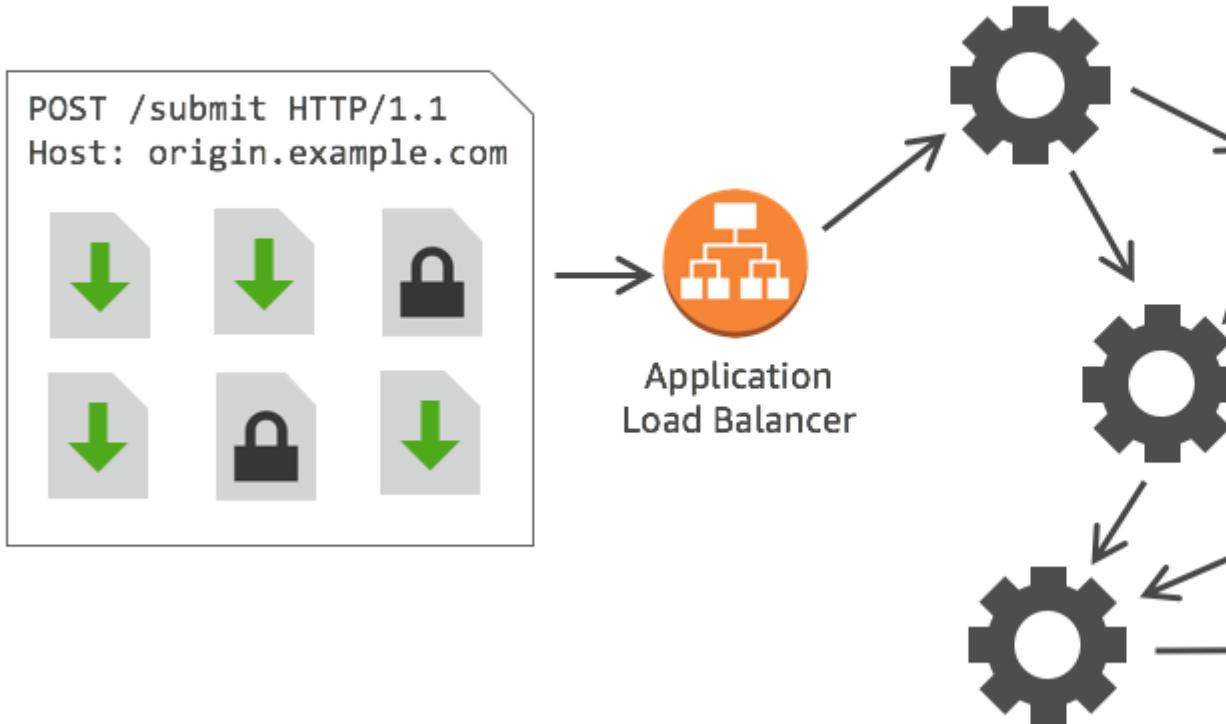
To use field-level encryption, you configure your CloudFront distribution to specify the set of fields in POST requests that you want to be encrypted, and the public key to use to encrypt them. You can encrypt up to 10 data fields in a request. (You can't encrypt all of the data in a request with field-level encryption; you must specify individual fields to encrypt.)

When the HTTPS request with field-level encryption is forwarded to the origin, and the request is routed throughout your origin sub-system, the sensitive data is still encrypted, reducing the risk of a data breach or accidental data loss of the sensitive data. Components that need access to the sensitive data for business reasons, such as a payment processing system needing access to a credit number, can use the appropriate private key to decrypt and access the data.

Be aware that in order to use field-level encryption, your origin must support chunked encoding.



CloudFront field-level encryption uses asymmetric encryption, also known as public-key encryption. You provide a public key to CloudFront, and all sensitive data that you specify is encrypted automatically. The key you provide to CloudFront cannot be used to decrypt the encrypted values; only your private key can do that.



Topics

- [Overview of Field-Level Encryption \(p. 87\)](#)
- [Setting Up Field-Level Encryption \(p. 88\)](#)
- [Decrypting Data Fields at Your Origin \(p. 91\)](#)

Overview of Field-Level Encryption

The following steps provide an overview of setting up field-level encryption. For specific steps, see [Setting Up Field-Level Encryption \(p. 88\)](#).

1. **Get a public key-private key pair.** You must obtain and add the public key before you start setting up field-level encryption in CloudFront.
2. **Create a field-level encryption profile.** Field-level encryption profiles, which you create in CloudFront, define the fields that you want to be encrypted.
3. **Create a field-level encryption configuration.** A configuration specifies the profiles to use—based on the content type of the request or a query argument—for encrypting specific data fields. You can also choose the request-forwarding behavior options that you want for different scenarios; for example, when the profile name specified by the query argument in a request URL doesn't exist in CloudFront.

4. **Link to a cache behavior.** Link the configuration to a cache behavior for a distribution, to specify when CloudFront should encrypt data.

Setting Up Field-Level Encryption

Follow these steps to get started using field-level encryption. To learn about limits in field-level encryption, see [Limits \(p. 359\)](#).

- [Step 1: Get an RSA Key Pair \(p. 88\)](#)
- [Step 2: Add Your Public Key to CloudFront \(p. 88\)](#)
- [Step 3: Create a Profile for Field-Level Encryption \(p. 88\)](#)
- [Step 4: Create a Configuration \(p. 89\)](#)
- [Step 5: Add a Configuration to a Cache Behavior \(p. 91\)](#)

Step 1: Get an RSA Key Pair

To get started, you must obtain a RSA key pair that includes a public key, so CloudFront can encrypt data, and a private key, so components at your origin to decrypt the fields that have been encrypted. For example, you can use Open SSL or another tool to create a key pair. The key size must be 2048 bits. For more information, see [To create an RSA key pair and upload the public key in the AWS Management Console \(p. 217\)](#).

Step 2: Add Your Public Key to CloudFront

After you get your RSA key pair, add your public key to CloudFront.

To add your public key to CloudFront (console)

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the navigation pane, choose **Public key**.
3. Choose **Add public key**.
4. For **Key name**, type a unique name for the key. The name can't have spaces and can include only alphanumeric characters, underscores (_), and hyphens (-). The maximum number of characters is 128.
5. For **Encoded key**, copy and paste the encoded key value for your public key, including the " ----- BEGIN PUBLIC KEY-----" and "-----END PUBLIC KEY-----" lines.
6. For **Comment**, add an optional comment. For example, you could include the expiration date for the public key.
7. Choose **Add key**.

You can add more keys to use with CloudFront by repeating the steps in the procedure.

Step 3: Create a Profile for Field-Level Encryption

After you add at least one public key to CloudFront, create a profile that tells CloudFront which fields to encrypt.

To create a profile for field-level encryption (console)

1. In the navigation pane, choose **Field-level encryption**.
2. Choose **Create profile**.

3. Fill in the following fields:

Profile name

Type a unique name for the profile. The name can't have spaces and can include only alphanumeric characters, underscores (_), and hyphens (-). The maximum number of characters is 128.

Public key name

In the dropdown list, choose the name of a public key that you added to CloudFront in step 2. CloudFront uses the key to encrypt the fields that you specify in this profile.

Provider name

Type a phrase to help identify the key, such as the provider where you got the key pair. This information, along with the private key, will be needed when applications decrypt data fields. The provider name can't have spaces and can include only alphanumeric characters, colons (:), underscores (_), and hyphens (-). The maximum number of characters is 128.

Field name pattern to match

Type the names of the data fields, or patterns that identify data field names in the request, that you want CloudFront to encrypt. Choose the + option to add all the fields that you want to encrypt with this key.

For the field name pattern, you can type the entire name of the data field, like DateOfBirth, or just the first part of the name with a wildcard character (*), like CreditCard*. The field name pattern must include only alphanumeric characters, square brackets ([and]), periods (.), underscores (_), and hyphens (-), in addition to the optional wildcard character (*).

Make sure that you don't use overlapping characters for different field name patterns. For example, if you have a field name pattern of ABC*, you can't add another field name pattern that is AB*. In addition, note that field names are case sensitive and the maximum number of characters that you can use is 128.

Comment

(Optional) Type a comment about this profile. The maximum number of characters that you can use is 128.

4. After you fill in the fields, choose **Create profile**.
5. If you want to add more profiles, choose **Add profile**.

Step 4: Create a Configuration

After you create one or more field-level encryption profiles, create a configuration that specifies the content type of the request that includes the data to be encrypted, the profile to use for encryption, and other options that specify how you want CloudFront to handle encryption.

For example, when CloudFront can't encrypt the data, you can specify whether CloudFront should block or forward a request to your origin in the following scenarios:

- **When a request's content type isn't in a configuration.** If you haven't added a content type to a configuration, you can specify whether CloudFront should forward the request with that content type to the origin without encrypting data fields, or block the request and return an error.

Note: If you add a content type to a configuration but haven't specified a profile to use with that type, requests with that content type will always be forwarded to the origin.

- **When the profile name provided in a query argument is unknown.** When you specify the `file-profile` query argument with a profile name that doesn't exist for your distribution, you can specify

whether CloudFront should send the request to the origin without encrypting data fields, or block the request and return an error.

In a configuration, you can also specify whether providing a profile as a query argument in a URL overrides a profile that you've mapped to the content type for that query. By default, CloudFront uses the profile that you've mapped to a content type, if you specify one. This lets you have a profile that's used by default but decide for certain requests that you want to enforce a different profile.

So, for example, you might specify (in your configuration) SampleProfile as the query argument profile to use. Then you could use the URL `https://d1234.cloudfront.net?file-profile=SampleProfile` instead of `https://d1234.cloudfront.net`, to have CloudFront use SampleProfile for this request, instead of the profile you'd set up for the content type of the request.

You can create up to 10 configurations for a single account, and then associate one of the configurations to the cache behavior of any distribution for the account.

To create a configuration for field-level encryption (console)

1. On the **Field-level encryption** page, choose **Create configuration**.

Note: If you haven't created at least one profile, you won't see the option to create a configuration.

2. Fill in the following fields to specify the profile to use. (Some fields can't be changed.)

Content type (can't be changed)

The content format is set to `application/x-www-form-urlencoded` and can't be changed.

Default profile ID (optional)

In the dropdown list, choose the profile that you want to map to the content type in the **Content type** field.

Content format (can't be changed)

The content format is set to `URLencoded` and can't be changed.

3. If you want to change the CloudFront default behavior for the following options, select the appropriate check box.

Forward request to origin when request's content type is not configured

Select the check box if you want to allow the request to go to your origin *if you have not specified a profile to use for the content type of the request*.

Override the profile for a content type with a provided query argument

Select the check box if you want to allow a profile provided in a query argument to *override the profile that you've specified for a content type*.

4. If you select the check box to allow a query argument to override the default profile, you must complete the following additional fields for the configuration. You can create up to five of these query argument mappings to use with queries.

Query argument

Type the value that you want to include in URLs for the `file-profile` query argument. This value tells CloudFront to use the profile ID (that you specify in the next field) associated with this query argument for field-level encryption for this query.

The maximum number of characters that you can use is 128. The value can't include spaces, and must use only alphanumeric or the following characters: dash (-), period (.), underscore (_), asterisk (*), plus-sign (+), percent (%).

Profile ID

In the dropdown list, choose the profile that you want to associate with the value that you typed for **Query argument**.

Forward request to origin when the profile specified in a query argument does not exist

Select the check box if you want to allow the request to go to your origin *if the profile specified in a query argument isn't defined in CloudFront*.

Step 5: Add a Configuration to a Cache Behavior

To use field-level encryption, link a configuration to a cache behavior for a distribution by adding the configuration ID as a value for your distribution. Note that the Viewer Protocol Policy and Origin Protocol Policy must be HTTPS in order for you to link a configuration to a cache behavior.

For more information, see [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

Decrypting Data Fields at Your Origin

CloudFront encrypts data fields by using the [AWS Encryption SDK](#). The data remains encrypted throughout your application stack and can be accessed only by applications that have the credentials to decrypt it.

After encryption, the cipher text is base64 encoded. When your applications decrypt the text at the origin, they must first decode the cipher text, and then use the AWS Encryption SDK to decrypt the data.

The following code sample illustrates how applications can decrypt data at your origin. Note the following:

- To simplify the example, this sample loads public and private keys (in DER format) from files in the working directory. In practice, you would store the private key in a secure offline location, such as an offline hardware security module, and distribute the public key to your development team.
- CloudFront uses specific information while encrypting the data, and the same set of parameters should be used at the origin to decrypt it. Parameters CloudFront uses while initializing the MasterKey include the following:
 - PROVIDER_NAME: You specified this value when you created a field-level encryption profile. Use the same value here.
 - KEY_NAME: You created a name for your public key when you uploaded it to CloudFront, and then specified the key name in the profile. Use the same value here.
 - ALGORITHM: CloudFront uses "RSA/ECB/OAEPWithSHA-256AndMGF1Padding" as the algorithm for encrypting, so you must use the same algorithm to decrypt the data.
- If you run the following sample program with cipher text as input, the decrypted data is output to your console. For more information, see the [Java Example Code](#) in the AWS Encryption SDK.

Sample Code

```
import java.nio.file.Files;
import java.nio.file.Paths;
import java.security.KeyFactory;
import java.security.PrivateKey;
import java.security.PublicKey;
```

```

import java.security.spec.PKCS8EncodedKeySpec;
import java.security.spec.X509EncodedKeySpec;

import org.apache.commons.codec.binary.Base64;

import com.amazonaws.encryptionsdk.AwsCrypto;
import com.amazonaws.encryptionsdk.CryptoResult;
import com.amazonaws.encryptionsdk.jce.JceMasterKey;

/**
 * Sample example of decrypting data that has been encrypted by CloudFront Field-Level
Encryption.
 */
public class DecryptExample {

    private static final String PRIVATE_KEY_FILENAME = "private_key.der";
    private static final String PUBLIC_KEY_FILENAME = "public_key.der";
    private static PublicKey publicKey;
    private static PrivateKey privateKey;

    // CloudFront uses the following values to encrypt data, and your origin must use same
values to decrypt it.
    // In your own code, for PROVIDER_NAME, use the provider name that you specified when
you created your Field Level
    // Encryption Profile. This sample uses 'DEMO' for the value.
    private static final String PROVIDER_NAME = "DEMO";
    // In your own code, use the Key name that you specified when you added your public key
to CloudFront. This sample
    // uses 'DEMOKEY' for the Key name.
    private static final String KEY_NAME = "DEMOKEY";
    // Cloudfront uses this algorithm when encrypting data.
    private static final String ALGORITHM = "RSA/ECB/OAEPWithSHA-256AndMGF1Padding";

    public static void main(final String[] args) throws Exception {

        final String dataToDecrypt = args[0];

        // This sample uses files to get public and private keys.
        // In practice, you should distribute the public key and save the private key in
secure storage.
        populateKeyPair();

        System.out.println(debase64(dataToDecrypt));
    }

    private static String decrypt(final byte[] bytesToDecrypt) throws Exception {
        // You can decrypt the stream only by using the private key.

        // 1. Instantiate the SDK
        final AwsCrypto crypto = new AwsCrypto();

        // 2. Instantiate a JCE master key
        final JceMasterKey masterKey = JceMasterKey.getInstance(
            publicKey,
            privateKey,
            PROVIDER_NAME,
            KEY_NAME,
            ALGORITHM);

        // 3. Decrypt the data
        final CryptoResult <byte[], ? > result = crypto.decryptData(masterKey,
bytesToDecrypt);
        return new String(result.getResult());
    }

    // Function to decode base64 cipher text.
}

```

```
private static byte[] debase64(final String value) {
    return Base64.decodeBase64(value.getBytes());
}

private static void populateKeyPair() throws Exception {
    final byte[] PublicKeyBytes = Files.readAllBytes(Paths.get(PUBLIC_KEY_FILENAME));
    final byte[] privateKeyBytes = Files.readAllBytes(Paths.get(PRIVATE_KEY_FILENAME));
    publicKey = KeyFactory.getInstance("RSA").generatePublic(new
X509EncodedKeySpec(PublicKeyBytes));
    privateKey = KeyFactory.getInstance("RSA").generatePrivate(new
PKCS8EncodedKeySpec(privateKeyBytes));
}
```

Using AWS WAF to Control Access to Your Content

AWS WAF is a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to CloudFront, and lets you control access to your content. Based on conditions that you specify, such as the IP addresses that requests originate from or the values of query strings, CloudFront responds to requests either with the requested content or with an HTTP 403 status code (Forbidden). You can also configure CloudFront to return a custom error page when a request is blocked. For more information about AWS WAF, see the [AWS WAF Developer Guide](#).

After you create an AWS WAF web access control list (web ACL), you create or update a web distribution and associate the distribution with a web ACL. You can associate as many CloudFront distributions as you want with the same web ACL or with different web ACLs. For information about creating a web distribution and associating it with a web ACL, see [Creating or Updating a Web Distribution Using the CloudFront Console \(p. 60\)](#).

To associate or disassociate a web ACL and an existing distribution, or change the web ACL that is associated with a distribution, perform the following procedure.

To associate or disassociate an AWS WAF web ACL and an existing CloudFront distribution by using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the ID for the distribution that you want to update.
3. On the **General** tab, choose **Edit**.
4. On the **Distribution Settings** page, in the **AWS WAF Web ACL** list, choose the web ACL that you want to associate with this distribution.

If you want to disassociate the distribution from all web ACLs, choose **None**. If you want to associate the distribution with a different web ACL, choose the new web ACL.

5. Choose **Yes, Edit**.
6. Repeat steps 2 through 5 for other distributions, if any, for which you want to add, delete, or change associations with AWS WAF web ACLs.
7. After you change settings, the value of the **Status** column for the distributions that you updated changes to **InProgress** while CloudFront propagates the changes to edge locations. When **Status** changes to **Deployed** for a distribution, the distribution is ready to use AWS WAF when it processes requests. (The value of the **State** column for the distribution must also be **Enabled**.) This should take less than 15 minutes after you save the last change to a distribution.

Restricting the Geographic Distribution of Your Content

You can use *geo restriction*, also known as *geoblocking*, to prevent users in specific geographic locations from accessing content that you're distributing through a CloudFront web distribution. To use geo restriction, you have two options:

- Use the CloudFront geo restriction feature. Use this option to restrict access to all of the files that are associated with a distribution and to restrict access at the country level.
- Use a third-party geolocation service. Use this option to restrict access to a subset of the files that are associated with a distribution or to restrict access at a finer granularity than the country level.

Topics

- [Using CloudFront Geo Restriction \(p. 94\)](#)
- [Using a Third-Party Geolocation Service \(p. 95\)](#)

Using CloudFront Geo Restriction

When a user requests your content, CloudFront typically serves the requested content regardless of where the user is located. If you need to prevent users in specific countries from accessing your content, you can use the CloudFront geo restriction feature to do one of the following:

- Allow your users to access your content only if they're in one of the countries on a whitelist of approved countries.
- Prevent your users from accessing your content if they're in one of the countries on a blacklist of banned countries.

For example, if a request comes from a country where, for copyright reasons, you are not authorized to distribute your content, you can use CloudFront geo restriction to block the request.

Note

CloudFront determines the location of your users by using a third-party GeoIP database. The accuracy of the mapping between IP addresses and countries varies by region. Based on recent tests, the overall accuracy is 99.8%. Be aware that if CloudFront can't determine a user's location, CloudFront will serve the content that the user has requested.

Here's how geo restriction works:

1. Suppose you have rights to distribute your content only in Liechtenstein. You update your CloudFront web distribution and add a whitelist that contains only Liechtenstein. (Alternatively, you could add a blacklist that contains every country except Liechtenstein.)
2. A user in Monaco requests your content, and DNS routes the request to the CloudFront edge location in Milan, Italy.
3. The edge location in Milan looks up your distribution and determines that the user in Monaco is not allowed to download your content.
4. CloudFront returns an HTTP status code of 403 (Forbidden) to the user.

You can optionally configure CloudFront to return a custom error message to the user, and you can specify how long you want CloudFront to cache the error response for the requested object; the default value is five minutes. For more information, see [Customizing Error Responses \(p. 172\)](#).

Geo restriction applies to an entire web distribution. If you need to apply one restriction to part of your content and a different restriction (or no restriction) to another part of your content, you must either create separate CloudFront web distributions or use a third-party geolocation service.

If you enable CloudFront access logging, you can identify the requests that CloudFront rejected by searching for the log entries for which the value of `sc-status` (the HTTP status code) is 403. However, using only the access logs, you can't distinguish a request that CloudFront rejected based on the location of the user from a request that CloudFront rejected because the user didn't have permission to access the object for another reason. If you have a third-party geolocation service such as Digital Element or MaxMind, you can identify the location of requests based on the IP address in the `c-ip` (client IP) column in the access logs. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

The following procedure explains how to use the CloudFront console to add geo restriction to an existing web distribution. For information about how to use the console to create a web distribution, see [Working with Web Distributions \(p. 59\)](#).

To use the CloudFront console to add geo restriction to your CloudFront web distribution

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Select the distribution that you want to update.
3. In the **Distribution Settings** pane, choose the **Restrictions** tab.
4. Choose **Edit**.
5. Enter the applicable values. For more information, see [Restrictions \(p. 83\)](#).
6. Choose **Yes, Edit**.

Using a Third-Party Geolocation Service

The CloudFront geo restriction feature lets you control distribution of your content at the country level for all files that you're distributing with a given web distribution. If you have geographic restrictions on where your content can be distributed and the restrictions don't follow country boundaries, or if you want to limit access to only some of the files that you're distributing through CloudFront, you can combine CloudFront with a third-party geolocation service. This can allow you to control access to your content based not only on country but also based on city, zip or postal code, or even latitude and longitude.

When you're using a third-party geolocation service, we recommend that you use CloudFront signed URLs, which let you specify an expiration date and time after which the URL is no longer valid. In addition, we recommend that you use an Amazon S3 bucket as your origin because you can then use a CloudFront origin access identity to prevent users from accessing your content directly from the origin. For more information about signed URLs and origin access identities, see [Serving Private Content through CloudFront \(p. 207\)](#).

The following task list explains how to control access to your files by using a third-party geolocation service.

Task list for restricting access to files in a CloudFront distribution based on geographic location

1. Get an account with a geolocation service.
2. Upload your content to an Amazon Simple Storage Service (S3) bucket. For more information, see the [Amazon S3 documentation](#).
3. Configure Amazon CloudFront and Amazon S3 to serve private content. For more information, see [Serving Private Content through CloudFront \(p. 207\)](#).
4. Write your web application to do the following:

- a. Send the IP address for each user request to the geolocation service.
- b. Evaluate the return value from the geolocation service to determine whether the user is in a location to which you want CloudFront to distribute your content.
- c. Based on whether you want to distribute your content to the user's location, either generate a signed URL for your CloudFront content, or return HTTP status code 403 (Forbidden) to the user. Alternatively, you can configure CloudFront to return a custom error message. For more information, see [Customizing Error Responses \(p. 172\)](#).

For more information, refer to the documentation for the geolocation service that you're using.

You can use a web server variable to get the IP addresses of the users who are visiting your website. Note the following caveats:

- If your web server is not connected to the internet through a load balancer, you can use a web server variable to get the remote IP address. However, this IP address isn't always the user's IP address—it can also be the IP address of a proxy server, depending on how the user is connected to the internet.
- If your web server is connected to the internet through a load balancer, a web server variable might contain the IP address of the load balancer, not the IP address of the user. In this configuration, we recommend that you use the last IP address in the `X-Forwarded-For` http header. This header typically contains more than one IP address, most of which are for proxies or load balancers. The last IP address in the list is the one most likely to be associated with the user's geographic location.

If your web server is not connected to a load balancer, we recommend that you use web server variables instead of the `X-Forwarded-For` header to avoid IP address spoofing.

Configuring On-Demand Smooth Streaming

You can use CloudFront for on-demand streaming of media files that you've transcoded into the Microsoft Smooth Streaming format. To distribute Smooth Streaming content on demand, you have two options:

- As the origin for your distribution, specify a web server that can stream files that have been transcoded into Microsoft Smooth Streaming format.
- Enable Smooth Streaming in a CloudFront distribution. Smooth Streaming is a property of cache behaviors, which means that you can use one distribution to distribute Smooth Streaming media files as well as other content.

If you enable Smooth Streaming, note the following:

- You can still distribute other content using the same cache behavior if the content matches the value of **Path Pattern** for that cache behavior.
- CloudFront can use either an Amazon S3 bucket or a custom origin for Smooth Streaming media files. However, CloudFront cannot use a Microsoft IIS Server as an origin if the server is configured for Smooth Streaming.
- You cannot invalidate media files in the Smooth Streaming format. If you want to update files before they expire, you must rename them. For more information, see [Adding, Removing, or Replacing Objects in a Distribution \(p. 158\)](#).

For information about Smooth Streaming clients, see [Smooth Streaming Primer](#) on the Microsoft website.

To use CloudFront to stream media files that have been encoded in the Microsoft Smooth Streaming format without using a web server that can stream files in Smooth Streaming format, perform the following tasks:

1. Transcode your media files into Smooth Streaming fragmented-MP4 format. For a list of applications that can transcode into Smooth Streaming format, see [Smooth Streaming Primer](#) on the Microsoft website.
2. Do one of the following:
 - **If you're using the CloudFront console:** When you create a web distribution, enable Smooth Streaming in the default cache behavior. Alternatively, you can enable Smooth Streaming in the default cache behavior and/or one or more custom cache behaviors in an existing CloudFront web distribution.
 - **If you're using the CloudFront API:** Add the SmoothStreaming element to the DistributionConfig complex type for the default cache behavior and/or one or more custom cache behaviors.
3. Upload the files in your Smooth Streaming presentations to the applicable origin.
4. Create either a clientaccesspolicy.xml or a crossdomainpolicy.xml file, and add it to a location that is accessible at the root of your distribution, for example, `http://d111111abcdef8.cloudfront.net/clientaccesspolicy.xml`. The following is an example policy:

```
<?xml version="1.0" encoding="utf-8"?>
<access-policy>
<cross-domain-access>
<policy>
<allow-from http-request-headers="*"/>
<domain uri="*"/>
</allow-from>
<grant-to>
<resource path="/" include-subpaths="true"/>
</grant-to>
</policy>
</cross-domain-access>
</access-policy>
```

For more information, see [Making a Service Available Across Domain Boundaries](#) on the Microsoft Developer Network website.

5. For links in your application, specify the client manifest in the following format:

`http://d111111abcdef8.cloudfront.net/video/presentation.ism/Manifest`

Configuring On-Demand Progressive Downloads

To use CloudFront to distribute media files using progressive download, perform the following tasks:

1. Transcode your media files, if applicable.
2. Create a CloudFront web distribution. No special settings are required.
3. Upload your files to the origin that you specified when you created your distribution.
4. For links in your application (for example, a media player), specify the name of the media file in the same format that you use for other objects that you're distributing using CloudFront. For more information, see [Format of URLs for Objects \(p. 143\)](#).

Configuring On-Demand Apple HTTP Live Streaming (HLS)

To use CloudFront to stream media files in Apple HTTP Live Streaming (HLS) format, perform the following tasks:

1. Transcode your media files into HLS format. You can use any transcoding application that supports HLS format, for example, [Amazon Elastic Transcoder](#).
2. Create a CloudFront web distribution. No special settings are required.
3. Upload your files to the origin that you specified when you created your distribution.
4. For links in your application (for example, a media player), specify the master playlist in the same format that you use for other objects that you're distributing using CloudFront. For more information, see [Format of URLs for Objects \(p. 143\)](#).

Using CloudFront with Lambda@Edge

Lambda@Edge is an extension of AWS Lambda, a compute service that lets you execute functions that customize the content that is delivered through CloudFront. You can author functions in one region and execute them in AWS Regions and edge locations globally without provisioning or managing servers. Lambda@Edge scales automatically, from a few requests per day to thousands per second. Processing requests at edge locations instead of on origin servers significantly reduces latency and improves the user experience.

When you associate a CloudFront distribution with a Lambda@Edge function, CloudFront intercepts requests and responses at CloudFront edge locations. Lambda@Edge functions execute in response to CloudFront events in the region or edge location that is closest to your customer.

You can execute Lambda functions when the following CloudFront events occur:

- When CloudFront receives a request from a viewer (viewer request)
- Before CloudFront forwards a request to the origin (origin request)
- When CloudFront receives a response from the origin (origin response)
- Before CloudFront returns the response to the viewer (viewer response)

There are many uses for Lambda@Edge processing. Here are just a few possibilities:

- You can write Lambda functions that inspect cookies and rewrite URLs so that users see different versions of a site for A/B testing.
- CloudFront can return different objects to viewers based on the `User-Agent` header, which includes information about the devices that users are using to view your content. For example, CloudFront can return different images based on the screen size of devices. Similarly, the function could consider the value of the `Referer` header and cause CloudFront to return the images that have the lowest available resolution to bots.

You might also want to consider additional criteria. For example, on a retail website that sells clothing, if you use cookies to indicate which color a user chose for a jacket, the function can change the request so that CloudFront returns the image of a jacket in the selected color.

- You can use a Lambda function to generate HTTP responses when CloudFront viewer request or origin request events occur.
- A Lambda function can inspect headers or authorization tokens, and insert the applicable header to control access to your content before CloudFront forwards a request to the origin.
- A Lambda function can add, drop, and modify headers, and can rewrite URL paths so that CloudFront returns different objects.
- A Lambda function can make network calls to external resources to confirm user credentials, or fetch additional content to customize your responses.

For more information about setting up CloudFront with Lambda@Edge, see [Lambda@Edge](#) in the *AWS Lambda Developer Guide*. After you complete the setup steps, see the following topics to learn how to use CloudFront with Lambda@Edge.

Topics

- [CloudFront Events That Can Trigger a Lambda Function \(p. 100\)](#)

- [How You Decide Which CloudFront Event to Use to Trigger a Lambda Function \(p. 101\)](#)
- [Requirements and Restrictions on Lambda Functions \(p. 101\)](#)
- [Creating Lambda Functions and Adding Triggers \(p. 107\)](#)
- [Event Structure \(p. 108\)](#)
- [Generating HTTP Responses in Request Triggers \(p. 114\)](#)
- [Updating HTTP Responses in an Origin-Response Trigger \(p. 116\)](#)
- [CloudWatch Metrics and CloudWatch Logs for Lambda Functions \(p. 117\)](#)
- [Example Functions \(p. 118\)](#)

CloudFront Events That Can Trigger a Lambda Function

For each cache behavior in a CloudFront distribution, you can add up to four triggers (associations) that cause a Lambda function to execute when one or more of the following CloudFront events occur. When you add multiple triggers to the same cache behavior, you can use the same function or different functions for the triggers. You can also use the same function for more than one distribution.

Note

When a CloudFront event triggers the execution of a Lambda function, the function must complete before CloudFront continues. For example, if you associate a Lambda function with a CloudFront viewer request event, CloudFront won't return a response to the viewer or forward the request to the origin until the Lambda function completes. Each request that triggers a Lambda function increases latency for the request, so you'll want the function to execute as fast as possible.

CloudFront Viewer Request

The function executes when CloudFront receives a request from a viewer and before it checks to see whether the requested object is in the edge cache.

CloudFront Origin Request

The function executes only when CloudFront forwards a request to your origin. When the requested object is in the edge cache, the function doesn't execute.

CloudFront Origin Response

The function executes after CloudFront receives a response from the origin and before it caches the object in the response. The function will execute even if an error is returned from the origin.

The function doesn't execute in the following cases:

- When the requested object is in the edge cache
- When the response is generated from a function that was triggered by an origin request event

CloudFront Viewer Response

The function executes before returning the requested object to the viewer. The function executes regardless of whether the object is already in the edge cache.

The function doesn't execute in the following cases:

- When the origin returns an HTTP status code of 400 or higher
- When a custom error page is returned
- When the response is generated from a function that was triggered by a viewer request event
- When CloudFront automatically redirects an HTTP request to HTTPS (when the value of [Viewer Protocol Policy \(p. 72\)](#) is [Redirect HTTP to HTTPS](#))

How You Decide Which CloudFront Event to Use to Trigger a Lambda Function

When you're deciding which CloudFront event you want to use to trigger a Lambda function, consider the following:

Do you want CloudFront to cache objects that are changed by a Lambda function?

If you want CloudFront to cache an object that was modified by a Lambda function so that CloudFront can serve the object from the edge location the next time it's requested, use the origin request or origin response event. This reduces the load on the origin, reduces latency for subsequent requests, and reduces the cost of invoking Lambda@Edge on subsequent requests.

For example, if you want to add, remove, or change headers for objects that are returned by the origin and you want CloudFront to cache the result, use the origin response event.

Do you want the function to execute for every request?

If you want the function to execute for every request that CloudFront receives for the distribution, use the viewer request or viewer response events. Origin request and origin response events occur only when a requested object isn't cached in an edge location and CloudFront forwards a request to the origin.

Does the function change the cache key?

If you want the function to change a value that you're using as a basis for caching, use the viewer request event. For example, if a function changes the URL to include a language abbreviation in the path (for example, because the user chose their language from a dropdown list), use the viewer request event:

- **URL in the viewer request** – <http://example.com/en/index.html>
- **URL when the request comes from an IP address in Germany** – <http://example.com/de/index.html>

You also use the viewer request event if you're caching based on cookies or request headers.

Note

If the function changes cookies or headers, configure CloudFront to forward the applicable part of the request to the origin. For more information, see the following topics:

- [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#)
- [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#)

Does the function affect the response from the origin?

If you want the function to change the request in a way that affects the response from the origin, use the origin request event. Typically, most viewer request events aren't forwarded to the origin; CloudFront responds to a request with an object that's already in the edge cache. If the function changes the request based on an origin request event, CloudFront caches the response to the changed origin request.

Requirements and Restrictions on Lambda Functions

See the following sections for requirements and restrictions on using Lambda functions with CloudFront.

Topics

- [CloudFront Distributions and Associations \(p. 102\)](#)
- [CloudFront Triggers for Lambda Functions \(p. 102\)](#)
- [CloudWatch Logs \(p. 102\)](#)
- [Headers \(p. 102\)](#)
- [HTTP Status Codes \(p. 105\)](#)
- [Lambda Function Configuration and Execution Environment \(p. 105\)](#)
- [Limits \(p. 105\)](#)
- [Microsoft Smooth Streaming \(p. 105\)](#)
- [Network Access \(p. 105\)](#)
- [Query String Parameters \(p. 105\)](#)
- [URI \(p. 106\)](#)
- [URI and Query String Encoding \(p. 106\)](#)

CloudFront Distributions and Associations

- You can create triggers (associations) for Lambda functions for a maximum of 25 distributions per AWS account.
- You can create a maximum of 100 triggers (associations) for a distribution.

CloudFront Triggers for Lambda Functions

- You can add triggers only for a numbered version, not for \$LATEST or for aliases.
- You can add triggers only for functions in the US East (N. Virginia) Region.
- To add triggers, the IAM execution role associated with your Lambda function must be assumable by the service principals lambda.amazonaws.com and edgelambda.amazonaws.com. For more information, see [Setting IAM Permissions and Roles for Lambda@Edge](#) in the *IAM User Guide*.

CloudWatch Logs

For information about Amazon CloudWatch Logs limits, see [CloudWatch Logs Limits](#) in the *Amazon CloudWatch User Guide*.

Headers

Note the following requirements and restrictions on using headers with Lambda@Edge.

Topics

- [Blacklisted Headers \(p. 102\)](#)
- [Read-only Headers \(p. 103\)](#)
- [Restricted Headers for CloudFront Origin Request Events \(p. 104\)](#)
- [CloudFront-* Headers \(p. 104\)](#)

Blacklisted Headers

Blacklisted headers aren't exposed and can't be added by Lambda@Edge functions. If your Lambda function adds a blacklisted header, the request fails CloudFront validation. CloudFront returns HTTP status code 502 (Bad Gateway) to the viewer.

- Connection
- Expect
- Keep-alive
- Proxy-Authenticate
- Proxy-Authorization
- Proxy-Connection
- Trailer
- Upgrade
- X-Accel-Buffering
- X-Accel-Charset
- X-Accel-Limit-Rate
- X-Accel-Redirect
- X-Amz-Cf-*
- X-Amzn-*
- X-Cache
- X-Edge-*
- X-Forwarded-Proto
- X-Real-IP

Read-only Headers

Read-only headers can be read but not edited. You can use them as input to CloudFront caching logic, and your Lambda function can read the header values, but it can't change the values. If your Lambda function adds or edits a read-only header, the request fails CloudFront validation. CloudFront returns HTTP status code 502 (Bad Gateway) to the viewer.

Read-only Headers for CloudFront Viewer Request Events

- Content-Length
- Host
- Transfer-Encoding
- Via

Read-only Headers for CloudFront Origin Request Events

- Accept-Encoding
- Content-Length
- If-Modified-Since
- If-None-Match
- If-Range
- If-Unmodified-Since
- Range
- Transfer-Encoding
- Via

Read-only Headers for CloudFront Origin Response Events

- Content-Encoding

- Content-Length
- Transfer-Encoding
- Via

Read-only Headers for CloudFront Viewer Response Events

- Content-Encoding
- Content-Length
- Transfer-Encoding
- Warning
- Via

Restricted Headers for CloudFront Origin Request Events

By default, CloudFront removes the following headers from viewer requests. You can add or edit these headers in CloudFront origin request events only if the CloudFront distribution is configured to cache based on the headers. This causes CloudFront to forward the headers to your origin instead of removing them.

If a Lambda function adds or changes a restricted header and the CloudFront distribution is not configured to cache based on that header, the request fails CloudFront validation. CloudFront returns HTTP status code 502 (Bad Gateway) to the viewer.

For information about how to configure CloudFront to cache based on specified headers, see [Cache Based on Selected Request Headers \(p. 73\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

- Accept
- Accept-Charset
- Accept-Language
- Authorization
- Referer
- TE

CloudFront-* Headers

A Lambda function can read, edit, remove, or add any of the following headers.

- CloudFront-Forwarded-Proto
- CloudFront-Is-Desktop-Viewer
- CloudFront-Is-Mobile-Viewer
- CloudFront-Is-SmartTV-Viewer
- CloudFront-Is-Tablet-Viewer
- CloudFront-Viewer-Country

Note the following:

- If you want CloudFront to add these headers, you must configure CloudFront to cache based on these headers. For information about configuring CloudFront to cache based on specified headers, see [Cache Based on Selected Request Headers \(p. 73\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

- CloudFront adds the headers after the viewer request event.
- If the viewer adds headers that have these names, CloudFront overwrites the header values.
- For viewer events, CloudFront-Viewer-Country is blacklisted. Blacklisted headers aren't exposed and can't be added by Lambda@Edge functions. If your Lambda function adds a blacklisted header, the request fails CloudFront validation, and CloudFront returns HTTP status code 502 (Bad Gateway) to the viewer.

For more information, see the following examples:

- [Example: Redirecting Viewer Requests to a Country-Specific URL \(p. 123\)](#)
- [Example: Serving Different Versions of an Object Based on the Device \(p. 124\)](#)

HTTP Status Codes

CloudFront doesn't execute Lambda functions for viewer response events if the origin returns HTTP status code 400 or higher.

Lambda Function Configuration and Execution Environment

- You must create functions with the `nodejs6.10` runtime property.
- You can't configure your Lambda function to access resources inside your VPC.
- The Dead Letter Queue (DLQ) isn't supported.
- Environment variables aren't supported.

Limits

For information about limits, see the following documentation:

- [Limits on Lambda@Edge \(p. 362\)](#) in this guide
- [AWS Lambda Limits in the AWS Lambda Developer Guide](#)

Microsoft Smooth Streaming

You can't create triggers for a CloudFront distribution that you're using for on-demand streaming of media files that you've transcoded into the Microsoft Smooth Streaming format.

Network Access

Functions triggered by origin request and response events as well as functions triggered by viewer request and response events can make network calls to resources on the internet, and to services in AWS regions such as Amazon S3 buckets, DynamoDB tables, or Amazon EC2 instances.

Query String Parameters

- To access a query string in a Lambda function, use `event.Records[0].cf.request.querystring`.
- A function can update a query string for viewer and origin request events. The updated query string can't include spaces, control characters, or the fragment identifier (#).

- A function can read a query string only for origin and viewer response events.
- Configuring CloudFront to cache based on query string parameters affects whether a function can access the query string:
 - **Viewer request and response events** – A function can access a query string regardless of the setting for [Query String Forwarding and Caching \(p. 76\)](#).
 - **Origin request and response events** – A function can access a query string only if [Query String Forwarding and Caching \(p. 76\)](#) is set either to **Forward All, Cache Based on Whitelist** or to **Forward All, Cache Based on All**.
- We recommend that you use percent encoding for the URI and query string. For more information, see [URI and Query String Encoding \(p. 106\)](#).
- The total size of the URI (`event.Records[0].cf.request.uri`) and the query string (`event.Records[0].cf.request.querystring`) must be less than 8,192 characters.

For more information, see [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#).

URI

If a function changes the URI for a request, that doesn't change the cache behavior for the request or the origin that the request is forwarded to.

URI and Query String Encoding

Lambda functions require the URI and query string to be UTF-8 encoded. (Percent encoding is compatible with UTF-8 encoding.) The behavior of CloudFront and Lambda depends on the following:

- The encoding of the URI and query string that CloudFront received in the request from the viewer
- Whether the URI or query string is changed by a function that is triggered by a viewer request or origin request event

Values Are UTF-8 Encoded

CloudFront forwards the values to your Lambda function without changing them.

Values Are ISO 8859-1 Encoded

CloudFront converts [ISO 8859-1 character encoding](#) to UTF-8 encoding before forwarding the values to your Lambda function.

Values Are Encoded Using Some Other Character Encoding

If the values are encoded using any other character encoding, CloudFront assumes that they're ISO 8859-1 encoded and tries to convert from ISO 8859-1 encoding to UTF-8 encoding.

Important

The converted version might be an inaccurate interpretation of the values in the original request. This can cause a Lambda function or your origin to produce an unintended result.

The value that CloudFront forwards to your origin server depends on whether functions that are triggered by viewer request or origin request events change the URI or query string:

- **If the functions don't change the URI or query string** – CloudFront forwards the values that CloudFront received in the request from the viewer to your origin server.
- **If the functions do change the URI or query string** – CloudFront forwards the UTF-8 encoded value.

In both cases, the behavior is unaffected by the character encoding of the request from the viewer.

Creating Lambda Functions and Adding Triggers

To use Lambda@Edge, you create Lambda functions and add triggers. If you use the AWS console, note the following:

- You can use only the AWS Lambda console to create Lambda functions.
- You can use either the AWS Lambda console or the CloudFront console to add triggers.

You can also perform all operations programmatically. For more information, see the following documentation:

- [API Reference](#) in the *AWS Lambda Developer Guide*
- [Amazon CloudFront API Reference](#)
- **AWS CLI**
 - [Lambda create-function command](#)
 - [CloudFront create-distribution command](#)
 - [CloudFront create-distribution-with-tags command](#)
 - [CloudFront update-distribution command](#)
- [AWS SDKs](#) (See the **SDKs & Toolkits** section.)
- [AWS Tools for Windows PowerShell Reference](#)

Creating Lambda Functions (Console)

For information about creating Lambda functions in the AWS Lambda console, see [AWS Lambda@Edge](#) in the *AWS Lambda Developer Guide*.

Adding Triggers for CloudFront Events to a Lambda Function

You can add triggers for CloudFront events with either the AWS Lambda console or the CloudFront console.

Topics

- [Adding Triggers for CloudFront Events \(AWS Lambda Console\) \(p. 107\)](#)
- [Adding Triggers for CloudFront Events \(CloudFront Console\) \(p. 107\)](#)

Adding Triggers for CloudFront Events (AWS Lambda Console)

For information about adding triggers for CloudFront events in the Lambda console, see [AWS Lambda@Edge](#) in the *AWS Lambda Developer Guide*.

Adding Triggers for CloudFront Events (CloudFront Console)

To add triggers for CloudFront events in the CloudFront console, perform the following procedure.

To add triggers for CloudFront events to a Lambda function (CloudFront console)

1. Get the ARN of the Lambda function that you want to add triggers for:

- a. Sign in to the AWS Management Console and open the AWS Lambda console at <https://console.aws.amazon.com/lambda/>.
- b. In the list of regions at the top of the page, choose **US East (N. Virginia)**.
- c. In the list of functions, choose name of the function that you want to add triggers to.
- d. Choose **Qualifiers**, choose the **Versions** tab, and choose the numbered version that you want to add triggers to.

Important

You can add triggers only to a numbered version, not to **\$LATEST**.

- e. Copy the ARN that appears at the top of the page, for example:

```
arn:aws:lambda:us-east-1:123456789012:function:TestFunction:2
```

The number at the end (**2** in this example) is the version number of the function.

2. Open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
3. In the list of distributions, choose the ID of the distribution that you want to add triggers to.
4. Choose the **Behaviors** tab.
5. Select the check box for the cache behavior that you want to add triggers to, and then choose **Edit**.
6. At **Lambda Function Associations**, in the **Event Type** list, choose when you want the function to execute: for viewer requests, viewer responses, origin requests, or origin responses.

For more information, see [How You Decide Which CloudFront Event to Use to Trigger a Lambda Function \(p. 101\)](#).

7. Paste the ARN of the Lambda function that you want to execute when the chosen event occurs. This is the value that you copied in step 1.
8. To execute the same function for more event types, choose **+** and repeat steps 6 and 7.
9. Choose **Yes, Edit**.
10. To add triggers to more cache behaviors for this distribution, repeat steps 5 through 9.

The function starts to process requests for the specified CloudFront events when the updated CloudFront distribution is deployed. To determine whether a distribution is deployed, choose **Distributions** in the navigation pane. When a distribution is deployed, the value of the **Status** column for the distribution changes from **In Progress** to **Deployed**.

Event Structure

The examples in this section show request and response events that CloudFront passes to or returns from a Lambda@Edge function when it's triggered.

You can use the path pattern in a cache behavior to route requests to an origin, based on the path and name of the requested object, such as `images/*.jpg`. Using Lambda@Edge, you can also route requests to an origin based on other characteristics, such as the values in request headers.

There are a number of ways this content-based dynamic origin selection can be useful. For example, you can distribute requests across origins in different geographic areas to help with global load balancing. Or you can selectively route requests to different origins that each serve a particular function: bot handling, SEO optimization, authentication, and so on. For code samples that demonstrate how to use this feature, see [Example Functions \(p. 118\)](#).

In the CloudFront origin request event, the `origin` element in the event structure contains information about the Amazon S3 or custom origin that the request would be routed to, based on the path pattern. You can update the values in the `origin` object to route a request to a different origin. The `origin` doesn't

need to be defined in the distribution, and you can replace an Amazon S3 origin object with a custom origin object and vice versa.

You can specify either a custom origin or an Amazon S3 origin in a single request, but not both.

Topics

- [Request Event \(p. 109\)](#)
- [Response Event \(p. 112\)](#)

Request Event

Here's the format of the event object that CloudFront passes to a Lambda function when the function is triggered by a CloudFront viewer request event or an origin request event:

```
{  
    "Records": [  
        {  
            "cf": {  
                "config": {  
                    "distributionId": "EDFDVBD6EXAMPLE",  
                    "requestId": "MRVMF7KydiIvxMwfJIglgwHQwZsbG2IhRJ07sn9AkKUFSHS9EXAMPLE=="  
                },  
                "request": {  
                    "clientIp": "2001:0db8:85a3:0:0:8a2e:0370:7334",  
                    "queryString": "size=large",  
                    "uri": "/picture.jpg",  
                    "method": "GET",  
                    "headers": {  
                        "host": [  
                            {  
                                "key": "Host",  
                                "value": "d111111abcdef8.cloudfront.net"  
                            }  
,  
                        "user-agent": [  
                            {  
                                "key": "User-Agent",  
                                "value": "curl/7.51.0"  
                            }  
,  
                        ],  
                    },  
                    "origin": {  
                        "custom": {  
                            "customHeaders": {  
                                "my-origin-custom-header": [  
                                    {  
                                        "key": "My-Origin-Custom-Header",  
                                        "value": "Test"  
                                    }  
,  
                                ],  
                            },  
                            "domainName": "example.com",  
                            "keepaliveTimeout": 5,  
                            "path": "/custom_path",  
                            "port": 443,  
                            "protocol": "https",  
                            "readTimeout": 5,  
                            "sslProtocols": [  
                                "TLSv1",  
                                "TLSv1.1"  
,  
                            ],  
                        },  
                    },  
                },  
            }  
,  
        }  
,  
    ]  
,  
},  
}
```

```
        "s3": {
            "authMethod": "origin-access-identity",
            "customHeaders": [
                {
                    "key": "My-Origin-Custom-Header",
                    "value": "Test"
                }
            ],
            "domainName": "my-bucket.s3.amazonaws.com",
            "path": "/s3_path",
            "region": "us-east-1"
        }
    }
}
```

Request events include the following values:

Config Values

distributionID (read-only)

The ID of the distribution that's associated with the request.

requestId (read-only, viewer request events only)

An encrypted string that uniquely identifies a request. The `requestId` value also appears in CloudFront access logs as `x-edge-request-id`. For more information, see [Access Logs \(p. 315\)](#) and [Web Distribution Log File Format \(p. 319\)](#).

Request Values - General

clientIp (read-only)

The IP address of the viewer that made the request. If the viewer used an HTTP proxy or a load balancer to send the request, the value is the IP address of the proxy or load balancer.

headers (read/write)

The headers in the request. The keys in the `headers` object are lowercase versions of the header names in the HTTP request. These lowercase keys give you case-insensitive access to the header values. Each header (for example, `headers["accept"]` or `headers["host"]`) is an array of key-value pairs, where `key` is the case-sensitive name of the header as it appears in an HTTP request and `value` is a header value. The number of elements in the array is the number of times that a header is repeated in an HTTP request.

For information about restrictions on header usage, see [Headers \(p. 102\)](#).

method (read-only)

The HTTP method of the viewer request.

querystring

The query string, if any, that CloudFront received in the viewer request. If the viewer request doesn't include a query string, the event structure still includes `querystring` with an empty value. For more information about query strings, see [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#).

uri (read/write)

The relative path of the requested object. Note the following:

- The new relative path must begin with a slash (like this: /).
- If a function changes the URI for a request, that changes the object that the viewer is requesting.
- If a function changes the URI for a request, that doesn't change the cache behavior for the request or the origin that the request is forwarded to.

Request Values - Custom Origin

You can specify either a custom origin or an Amazon S3 origin in a single request; not both.

customHeaders

You can include custom headers with the request by specifying a header name and value pair for each custom header. You can't add headers that are blacklisted for origin custom headers or hooks, and a header with the same name can't be present in `request.headers` or in `request.origin.customHeaders`. The restrictions for `request.headers` also apply to custom headers. For more information, see [Custom Headers that CloudFront Can't Forward to Your Origin \(p. 157\)](#) and [Blacklisted Headers \(p. 102\)](#).

domainName

The domain name of the origin server, like `www.example.com`. The domain name can't be empty, can't include a colon (:), and can't use the IPV4 address format. The domain name can be up to 253 characters.

keepaliveTimeout

How long, in seconds, that CloudFront should try to maintain the connection to your origin after receiving the last packet of a response. The value must be a number in the range of 1 to 60 seconds.

path

The directory path at the server where the request should locate content. The path should start with a slash (/) but should have no trailing / (like `path/`). The path should be URL encoded, with a maximum length of 255 characters.

port

The port at your custom origin. The port must be 80 or 443, or a number in the range of 1024 to 65535.

protocol

The origin protocol policy that CloudFront should use when fetching objects from your origin. The value can be `http` or `https`.

readTimeout

How long, in seconds, CloudFront should wait for a response after forwarding a request to the origin, and how long CloudFront should wait after receiving a packet of a response before receiving the next packet. The value must be a number in the range of 4 to 60 seconds.

sslProtocols

The SSL protocols that CloudFront can use when establishing an HTTPS connection with your origin. Values can be the following: `TLSv1.2`, `TLSv1.1`, `TLSv1`, `SSLv3`.

Request Values - Amazon S3 Origin

You can specify either a custom origin or an Amazon S3 origin in a single request; not both.

authMethod

Set to `origin-access-identity` if your Amazon S3 bucket has an origin access identity (OAI) set up, or `none` if you aren't using OAI. If you set `authMethod` to `origin-access-identity`, there are several requirements:

- You must specify a region in your header.
 - You must use the same OAI when you switch from one Amazon S3 origin to another.
 - You can't use an OAI when you switch from a custom origin to an Amazon S3 origin.

For more information about using an OAI, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content](#) (p. 211).

customHeaders

You can include custom headers with the request by specifying a header name and value pair for each custom header. You can't add headers that are blacklisted for origin custom headers or hooks, and a header with the same name can't be present in `request.headers` or in `request.origin.custom.customHeaders`. The restrictions for `request.headers` also apply to custom headers. For more information, see [Custom Headers that CloudFront Can't Forward to Your Origin \(p. 157\)](#) and [Blacklisted Headers \(p. 102\)](#).

domainName

The domain name of the Amazon S3 origin server, like `my-bucket.s3.amazonaws.com`. The domain name can't be empty, and must be an allowed bucket name (as defined by Amazon S3). The name can be up to 128 characters, and must be all lowercase.

path

The directory path at the server where the request should locate content. The path should start with a slash (/) but should have no trailing / (like path/).

region

The region for your Amazon S3 bucket. This is required only if you use OAI.

Response Event

Here's the format of the event object that CloudFront passes to a Lambda function when the function is triggered by a CloudFront viewer response event or an origin response event:

```
{
    "Records": [
        {
            "cf": {
                "config": {
                    "distributionId": "EDFDVBD6EXAMPLE",
                    "requestId": "xGN7KWpVEmB9Dp7ctcVFQC4E-nrcOcEKS3QyAez--06dV7TEXAMPLE=="
                },
                "request": {
                    "clientIp": "2001:0db8:85a3:0:0:8a2e:0370:7334",
                    "method": "GET",
                    "uri": "/picture.jpg",
                    "queryString": "size=large",
                    "headers": {
                        "host": [
                            {
                                "key": "Host",
                                "value": "d111111abcdef8.cloudfront.net"
                            }
                        ],
                    }
                }
            }
        }
    ]
}
```

```

        "user-agent": [
            {
                "key": "User-Agent",
                "value": "curl/7.18.1"
            }
        ],
        "response": {
            "status": "200",
            "statusDescription": "OK",
            "headers": {
                "server": [
                    {
                        "key": "Server",
                        "value": "MyCustomOrigin"
                    }
                ],
                "set-cookie": [
                    {
                        "key": "Set-Cookie",
                        "value": "theme=light"
                    },
                    {
                        "key": "Set-Cookie",
                        "value": "sessionToken=abc123; Expires=Wed, 09 Jun 2021
10:18:14 GMT"
                    }
                ]
            }
        }
    }
}

```

Response events include the values that appear in the corresponding request event, and in addition, the following values. If your Lambda@Edge function generates an HTTP response, see [Generating HTTP Responses in Request Triggers \(p. 114\)](#).

distributionID (read-only)

The ID of the distribution that's associated with the request.

headers

Headers that you want CloudFront to return in the generated response. Note the following:

- The keys in the `headers` object are lowercase versions of standard HTTP header names. Using lowercase keys gives you case-insensitive access to the header values.
- Each header (for example, `headers["accept"]` or `headers["host"]`) is an array of key-value pairs. For a given header, the array contains one key-value pair for each value in the generated response.
- Specify `key` as the case-sensitive name of the header as it appears in an HTTP request; for example, `accept` or `host`.
- Specify `value` as a header value.

For information about restrictions on header usage, see [Headers \(p. 102\)](#).

requestId (read-only, viewer response events only)

An encrypted string that uniquely identifies a request. The `requestId` value also appears in CloudFront access logs as the `x-edge-request-id`. For more information, see [Access Logs \(p. 315\)](#) and [Web Distribution Log File Format \(p. 319\)](#).

request – One of the following:

- Viewer response – The request that CloudFront received from the viewer and that might have been modified by the Lambda function that was triggered by a viewer request event
- Origin response – The request that CloudFront forwarded to the origin and that might have been modified by the Lambda function that was triggered by an origin request event

If the Lambda function modifies the request object, the changes are ignored.

response – One of the following:

- Viewer response – The response that CloudFront will return to the viewer for viewer response events.
- Origin response – The response that CloudFront received from the origin for origin response events.

status

The HTTP status code that CloudFront returns to the viewer.

statusDescription

The HTTP status description that CloudFront returns to the viewer.

Generating HTTP Responses in Request Triggers

When CloudFront receives a request, you can use a Lambda function to generate an HTTP response that CloudFront returns directly to the viewer without forwarding the response to the origin. Generating HTTP responses reduces the load on the origin, and typically also reduces latency for the viewer.

Some common scenarios for generating HTTP responses include the following:

- Returning a small web page to the viewer
- Returning an HTTP 301 or 302 status code to redirect the user to another web page
- Returning an HTTP 401 status code to the viewer when the user hasn't authenticated

A Lambda@Edge function can generate an HTTP response when the following CloudFront events occur:

Viewer request events

When a function is triggered by a viewer request event, CloudFront returns the response to the viewer and doesn't cache it.

Origin request events

When a function is triggered by an origin request event, CloudFront checks the edge cache for a response that was previously generated by the function.

- If the response is in the cache, the function isn't executed and CloudFront returns the cached response to the viewer.
- If the response isn't in the cache, the function is executed, CloudFront returns the response to the viewer, and also caches it.

For some examples of generating HTTP responses, see [Example Functions \(p. 118\)](#).

Programming Model

This section describes the programming model for using Lambda@Edge to generate HTTP responses.

Topics

- [Response Object \(p. 115\)](#)
- [Errors \(p. 116\)](#)
- [Required Fields \(p. 116\)](#)

Response Object

The response you return as the `result` parameter of the `callback` method should have the following structure (note that only the `status` field is required).

```
const response = {
    body: 'content',
    bodyEncoding: 'text' | 'base64',
    headers: {
        'header name in lowercase': [
            {key: 'header name in standard case',
             value: 'header value'}
        ],
        ...
    },
    status: 'HTTP status code',
    statusDescription: 'status description'
};
```

The response object can include the following values:

body

The body, if any, that you want CloudFront to return in the generated response.

bodyEncoding

The encoding for the value that you specified in the body. The only valid encodings are `text` and `base64`. If you include `body` in the `response` object but omit `bodyEncoding`, CloudFront treats the body as `text`.

If you specify `bodyEncoding` as `base64` but the body is not valid `base64`, CloudFront returns an error.

headers

Headers that you want CloudFront to return in the generated response. Note the following:

- The keys in the `headers` object are lowercase versions of standard HTTP header names. Using lowercase keys gives you case-insensitive access to the header values.
- Each header (for example, `headers["accept"]` or `headers["host"]`) is an array of key-value pairs. For a given header, the array contains one key-value pair for each value in the generated response. For example, if you want to include 3 values in the host header, the `headers["host"]` array will contain 3 key-value pairs.
- `key` is the case-sensitive name of the header as it appears in an HTTP request; for example, `accept` or `host`.
- `value` is a header value.

For information about restrictions on header usage, see [Headers \(p. 102\)](#).

status

The HTTP status code that you want CloudFront to use for the following:

- Return in the response

- Cache in the CloudFront edge cache, when the response was generated by a function that was triggered by an origin request event
- Log in CloudFront [Access Logs \(p. 315\)](#)

If the status value isn't between 200 and 599, CloudFront returns an error to the viewer.

statusDescription

The description that you want CloudFront to return in the response, to accompany the HTTP status code. You don't need to use standard descriptions, such as `OK` for an HTTP status code of 200.

Errors

The following are possible errors for generated HTTP responses.

Response Contains a Body and Specifies 204 (No Content) for Status

When a function is triggered by a viewer request, CloudFront returns an HTTP 502 status code (Bad Gateway) to the viewer when both of the following are true:

- The value of `status` is 204 (No Content)
- The response includes a value for `body`

This is because Lambda@Edge imposes the optional restriction found in RFC 2616, which states that an HTTP 204 response does not need to contain a message body.

Limits on the Size of the Generated Response

The maximum size of a response that is generated by a Lambda function (including the headers and body) depends on the event that triggered the function:

- **Viewer request events** – 40 KB
- **Origin request events** – 1 MB

If the response is larger than the allowed size, CloudFront returns an HTTP 502 status code (Bad Gateway) to the viewer.

Required Fields

The `status` field is required.

All other fields are optional.

Updating HTTP Responses in an Origin-Response Trigger

When CloudFront receives an HTTP response from the origin server, if there is an origin-response trigger associated with the cache behavior, you can modify the HTTP response to override what was returned from the origin.

Some common scenarios for updating HTTP responses include the following:

- Changing the status to set an HTTP 200 status code and creating static body content to return to the viewer when an origin returns an error status code (4xx or 5xx). For sample code, see [Example: Using an Origin-Response Trigger to Update the Error Status Code to 200-OK \(p. 129\)](#).

- Changing the status to set an HTTP 301 or HTTP 302 status code, to redirect the user to another web site when an origin returns an error status code (4xx or 5xx). For sample code, see [Example: Using an Origin-Response Trigger to Update the Error Status Code to 302-Found \(p. 130\)](#).

When you're working with the HTTP response, note that Lambda@Edge does not expose the HTML body that is returned by the origin server to the origin-response trigger. You can generate a static content body by setting it to the desired value, or remove the body inside the function by setting the value to be empty. If you don't update the body field in your function, the original body returned by the origin server is returned back to viewer.

CloudWatch Metrics and CloudWatch Logs for Lambda Functions

You can use CloudWatch metrics to monitor, in real time, the CloudFront requests that trigger Lambda functions. You can also use CloudWatch Logs to get aggregate data. There's no additional charge for metrics or logs.

Topics

- [CloudWatch Metrics \(p. 117\)](#)
- [CloudWatch Logs \(p. 117\)](#)

CloudWatch Metrics

When you create a trigger for a CloudFront event, Lambda automatically begins to send metrics to CloudWatch. Metrics are available for all Lambda regions. The name of each metric is /aws/lambda/us-east-1.*function-name* where *function-name* is the name that you gave to the function when you created it. CloudWatch sends metrics to the region that's closest to the location where the function is executed.

For more information about CloudWatch metrics, see the [Amazon CloudWatch User Guide](#).

CloudWatch Logs

When you create a trigger, Lambda automatically starts to send data to CloudWatch Logs about the CloudFront requests that trigger Lambda functions. You use CloudWatch Logs tools to access the logs.

Lambda creates CloudWatch Logs log streams in the CloudWatch Logs regions closest to the locations where the function is executed. The format of the name for each log stream is /aws/lambda/us-east-1.*function-name* where *function-name* is the name that you gave to the function when you created it.

For information about the permissions required to send data to CloudWatch Logs, see [Setting IAM Permissions and Roles for Lambda@Edge](#) in the *IAM User Guide*.

For information about adding logging to a Lambda function, see [Logging \(Node.js\)](#) in the *AWS Lambda Developer Guide*.

For information about CloudWatch Logs limits, see [Limits](#) in the *Amazon CloudWatch Logs User Guide*.

Lambda@Edge throttles logs based on the request volume and the size of logs.

Example Functions

Each Lambda@Edge function must contain the `callback` parameter to successfully process a request or return a response. For more information, see [Authoring Functions for Lambda@Edge](#) in the *AWS Lambda Developer Guide*.

Topics

- [Example: A/B Testing \(p. 118\)](#)
- [Example: Overriding a Response Header \(p. 119\)](#)
- [Example: Serving Static Content \(Generated Response\) \(p. 120\)](#)
- [Example: Generating an HTTP Redirect \(Generated Response\) \(p. 121\)](#)
- [Example: Adding a Header Based on a Query String Parameter \(p. 121\)](#)
- [Example: Normalizing Query String Parameters to Improve the Cache Hit Ratio \(p. 122\)](#)
- [Example: Redirecting Unauthenticated Users to a Sign-In Page \(p. 122\)](#)
- [Example: Redirecting Viewer Requests to a Country-Specific URL \(p. 123\)](#)
- [Example: Serving Different Versions of an Object Based on the Device \(p. 124\)](#)
- [Example: Serving Static Website Content as Gzip Compressed Content \(p. 125\)](#)
- [Example: Using an Origin-Request Trigger to Change From a Custom Origin to an Amazon S3 Origin \(p. 126\)](#)
- [Example: Using an Origin-Request Trigger to Change the Amazon S3 Origin Region \(p. 127\)](#)
- [Example: Using an Origin-Request Trigger to Change From an Amazon S3 Origin to a Custom Origin \(p. 127\)](#)
- [Example: Using an Origin-Request Trigger to Gradually Transfer Traffic From One Amazon S3 Bucket to Another \(p. 128\)](#)
- [Example: Using an Origin-Request Trigger to Change the Origin Domain Name Based on the Country Header \(p. 129\)](#)
- [Example: Using an Origin-Response Trigger to Update the Error Status Code to 200-OK \(p. 129\)](#)
- [Example: Using an Origin-Response Trigger to Update the Error Status Code to 302-Found \(p. 130\)](#)

Example: A/B Testing

You can use the following example if you want to test two different versions of your home page, but you don't want to create redirects or change the URL. This example sets cookies when CloudFront receives a request, randomly assigns the user to version A or B, and then returns the corresponding version to the viewer.

```
'use strict';

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const headers = request.headers;

    if (request.uri !== '/experiment-pixel.jpg') {
        // do not process if this is not an A-B test request
        callback(null, request);
        return;
    }

    const cookieExperimentA = 'X-Experiment-Name=A';
    const cookieExperimentB = 'X-Experiment-Name=B';
```

```

const pathExperimentA = '/experiment-group/control-pixel.jpg';
const pathExperimentB = '/experiment-group/treatment-pixel.jpg';

/*
 * Lambda at the Edge headers are array objects.
 *
 * Client may send multiple Cookie headers, i.e.:
 * > GET /viewerRes/test HTTP/1.1
 * > User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1 OpenSSL/1.0.1u
zlib/1.2.3
 * > Cookie: First=1; Second=2
 * > Cookie: ClientCode=abc
 * > Host: example.com
 *
 * You can access the first Cookie header at headers["cookie"][0].value
 * and the second at headers["cookie"][1].value.
 *
 * Header values are not parsed. In the example above,
 * headers["cookie"][0].value is equal to "First=1; Second=2"
 */
let experimentUri;
if (headers.cookie) {
    for (let i = 0; i < headers.cookie.length; i++) {
        if (headers.cookie[i].value.indexOf(cookieExperimentA) >= 0) {
            console.log('Experiment A cookie found');
            experimentUri = pathExperimentA;
            break;
        } else if (headers.cookie[i].value.indexOf(cookieExperimentB) >= 0) {
            console.log('Experiment B cookie found');
            experimentUri = pathExperimentB;
            break;
        }
    }
}

if (!experimentUri) {
    console.log('Experiment cookie has not been found. Throwing dice...');
    if (Math.random() < 0.75) {
        experimentUri = pathExperimentA;
    } else {
        experimentUri = pathExperimentB;
    }
}

request.uri = experimentUri;
console.log(`Request uri set to "${request.uri}"`);
callback(null, request);
},

```

Example: Overriding a Response Header

The following example shows how to change the value of a response header based on the value of another header:

```

'use strict';

exports.handler = (event, context, callback) => {
    const response = event.Records[0].cf.response;
    const headers = response.headers;

    const headerNameSrc = 'X-Amz-Meta-Last-Modified';
    const headerNameDst = 'Last-Modified';

```

```

if (headers[headerNameSrc.toLowerCase()]) {
    headers[headerNameDst.toLowerCase()] = [
        headers[headerNameSrc.toLowerCase()][0],
    ];
    console.log(`Response header "${headerNameDst}" was set to ` +
        `${headers[headerNameDst.toLowerCase()][0].value}`);
}

callback(null, response);
};

```

Example: Serving Static Content (Generated Response)

The following example shows how to use a Lambda function to serve static website content, which reduces the load on the origin server and reduces overall latency.

Note

You can generate HTTP responses only for viewer request and origin request events. For more information, see [Generating HTTP Responses in Request Triggers \(p. 114\)](#).

```

'use strict';

let content = `
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Simple Lambda@Edge Static Content Response</title>
  </head>
  <body>
    <p>Hello from Lambda@Edge!</p>
  </body>
</html>
`;

exports.handler = (event, context, callback) => {
    /*
     * Generate HTTP OK response using 200 status code with HTML body.
     */
    const response = {
        status: '200',
        statusDescription: 'OK',
        headers: [
            'cache-control': [{key: 'Cache-Control', value: 'max-age=100'}],
            'content-type': [{key: 'Content-Type', value: 'text/html'}],
            'content-encoding': [{key: 'Content-Encoding', value: 'UTF-8'}],
        ],
        body: content,
    };
    callback(null, response);
};

```

Example: Generating an HTTP Redirect (Generated Response)

The following example shows how to generate an HTTP redirect.

Note

You can generate HTTP responses only for viewer request and origin request events. For more information, see [Generating HTTP Responses in Request Triggers \(p. 114\)](#).

```
'use strict';

exports.handler = (event, context, callback) => {
    /*
     * Generate HTTP redirect response with 302 status code and Location header.
     */
    const response = {
        status: '302',
        statusDescription: 'Found',
        headers: [
            location: [
                {
                    key: 'Location',
                    value: 'http://docs.aws.amazon.com/lambda/latest/dg/lambda-edge.html',
                },
            ],
        ],
    };
    callback(null, response);
};
```

Example: Adding a Header Based on a Query String Parameter

The following example shows how to get the key-value pair of a query string parameter and add a header based on those values.

```
'use strict';

const querystring = require('querystring');
exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    /* When a request contains a query string key-value pair but the origin server
     * expects the value in a header, you can use this Lambda function to
     * convert the key-value pair to a header. Here's what the function does:
     * 1. Parses the query string and gets the key-value pair.
     * 2. Adds a header to the request using the key-value pair that the function got in
     * step 1.
     */

    /* Parse request querystring to get javascript object */
    const params = querystring.parse(request.querystring);

    /* Move auth param from querystring to headers */
    const headerName = 'Auth-Header';
    request.headers[headerName.toLowerCase()] = [{ key: headerName, value: params.auth }];
    delete params.auth;

    /* Update request querystring */
    request.querystring = querystring.stringify(params);

    callback(null, request);
};
```

```
}
```

```
;
```

Example: Normalizing Query String Parameters to Improve the Cache Hit Ratio

The following example shows how to improve your cache hit ratio by making the following changes to query strings before CloudFront forwards requests to your origin:

- Alphabetize key-value pairs by the name of the parameter
- Change the case of key-value pairs to lowercase

For more information, see [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#).

```
'use strict';

const querystring = require('querystring');

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    /* When you configure a distribution to forward query strings to the origin and
     * to cache based on a whitelist of query string parameters, we recommend
     * the following to improve the cache-hit ratio:
     * - Always list parameters in the same order.
     * - Use the same case for parameter names and values.
     *
     * This function normalizes query strings so that parameter names and values
     * are lowercase and parameter names are in alphabetical order.
     *
     * For more information, see:
     * http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/
QueryStringParameters.html
    */

    console.log('Query String: ', request.querystring);

    /* Parse request query string to get javascript object */
    const params = querystring.parse(request.querystring.toLowerCase());
    const sortedParams = {};

    /* Sort param keys */
    Object.keys(params).sort().forEach(key => {
        sortedParams[key] = params[key];
    });

    /* Update request querystring with normalized */
    request.querystring = querystring.stringify(sortedParams);

    callback(null, request);
};
```

Example: Redirecting Unauthenticated Users to a Sign-In Page

The following example shows how to redirect users to a sign-in page if they haven't entered their credentials.

```
'use strict';

function parseCookies(headers) {
    const parsedCookie = {};
    if (headers.cookie) {
        headers.cookie[0].value.split(';').forEach((cookie) => {
            if (cookie) {
                const parts = cookie.split('=');
                parsedCookie[parts[0].trim()] = parts[1].trim();
            }
        });
    }
    return parsedCookie;
}

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const headers = request.headers;

    /* Check for session-id in request cookie in viewer-request event,
     * if session-id is absent, redirect the user to sign in page with original
     * request sent as redirect_url in query params.
     */

    /* Check for session-id in cookie, if present then proceed with request */
    const parsedCookies = parseCookies(headers);
    if (parsedCookies && parsedCookies['session-id']) {
        callback(null, request);
    }

    /* URI encode the original request to be sent as redirect_url in query params */
    const encodedRedirectUrl = encodeURIComponent(`https://${headers.host[0].value}${request.uri}?${request.QueryString}`);
    const response = {
        status: '302',
        statusDescription: 'Found',
        headers: {
            location: [
                {
                    key: 'Location',
                    value: `http://www.example.com/signin?redirect_url=${encodedRedirectUrl}`,
                },
            ],
        },
    };
    callback(null, response);
};
```

Example: Redirecting Viewer Requests to a Country-Specific URL

The following example shows how to generate an HTTP redirect response with a country-specific URL and return the response to the viewer. This is useful when you want to provide country-specific responses. For example:

- If you have country-specific subdomains, such as us.example.com and tw.example.com, you can generate a redirect response when a viewer requests example.com.
- If you're streaming video but you don't have rights to stream the content in a specific country, you can redirect users in that country to a page that explains why they can't view the video.

Note the following:

- You must configure your distribution to cache based on the `CloudFront-Viewer-Country` header. For more information, see [Cache Based on Selected Request Headers \(p. 73\)](#).
- CloudFront adds the `CloudFront-Viewer-Country` header after the viewer request event. To use this example, you must create a trigger for the origin request event.

```
'use strict';

/* This is an origin request function */
exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const headers = request.headers;

    /*
     * Based on the value of the CloudFront-Viewer-Country header, generate an
     * HTTP status code 302 (Redirect) response, and return a country-specific
     * URL in the Location header.
     * NOTE: 1. You must configure your distribution to cache based on the
     *        CloudFront-Viewer-Country header. For more information, see
     *        http://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
     *        2. CloudFront adds the CloudFront-Viewer-Country header after the viewer
     *           request event. To use this example, you must create a trigger for the
     *           origin request event.
    */

    let url = 'https://example.com/';
    if (headers['cloudfront-viewer-country']) {
        const countryCode = headers['cloudfront-viewer-country'][0].value;
        if (countryCode === 'TW') {
            url = 'https://tw.example.com/';
        } else if (countryCode === 'US') {
            url = 'https://us.example.com/';
        }
    }

    const response = {
        status: '302',
        statusDescription: 'Found',
        headers: [
            {
                location: [
                    {
                        key: 'Location',
                        value: url,
                    },
                ],
            },
        ],
    };
    callback(null, response);
};
```

Example: Serving Different Versions of an Object Based on the Device

The following example shows how to serve different versions of an object based on the type of device that the user is using, for example, a mobile device or a tablet. Note the following:

- You must configure your distribution to cache based on the `CloudFront-Is-**-Viewer` headers. For more information, see [Cache Based on Selected Request Headers \(p. 73\)](#).
- CloudFront adds the `CloudFront-Is-**-Viewer` headers after the viewer request event. To use this example, you must create a trigger for the origin request event.

```
'use strict';

/* This is an origin request function */
exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const headers = request.headers;

    /*
     * Serve different versions of an object based on the device type.
     * NOTE: 1. You must configure your distribution to cache based on the
     *       CloudFront-Is-*-Viewer headers. For more information, see
     *       the following documentation:
     *       http://docs.aws.amazon.com/console/cloudfront/cache-on-selected-headers
     *       http://docs.aws.amazon.com/console/cloudfront/cache-on-device-type
     * 2. CloudFront adds the CloudFront-Is-*-Viewer headers after the viewer
     *       request event. To use this example, you must create a trigger for the
     *       origin request event.
    */

    const desktopPath = '/desktop';
    const mobilePath = '/mobile';
    const tabletPath = '/tablet';
    const smarttvPath = '/smarttv';

    if (headers['cloudfront-is-desktop-viewer']
        && headers['cloudfront-is-desktop-viewer'][0].value === 'true') {
        request.uri = desktopPath + request.uri;
    } else if (headers['cloudfront-is-mobile-viewer']
        && headers['cloudfront-is-mobile-viewer'][0].value === 'true') {
        request.uri = mobilePath + request.uri;
    } else if (headers['cloudfront-is-tablet-viewer']
        && headers['cloudfront-is-tablet-viewer'][0].value === 'true') {
        request.uri = tabletPath + request.uri;
    } else if (headers['cloudfront-is-smarttv-viewer']
        && headers['cloudfront-is-smarttv-viewer'][0].value === 'true') {
        request.uri = smarttvPath + request.uri;
    }
    console.log(`Request uri set to "${request.uri}"`);

    callback(null, request);
};

};
```

Example: Serving Static Website Content as Gzip Compressed Content

This function demonstrates how to use a Lambda function to serve static website content as gzip compressed content, which reduces the load on the origin server and reduces overall latency.

You can generate HTTP responses only for viewer request and origin request events. For more information, see [Generating HTTP Responses in Request Triggers \(p. 114\)](#)

```
'use strict';

const zlib = require('zlib');

let content = `
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
        <title>Simple Lambda@Edge Static Content Response</title>
    </head>
```

```
<body>
    <p>Hello from Lambda@Edge!</p>
</body>
</html>
`;

exports.handler = (event, context, callback) => {

    /*
     * Generate HTTP OK response using 200 status code with a gzip compressed content HTML
     * body.
     */

    const buffer = zlib.gzipSync(content);
    const base64EncodedBody = buffer.toString('base64');

    var response = {
        headers: [
            'content-type': [{key:'Content-Type', value: 'text/html; charset=utf-8'}],
            'content-encoding' : [{key:'Content-Encoding', value: 'gzip'}]
        ],
        body: base64EncodedBody,
        bodyEncoding: 'base64',
        status: '200',
        statusDescription: "OK"
    }

    callback(null, response);
};
```

Example: Using an Origin-Request Trigger to Change From a Custom Origin to an Amazon S3 Origin

This function demonstrates how an origin-request trigger can be used to change from a custom origin to an Amazon S3 origin from which the content is fetched, based on request properties.

```
'use strict';

const querystring = require('querystring');

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    /**
     * Reads query string to check if S3 origin should be used, and
     * if true, sets S3 origin properties.
     */

    const params = querystring.parse(request.querystring);

    if (params['useS3Origin']) {
        if (params['useS3Origin'] === 'true') {
            const s3DomainName = 'my-bucket.s3.amazonaws.com';

            /* Set S3 origin fields */
            request.origin = {
                s3: {
                    domainName: s3DomainName,
                    region: '',
                    authMethod: 'none',
                    path: '',
                    customHeaders: {}
                }
            };
        }
    }
};

module.exports = exports.handler;
```

```
        }
    );
    request.headers['host'] = [{ key: 'host', value: s3DomainName}];
}
}

callback(null, request);
};
```

Example: Using an Origin-Request Trigger to Change the Amazon S3 Origin Region

This function demonstrates how an origin-request trigger can be used to change the Amazon S3 origin from which the content is fetched, based on request properties.

In this example, we check to see if a region is provided as a parameter in the query string, and if present, we update the S3 bucket domain name to a bucket in that region. This can be useful in several ways:

- It reduces latencies when the region specified is nearer to the viewer's country.
- It provides data sovereignty by making sure that data is served from an origin that's in the same country that the request came from.

```
'use strict';

const querystring = require('querystring');

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const params = querystring.parse(request.querystring);

    if (params.region) {
        /**
         * If you've set up OAI, the bucket policy in the destination bucket
         * should allow the OAI GetObject operation, as configured by default
         * for an S3 origin with OAI. Another requirement with OAI is to provide
         * the region so it can be used for the SIGV4 signature. Otherwise, the
         * region is not required.
        */
        request.origin.s3.region = params.region;
        const domainName = `my-bucket-in-${params.region}.s3.amazonaws.com`;
        request.origin.s3.domainName = domainName;
        request.headers['host'] = [{ key: 'host', value: domainName }];
    }

    callback(null, request);
};
```

Example: Using an Origin-Request Trigger to Change From an Amazon S3 Origin to a Custom Origin

This function demonstrates how an origin-request trigger can be used to change the custom origin from which the content is fetched, based on request properties.

```
'use strict';

const querystring = require('querystring');
```

```

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    /**
     * Reads query string to check if custom origin should be used, and
     * if true, sets custom origin properties.
     */

    const params = querystring.parse(request.querystring);

    if (params['useCustomOrigin']) {
        if (params['useCustomOrigin'] === 'true') {

            /* Set custom origin fields*/
            request.origin = {
                custom: {
                    domainName: 'www.example.com',
                    port: 443,
                    protocol: 'https',
                    path: '',
                    sslProtocols: ['TLSv1', 'TLSv1.1'],
                    readTimeout: 5,
                    keepaliveTimeout: 5,
                    customHeaders: {}
                }
            };
            request.headers['host'] = [{ key: 'host', value: 'www.example.com'}];
        }
    }
    callback(null, request);
};

```

Example: Using an Origin-Request Trigger to Gradually Transfer Traffic From One Amazon S3 Bucket to Another

This function demonstrates how you can gradually transfer traffic from one Amazon S3 bucket to another, in a controlled way.

```

'use strict';

function getRandomInt(min, max) {
    /* Random number is inclusive of min and max*/
    return Math.floor(Math.random() * (max - min + 1)) + min;
}

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;
    const BLUE_TRAFFIC_PERCENTAGE = 80;

    /**
     * This Lambda function demonstrates how to gradually transfer traffic from
     * one S3 bucket to another in a controlled way.
     * We define a variable BLUE_TRAFFIC_PERCENTAGE which can take values from
     * 1 to 100. If the generated randomNumber less than or equal to
     * BLUE_TRAFFIC_PERCENTAGE, traffic
     * is re-directed to blue-bucket. If not, the default bucket that we've configured
     * is used.
     */
}

```

```
const randomNumber = getRandomInt(1, 100);

if (randomNumber <= BLUE_TRAFFIC_PERCENTAGE) {
    const domainName = 'blue-bucket.s3.amazonaws.com';
    request.origin.s3.domainName = domainName;
    request.headers['host'] = [{ key: 'host', value: domainName }];
}
callback(null, request);
};
```

Example: Using an Origin-Request Trigger to Change the Origin Domain Name Based on the Country Header

This function demonstrates how you can change the origin domain name based on the CloudFront-Viewer-Country header, so content is served from an origin closer to the viewer's country.

Based on the value of the CloudFront-Viewer-Country header, you can change the origin domain name so content is served from an origin nearer to viewer's country. This can have advantages such as the following:

- It reduces latencies when the region specified is nearer to the viewer's country.
- It provides data sovereignty by making sure that data is served from an origin that's in the same country that the request came from.

Note that to enable this functionality, you must configure your distribution to cache based on the CloudFront-Viewer-Country header. For more information, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

```
'use strict';

exports.handler = (event, context, callback) => {
    const request = event.Records[0].cf.request;

    if (request.headers['cloudfront-viewer-country']) {
        const countryCode = request.headers['cloudfront-viewer-country'][0].value;
        if (countryCode === 'UK' || countryCode === 'DE' || countryCode === 'IE') {
            const domainName = 'eu.example.com';
            request.origin.custom.domainName = domainName;
            request.headers['host'] = [{key: 'host', value: domainName}];
        }
    }

    callback(null, request);
};
```

Example: Using an Origin-Response Trigger to Update the Error Status Code to 200-OK

This function demonstrates how you can update the response status to 200 and generate static body content to return to the viewer in the following scenario:

- The function is triggered in an origin response
- The response status from the origin server is an error status code (4xx or 5xx)

```
'use strict';

exports.handler = (event, context, callback) => {
    const response = event.Records[0].cf.response;

    /**
     * This function updates the response status to 200 and generates static
     * body content to return to the viewer in the following scenario:
     * 1. The function is triggered in an origin response
     * 2. The response status from the origin server is an error status code (4xx or 5xx)
     */

    if (response.status >= 400 && response.status <= 599) {
        response.status = 200;
        response.statusDescription = 'OK';
        response.body = 'Body generation example';
    }

    callback(null, response);
};
```

Example: Using an Origin-Response Trigger to Update the Error Status Code to 302-Found

This function demonstrates how you can update the HTTP status code to 302 to redirect to another path (cache behavior) that has a different origin configured. Note the following:

- The function is triggered in an origin response
- The response status from the origin server is an error status code (4xx or 5xx)

```
'use strict';

exports.handler = (event, context, callback) => {
    const response = event.Records[0].cf.response;
    const request = event.Records[0].cf.request;

    /**
     * This function updates the HTTP status code in the response to 302, to redirect to
     * another
     * path (cache behavior) that has a different origin configured. Note the following:
     * 1. The function is triggered in an origin response
     * 2. The response status from the origin server is an error status code (4xx or 5xx)
     */

    if (response.status >= 400 && response.status <= 599) {
        const redirect_path = `/plan-b/path?${request.querystring}`;

        response.status = 302;
        response.statusDescription = 'Found';

        /* Drop the body, as it is not required for redirects */
        response.body = '';
        response.headers['location'] = [{ key: 'Location', value: redirect_path }];
    }

    callback(null, response);
};
```

Working with RTMP Distributions

Topics

- [How RTMP Distributions Work \(p. 131\)](#)
- [Task List for Streaming Media Files Using RTMP \(p. 132\)](#)
- [Creating an RTMP Distribution Using the CloudFront Console \(p. 133\)](#)
- [Values that You Specify When You Create or Update an RTMP Distribution \(p. 134\)](#)
- [Values that CloudFront Displays in the Console When You Create or Update an RTMP Distribution \(p. 138\)](#)
- [Configuring the Media Player \(p. 139\)](#)
- [Using an Amazon S3 Bucket as the Origin for an RTMP Distribution \(p. 140\)](#)
- [Creating Multiple RTMP Distributions for an Origin Server \(p. 140\)](#)
- [Restricting Access Using Crossdomain.xml \(p. 141\)](#)
- [Error Codes for RTMP Distributions \(p. 141\)](#)
- [Troubleshooting RTMP Distributions \(p. 141\)](#)

This section describes how you configure and manage RTMP distributions. For information about how to create an RTMP distribution, see [Task List for Streaming Media Files Using RTMP \(p. 132\)](#).

How RTMP Distributions Work

To stream media files using CloudFront, you provide two types of files to your end users:

- Your media files
- A media player, for example, JW Player, Flowplayer, or Adobe Flash

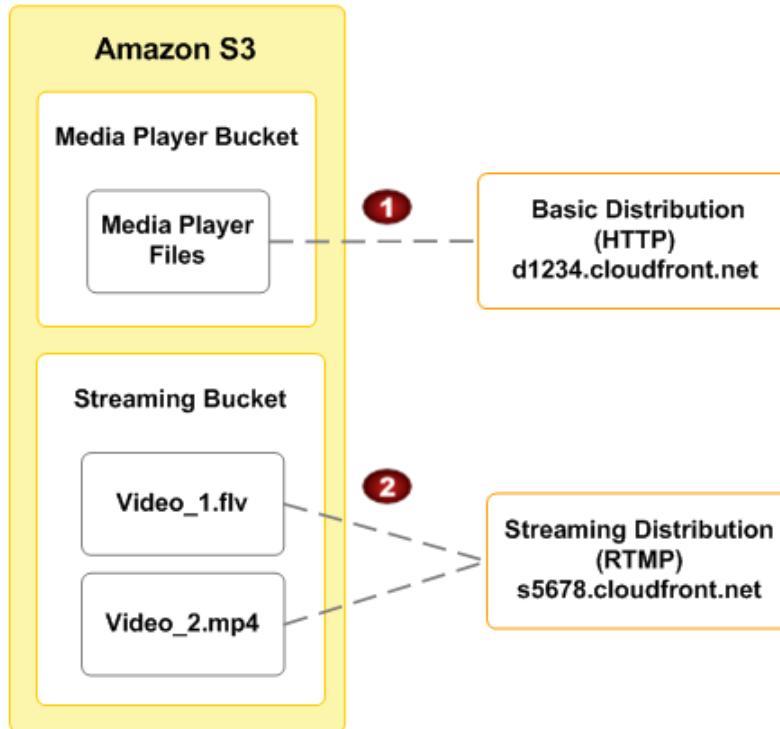
End users view your media files using the media player that you provide for them; they do not use the media player (if any) that is already installed on their computer or other device.

When an end user streams your media file, the media player begins to play the content of the file while the file is still being downloaded from CloudFront. The media file is not stored locally on the end user's system.

To use CloudFront to serve both the media player and the media files, you need two types of distributions: a web distribution for the media player, and an RTMP distribution for the media files. Web distributions serve files over HTTP, while RTMP distributions stream media files over RTMP (or a variant of RTMP).

The following example assumes that your media files and your media player are stored in different buckets in Amazon S3, but that isn't required—you can store media files and your media player in the same Amazon S3 bucket. You can also make the media player available to end users in other ways, for example, using CloudFront and a custom origin. However, the media files must use an Amazon S3 bucket as the origin.

In the following diagram, your site serves a cached copy of the media player to each end user through the `d1234.cloudfront.net` domain. The media player then accesses cached copies of your media files through the `s5678.cloudfront.net` domain.



1. Your media player bucket holds the media player and is the origin server for a regular HTTP distribution. In this example, the domain name for the distribution is `d1234.cloudfront.net`. (The `d` in `d1234.cloudfront.net` indicates that this is a web distribution.)
2. Your streaming media bucket holds your media files and is the origin server for an RTMP distribution. In this example, the domain name for the distribution is `s5678.cloudfront.net`. (The `s` in `s5678.cloudfront.net` indicates that this is an RTMP distribution.)

When you configure CloudFront to distribute media files, CloudFront uses Adobe Flash Media Server as the streaming server and streams your media files using Adobe's Real-Time Messaging Protocol (RTMP). CloudFront accepts RTMP requests over port 1935 and port 80.

CloudFront supports the following variants of the RTMP protocol:

- **RTMP** – Adobe's Real-Time Message Protocol
- **RTMPT** – Adobe streaming tunneled over HTTP
- **RTMPE** – Adobe encrypted
- **RTMPTE** – Adobe encrypted tunneled over HTTP

Task List for Streaming Media Files Using RTMP

This section summarizes the general process for configuring on-demand streaming using the Adobe RTMP protocol for any media player. If you're using JW Player for your media player, see [On-Demand Video Streaming Using CloudFront and JW Player \(p. 354\)](#).

The following task list summarizes the process for creating a web distribution.

To Create an RTMP Distribution

1. Create an Amazon S3 bucket for your media files. If you are using a different Amazon S3 bucket for your media player, create an Amazon S3 bucket for the media player files, too.
The names of your buckets must be all lowercase and cannot contain spaces.
2. Choose and configure a media player to play your media files. For more information, refer to the documentation for the media player.
3. Upload the files for your media player to the origin from which you want CloudFront to get the files. If you are using an Amazon S3 bucket as the origin for the media player, make the files (not the bucket) publicly readable.
4. Create a web distribution for your media player. (You can also use an existing distribution.) For more information, see [Task List for Creating a Web Distribution \(p. 59\)](#).
5. Upload your media files to the Amazon S3 bucket that you created for the media files, and make the content (not the bucket) publicly readable.

Important

Media files in a Flash Video container must include the .flv filename extension, or the media will not stream.

You can put media player files and media files in the same bucket.

6. Create an RTMP distribution for your media files:
 - For more information about creating a web distribution using the CloudFront console, see [Creating an RTMP Distribution Using the CloudFront Console \(p. 133\)](#).
 - For information about creating a web distribution using the CloudFront API, go to [POST Streaming Distribution](#) in the [Amazon CloudFront API Reference](#).
7. Configure your media player. For more information, see [Configuring the Media Player \(p. 139\)](#).

If you have trouble getting your content to play, see [Troubleshooting RTMP Distributions \(p. 141\)](#).

Creating an RTMP Distribution Using the CloudFront Console

The following procedure explains how to create an RTMP distribution using the CloudFront console. If you want to create an RTMP distribution using the CloudFront API, go to [POST Streaming Distribution](#) in the [Amazon CloudFront API Reference](#).

For the current limit on the number of RTMP distributions that you can create for each AWS account, see [Amazon CloudFront Limits](#) in the [Amazon Web Services General Reference](#). To request a higher limit, go to <https://console.aws.amazon.com/support/home#/case/create?issueType=service-limit-increase&limitType=service-code-cloudfront-distributions>.

To create an RTMP distribution using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click **Create Distribution**.
3. On the first page of the **Create Distribution Wizard**, in the **RTMP** section, choose **Get Started**.
4. Specify settings for the distribution. For more information, see [Values that You Specify When You Create or Update an RTMP Distribution \(p. 134\)](#).
5. Click **Create Distribution**.

6. After CloudFront creates your distribution, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. If you chose to enable the distribution, it will then be ready to process requests. This should take less than 15 minutes.

The domain name that CloudFront assigns to your distribution appears in the list of distributions. The domain name also appears on the **General** tab for a selected distribution.

Values that You Specify When You Create or Update an RTMP Distribution

To stream media files using CloudFront, you create an RTMP distribution and specify the following values.

Topics

- [Origin Domain Name \(Amazon S3 Bucket\) \(p. 134\)](#)
- [Restrict Bucket Access \(Amazon S3 Only\) \(p. 135\)](#)
- [Origin Access Identity \(Amazon S3 Only\) \(p. 135\)](#)
- [Comment for New Identity\(Amazon S3 Only\) \(p. 135\)](#)
- [Your Identities \(Amazon S3 Only\) \(p. 135\)](#)
- [Grant Read Permissions on Bucket \(Amazon S3 Only\) \(p. 135\)](#)
- [Price Class \(p. 136\)](#)
- [Alternate Domain Names \(CNAMEs\) \(p. 136\)](#)
- [Logging \(p. 136\)](#)
- [Bucket for Logs \(p. 136\)](#)
- [Log Prefix \(p. 136\)](#)
- [Comment \(p. 137\)](#)
- [Distribution State \(p. 137\)](#)
- [Restrict Viewer Access \(Use Signed URLs\) \(p. 137\)](#)
- [Trusted Signers \(p. 137\)](#)
- [AWS Account Numbers \(p. 138\)](#)

Origin Domain Name (Amazon S3 Bucket)

The DNS domain name of the Amazon S3 bucket from which you want CloudFront to get objects for this origin, for example, `myawsbucket.s3.amazonaws.com`. In the CloudFront console, click in the **Origin Domain Name** field, and a list enumerates the Amazon S3 buckets that are associated with the current AWS account. To use a bucket from a different AWS account, type the domain name of the bucket in the following format:

`bucket-name.s3.amazonaws.com`

If your bucket is in the US Standard region and you want Amazon S3 to route requests to a facility in Northern Virginia, use the following format:

`bucket-name.s3-external-1.amazonaws.com`

If you configured Amazon S3 Transfer Acceleration for your bucket, do not specify the `s3-accelerate` endpoint for **Origin Domain Name**.

The files must be publicly readable unless you secure your content in Amazon S3 by using a CloudFront origin access identity. For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

Important

The bucket name must conform to DNS naming requirements. For more information, go to [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

When you change the bucket from which CloudFront gets objects for the current origin, CloudFront immediately begins replicating the change to CloudFront edge locations. Until the distribution configuration is updated in a given edge location, CloudFront will continue to forward requests to the previous Amazon S3 bucket. As soon as the distribution configuration is updated in that edge location, CloudFront begins to forward requests to the new Amazon S3 bucket.

Changing the bucket does not require CloudFront to repopulate edge caches with objects from the new origin. As long as the viewer requests in your application have not changed, CloudFront will continue to serve objects that are already in an edge cache until the TTL on each object expires or until seldom-requested objects are evicted.

For more information, see [Using an Amazon S3 Bucket as the Origin for an RTMP Distribution \(p. 140\)](#).

Restrict Bucket Access (Amazon S3 Only)

Click **Yes** if you want to require end users to access objects in an Amazon S3 bucket by using only CloudFront URLs, not by using Amazon S3 URLs. Then specify the applicable values.

Click **No** if you want end users to be able to access objects using either CloudFront URLs or Amazon S3 URLs.

For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

Origin Access Identity (Amazon S3 Only)

If you chose **Yes** for **Restrict Bucket Access**, choose whether to create a new origin access identity or use an existing one that is associated with your AWS account. If you already have an origin access identity, we recommend that you reuse it to simplify maintenance. For more information about origin access identities, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

Comment for New Identity(Amazon S3 Only)

If you chose **Create a New Identity for Origin Access Identity**, enter a comment that identifies the new origin access identity. CloudFront will create the origin access identity when you create this distribution.

Your Identities (Amazon S3 Only)

If you chose **Use an Existing Identity for Origin Access Identity**, choose the origin access identity that you want to use. You cannot use an origin access identity that is associated with another AWS account.

Grant Read Permissions on Bucket (Amazon S3 Only)

If you want CloudFront to automatically grant the origin access identity the permission to read objects in your Amazon S3 bucket, click **Yes, Update Bucket Policy**.

Important

If you click **Yes, Update Bucket Policy**, CloudFront updates the bucket policy to grant the specified origin access identity the permission to read objects in your bucket. However,

CloudFront does not remove existing permissions in the bucket policy or permissions on individual objects. If users currently have permission to access the objects in your bucket using Amazon S3 URLs, they will still have that permission after CloudFront updates your bucket policy. To view or change the existing bucket policy and the existing permissions on the objects in your bucket, use a method provided by Amazon S3. For more information, see [Granting the Origin Access Identity Permission to Read Objects in Your Amazon S3 Bucket \(p. 213\)](#).

If you want to update permissions manually, for example, if you want to update ACLs on your objects instead of updating bucket permissions, click **No, I will Update Permissions**.

Price Class

The price class that corresponds with the maximum price that you want to pay for CloudFront service. By default, CloudFront serves your objects from edge locations in all CloudFront regions.

For more information about price classes and about how your choice of price class affects CloudFront performance for your distribution, see [Choosing the Price Class for a CloudFront Distribution \(p. 55\)](#). For information about CloudFront pricing, including how price classes map to CloudFront regions, go to [Amazon CloudFront Pricing](#).

Alternate Domain Names (CNAMEs)

Optional. You can associate one or more CNAME aliases with a distribution so that you can use your domain name (for example, example.com) in the URLs for your objects instead of using the domain name that CloudFront assigned when you created your distribution. For more information, see [Using Alternate Domain Names \(CNAMEs\) \(p. 50\)](#).

Logging

Whether you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket. You can enable or disable logging at any time. There is no extra charge if you enable logging, but you accrue the usual Amazon S3 charges for storing and accessing the files in an Amazon S3 bucket. You can delete the logs at any time. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Bucket for Logs

If you chose **On** for **Logging**, the Amazon S3 bucket that you want CloudFront to store access logs in, for example, `myawslogbucket.s3.amazonaws.com`. If you enable logging, CloudFront records information about each end-user request for an object and stores the files in the specified Amazon S3 bucket. You can enable or disable logging at any time. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Note

You must have the permissions required to get and update Amazon S3 bucket ACLs, and the S3 ACL for the bucket must grant you `FULL_CONTROL`. This allows CloudFront to give the `awsdatafeeds` account permission to save log files in the bucket. For more information, see [Permissions Required to Configure Logging and to Access Your Log Files \(p. 316\)](#).

Log Prefix

Optional. If you chose **On** for **Logging**, specify the string, if any, that you want CloudFront to prefix to the access log filenames for this distribution, for example, `exampleprefix/`. The trailing slash (/) is optional but recommended to simplify browsing your log files. For more information about CloudFront access logs, see [Access Logs \(p. 315\)](#).

Comment

Optional. When you create a distribution, you can include a comment of up to 128 characters. You can update the comment at any time.

Distribution State

When you create a distribution, you must specify whether you want the distribution to be enabled or disabled after it's created:

- *Enabled* means that as soon as the distribution is fully deployed you can deploy links that use the distribution's domain name and end users can retrieve content. Whenever a distribution is enabled, CloudFront accepts and processes any end-user requests for content that use the domain name associated with that distribution.

When you create, modify, or delete a CloudFront distribution, it takes time for your changes to propagate to the CloudFront database. An immediate request for information about a distribution might not show the change. Propagation usually completes within minutes, but a high system load or network partition might increase this time.

- *Disabled* means that even though the distribution might be deployed and ready to use, end users can't use it. When a distribution is disabled, CloudFront doesn't accept any end-user requests that use the domain name associated with that distribution. Until you switch the distribution from disabled to enabled (by updating the distribution's configuration), no one can use it.

You can toggle a distribution between disabled and enabled as often as you want. For information about updating a distribution's configuration, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#).

Restrict Viewer Access (Use Signed URLs)

If you want requests for objects served by this distribution to use public URLs, click **No**. If you want requests to use signed URLs, click **Yes**. Then specify the AWS accounts that you want to use to create signed URLs; these accounts are known as trusted signers.

For more information about trusted signers, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#).

Trusted Signers

Choose which AWS accounts you want to use as trusted signers for this distribution:

- **Self:** Use the account with which you're currently signed into the AWS Management Console as a trusted signer. If you're currently signed in as an IAM user, the associated AWS account is added as a trusted signer.
- **Specify Accounts:** Enter account numbers for trusted signers in the **AWS Account Numbers** field.

To create signed URLs, an AWS account must have at least one active CloudFront key pair.

Important

If you're updating a distribution that you're already using to distribute content, add trusted signers only when you're ready to start generating signed URLs for your objects. After you add trusted signers to a distribution, users must use signed URLs to access the objects served by this distribution.

AWS Account Numbers

If you want to create signed URLs using AWS accounts in addition to or instead of the current account, enter one AWS account number per line in this field. Note the following:

- The accounts that you specify must have at least one active CloudFront key pair. For more information, see [Creating CloudFront Key Pairs for Your Trusted Signers \(p. 217\)](#).
- You can't create CloudFront key pairs for IAM users, so you can't use IAM users as trusted signers.
- For information about how to get the AWS account number for an account, see [How Do I Get Security Credentials?](#) in the *Amazon Web Services General Reference*.
- If you enter the account number for the current account, CloudFront automatically checks the **Self** checkbox and removes the account number from the **AWS Account Numbers** list.

Values that CloudFront Displays in the Console When You Create or Update an RTMP Distribution

When you create a new RTMP distribution or update an existing distribution, CloudFront displays the following information in the CloudFront console.

Note

Active trusted signers, the AWS accounts that have an active CloudFront key pair and can be used to create valid signed URLs, are currently not visible in the CloudFront console.

Distribution ID

When you perform an action on a distribution using the CloudFront API, you use the distribution ID to specify which distribution you want to perform the action on, for example, EDFDVBD6EXAMPLE. You can't change the distribution ID.

Status

The possible status values for a distribution are listed in the following table.

| Value | Description |
|------------|--|
| InProgress | The distribution is still being created or updated. |
| Deployed | The distribution has been created or updated and the changes have been fully propagated through the CloudFront system. |

In addition to ensuring that the status for a distribution is **Deployed**, you must enable the distribution before end users can use CloudFront to access your content. For more information, see [Distribution State \(p. 137\)](#).

Last Modified

The date and time that the distribution was last modified, using ISO 8601 format, for example, 2012-05-19T19:37:58Z. For more information, go to <http://www.w3.org/TR/NOTE-datetime>.

Domain Name

You use the distribution's domain name in the links to your objects, unless you're using alternate domain names (CNAMEs). For example, if your distribution's domain name is `d111111abcdef8.cloudfront.net`, the link to the example `/images/image.jpg` file would be `http://d111111abcdef8.cloudfront.net/images/image.jpg`. You can't change the CloudFront domain name for your distribution. For more information about CloudFront URLs for links to your objects, see [Format of URLs for Objects \(p. 143\)](#).

If you specified one or more alternate domain names (CNAMEs), you can use your own domain names for links to your objects instead of using the CloudFront domain name. For more information about CNAMEs, see [Alternate Domain Names \(CNAMEs\) \(p. 78\)](#).

Note

CloudFront domain names are unique. Your distribution's domain name was never used for a previous distribution and will never be reused for another distribution in the future.

Configuring the Media Player

To play a media file, you must configure the media player with the correct path to the file. How you configure the media depends on which media player you're using and how you're using it.

When you configure the media player, the path you specify to the media file must contain the characters `cfx/st` immediately after the domain name, for example:

```
rtmp://s5c39gqb8ow64r.cloudfront.net/cfx/st/mediafile.flv.
```

Note

CloudFront follows Adobe's FMS naming requirements. Different players have their own rules about how to specify streams. The example above is for JW Player. Check your player's documentation. For example, Adobe's Flash Media Server does not allow the `.flv` extension to be present on the play path. Many players remove the `.flv` extension for you.

Your media player might ask for the path separate from the file name. For example, with the JW Player wizard, you specify a `streamer` and `file` variable:

- **streamer** – `rtmp://s5c39gqb8ow64r.cloudfront.net/cfx/st` (with no trailing slash)
- **file** – `mediafile.flv`

If you've stored the media files in a directory in your bucket (for example, `videos/mediafile.flv`), then the variables for JW Player would be:

- **streamer** – `rtmp://s5c39gqb8ow64r.cloudfront.net/cfx/st` (with no trailing slash)
- **file** – `videos/mediafile.flv`

To use the JW Player wizard, go to the [Setup Wizard](#) page on the JW Player website.

MPEG Files

To serve MP3 audio files or H.264/MPEG-4 video files, you might need to prefix the file name with `mp3:` or `mp4:`. Some media players can be configured to add the prefix automatically. The media player might also require you to specify the file name without the file extension (for example, `magicvideo` instead of `magicvideo.mp4`).

Using an Amazon S3 Bucket as the Origin for an RTMP Distribution

When you create a distribution, you specify where CloudFront gets the files that it distributes to edge locations. For an RTMP distribution, you must use an Amazon S3 bucket; custom origins are not supported. To get your objects into your bucket, you can use any method supported by Amazon S3, for example, the Amazon S3 API or a third-party tool. You can create a hierarchy in your bucket just as you would with any other Amazon S3 bucket. You incur regular Amazon S3 charges for storing the objects in the bucket. For more information about the charges to use CloudFront, see [CloudFront Reports \(p. 15\)](#).

Using an existing Amazon S3 bucket as your CloudFront origin server doesn't change the bucket in any way; you can still use it as you normally would to store and access Amazon S3 objects (at the normal Amazon S3 prices).

You can use the same Amazon S3 bucket for both RTMP and web distributions.

Note

After you create an RTMP distribution, you can't change its origin server. If you need to change the Amazon S3 bucket for an RTMP distribution, you must create a new distribution that uses the new bucket and update either your links or your DNS records to use the domain name for the new distribution. You can then delete the original distribution. For more information, see [Deleting a Distribution \(p. 50\)](#).

When you specify the name of the Amazon S3 bucket that you want CloudFront to get objects from, you generally use the following format:

`bucket-name.s3.amazonaws.com`

If your bucket is in the US Standard region and you want Amazon S3 to route requests to a facility in Northern Virginia, use the following format:

`bucket-name.s3-external-1.amazonaws.com`

Do not specify the name of the bucket using the following values:

- The Amazon S3 path style, `s3.amazonaws.com/bucket-name`
- The Amazon S3 CNAME, if any

Important

For your bucket to work with CloudFront, the name must conform to DNS naming requirements. For more information, go to [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

Creating Multiple RTMP Distributions for an Origin Server

You typically create one RTMP distribution per Amazon S3 bucket, but you can choose to create multiple RTMP distributions for the same bucket. For example, if you had two distributions for an Amazon S3 bucket, you could reference a single media file using either distribution. In this case, if you had a media file called `media.flv` in your origin server, CloudFront would work with each distribution as though it referenced an individual `media.flv` object: one `media.flv` accessible through one distribution, and another `media.flv` accessible through the other distribution.

Restricting Access Using Crossdomain.xml

The Adobe Flash Media Server `crossdomain.xml` file specifies which domains can access media files in a particular domain. CloudFront supplies a default file that allows all domains to access the media files in your RTMP distribution, and you cannot change this behavior. If you include a more restrictive `crossdomain.xml` file in your Amazon S3 bucket, CloudFront ignores it.

Error Codes for RTMP Distributions

The following table lists the error codes that CloudFront can send to your media player. The errors are part of the string returned with `Event.info.application.message` or `Event.info.description`.

| Error | Description |
|---------------------------------------|---|
| <code>DistributionNotFound</code> | The distribution was not found. |
| <code>DistributionTypeMismatch</code> | The distribution is not an RTMP distribution. |
| <code>InvalidInstance</code> | The instance is invalid. |
| <code>InvalidURI</code> | The URI is invalid. |

Troubleshooting RTMP Distributions

If you're having trouble getting your media files to play, check the following items.

| Item to Check | Description |
|---|---|
| Separate distributions for the media player files and media files | The media player must be served by a regular HTTP distribution (for example, domain name <code>d111111abcdef8.cloudfront.net</code>), and media files must be served by an RTMP distribution (for example, domain name <code>s5c39gqb8ow64r.cloudfront.net</code>). Make sure you're not using the same distribution for both. |
| <code>/cfx/st</code> in the file path | Confirm that the path for the file includes <code>/cfx/st</code> . You don't need to include <code>/cfx/st</code> in the path to the object in the Amazon S3 bucket. For more information, see Configuring the Media Player (p. 139) . |
| File names in the file path | Some media players require that you include the file name extension (for example, <code>mp4:</code>) before the file name in the file path. Some media players also require that you exclude the file name extension (for example, <code>.mp4</code>) from the file path. For more information, see MPEG Files (p. 139) . Note The names of the media files in your Amazon S3 bucket must always include the applicable file name extension. |
| Port 1935 on your firewall | Adobe Flash Media Server uses port 1935 for RTMP. Make sure your firewall has this port open. If it doesn't, the typical message returned is "Unable to play video." You can also switch to RTMPT to tunnel over HTTP using port 80. |

| Item to Check | Description |
|------------------------------|--|
| Adobe Flash Player messaging | <p>By default, the Adobe Flash Player won't display a message if the video file it's trying to play is missing. Instead, it waits for the file to show up. You might want to change this behavior to give your end users a better experience.</p> <p>To instead have the player send a message if the video is missing, use <code>play("vid", 0, -1)</code> instead of <code>play("vid")</code>.</p> |

Working with Objects

Topics

- [Format of URLs for Objects \(p. 143\)](#)
- [How CloudFront Processes HTTP and HTTPS Requests \(p. 144\)](#)
- [Increasing the Proportion of Requests that Are Served from CloudFront Edge Caches \(p. 145\)](#)
- [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#)
- [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#)
- [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#)
- [Forwarding Custom Headers to Your Origin \(Web Distributions Only\) \(p. 156\)](#)
- [Adding, Removing, or Replacing Objects in a Distribution \(p. 158\)](#)
- [Customizing Error Responses \(p. 172\)](#)
- [How CloudFront Processes Partial Requests for an Object \(Range GETs\) \(p. 176\)](#)
- [Specifying a Default Root Object \(Web Distributions Only\) \(p. 177\)](#)
- [Serving Compressed Files \(p. 179\)](#)

This section describes how you work with objects in CloudFront.

Format of URLs for Objects

The domain name that you use in the URLs for objects on your web pages or in your web application can be either of the following:

- The domain name, such as `d111111abcdef8.cloudfront.net`, that CloudFront automatically assigns when you create a distribution
- Your own domain name, such as `example.com`

For example, you might use one of the following URLs to return the file `image.jpg`:

`http://d111111abcdef8.cloudfront.net/images/image.jpg`

`http://example.com/images/image.jpg`

You use the same URL format whether you store the content in Amazon S3 buckets or at a custom origin, like one of your own web servers.

Note

The URL format depends in part on the value that you specify for **Origin Path** in your distribution. This value gives CloudFront a top directory path for your objects. For more information about setting the origin path when you create a web distribution, see [Origin Path \(p. 67\)](#).

For more information about URL formats, see the following sections.

Topics

- [Using Your Own Domain Name \(Example.com\) \(p. 144\)](#)
- [Using a Trailing Slash \(/\) in URLs \(p. 144\)](#)
- [Creating URLs for Streaming \(p. 144\)](#)
- [Creating Signed URLs for Restricted Content \(p. 144\)](#)

Using Your Own Domain Name (Example.com)

Instead of using the default domain name that CloudFront assigns for you when you create a distribution, you can [add an alternate domain name](#) that's easier to work with, like `example.com`. By setting up your own domain name with CloudFront, you can use a URL like this for objects in your distribution:

`http://example.com/images/image.jpg`

If you plan to use HTTPS between viewers and CloudFront, see [Using Alternate Domain Names and HTTPS \(p. 283\)](#).

Using a Trailing Slash (/) in URLs

When you specify URLs for directories in your CloudFront distribution, choose either to always use a trailing slash or to never use a trailing slash. For example, choose only one of the following formats for all of your URLs:

`http://d111111abcdef8.cloudfront.net/images/`

`http://d111111abcdef8.cloudfront.net/images`

Why does it matter?

Both formats work to link to CloudFront objects, but being consistent can help prevent issues when you want to invalidate a directory later. CloudFront stores URLs exactly as they are defined, including trailing slashes. So if your format is inconsistent, you'll need to invalidate directory URLs with and without the slash, to ensure that CloudFront removes the directory.

It's inconvenient to have to invalidate both URL formats, and it can lead to additional costs. That's because if you must double up invalidations to cover both types of URLs, you might reach the limit for free invalidations for the month. And if that happens, you'll have to pay for all the invalidations, even if only one format for each directory URL exists in CloudFront.

Creating URLs for Streaming

If your CloudFront distribution serves streaming content, you must include additional characters in the URLs for your objects. For more information, see [Configuring the Media Player \(p. 139\)](#).

Creating Signed URLs for Restricted Content

If you have content that you want to restrict access to, you can create signed URLs. For example, if you want to distribute your content only to users who have authenticated, you can create URLs that are valid only for a specified time period or that are available only from a specified IP address. For more information, see [Serving Private Content through CloudFront \(p. 207\)](#).

How CloudFront Processes HTTP and HTTPS Requests

For Amazon S3 origins, CloudFront accepts requests in both HTTP and HTTPS protocols for objects in a CloudFront distribution by default. CloudFront then forwards the requests to your Amazon S3 bucket using the same protocol in which the requests were made.

For custom origins, when you create your distribution, you can specify how CloudFront accesses your origin: HTTP only, or matching the protocol that is used by the viewer. For more information about how CloudFront handles HTTP and HTTPS requests for custom origins, see [Protocols \(p. 198\)](#).

For information about how to restrict your web distribution so that end users can only access objects using HTTPS, see [Using HTTPS with CloudFront \(p. 278\)](#). (This option doesn't apply to RTMP distributions, which use the RTMP protocol.)

Note

The charge for HTTPS requests is higher than the charge for HTTP requests. For more information about billing rates, go to the [CloudFront pricing plan](#).

Increasing the Proportion of Requests that Are Served from CloudFront Edge Caches

One of the purposes of using CloudFront is to reduce the number of requests that your origin server responds to. This reduces the load on your origin server and also reduces latency because more objects are served from CloudFront edge locations, which are closer to your users. The more requests that CloudFront is able to serve from edge caches as a proportion of all requests (the greater the cache hit ratio), the fewer requests that CloudFront needs to forward to your origin to get the latest version or a unique version of an object.

You can view the percentage of viewer requests that are hits, misses, and errors in the CloudFront console. For more information, see [CloudFront Cache Statistics Reports \(p. 16\)](#) in the *Amazon CloudFront Developer Guide*.

The following sections explain how to improve your cache hit ratio.

Topics

- [Specifying How Long CloudFront Caches Your Objects \(p. 145\)](#)
- [Caching Based on Query String Parameters \(p. 145\)](#)
- [Caching Based on Cookie Values \(p. 146\)](#)
- [Caching Based on Request Headers \(p. 147\)](#)
- [Serving Media Content by Using HTTP \(p. 147\)](#)

Specifying How Long CloudFront Caches Your Objects

To increase your cache hit ratio, you can configure your origin to add a `Cache-Control max-age` directive to your objects, and specify the longest practical value for `max-age`. The shorter the cache duration, the more frequently CloudFront forwards another request to your origin to determine whether the object has changed and, if so, to get the latest version. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

Caching Based on Query String Parameters

If you configure CloudFront to cache based on query string parameters, you can improve caching if you do the following:

- Configure CloudFront to forward only the query string parameters for which your origin will return unique objects.

- Use the same case (uppercase or lowercase) for all instances of the same parameter. For example, if one request contains `parameter1=A` and another contains `parameter1=a`, CloudFront forwards separate requests to your origin when a request contains `parameter1=A` and when a request contains `parameter1=a`. CloudFront then separately caches the corresponding objects returned by your origin separately even if the objects are identical. If you use just `A` or `a`, CloudFront forwards fewer requests to your origin.
- List parameters in the same order. As with differences in case, if one request for an object contains the query string `parameter1=a¶meter2=b` and another request for the same object contains `parameter2=b¶meter1=a`, CloudFront forwards both requests to your origin and separately caches the corresponding objects even if they're identical. If you always use the same order for parameters, CloudFront forwards fewer requests to your origin.

For more information, see [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#). If you want to review the query strings that CloudFront forwards to your origin, enable CloudFront access logs and see the values in the `cs-uri-query` column of your log files. For more information, see [Access Logs \(p. 315\)](#).

Caching Based on Cookie Values

If you configure CloudFront to cache based on cookie values, you can improve caching if you do the following:

- Configure CloudFront to forward only specified cookies instead of forwarding all cookies. For the cookies that you configure CloudFront to forward to your origin, CloudFront forwards every combination of cookie name and value, and separately caches the objects that your origin returns, even if they're all identical.

For example, suppose that viewers include two cookies in every request, that each cookie has three possible values, and that all combinations of cookie values are possible. CloudFront forwards up to six different requests to your origin for each object. If your origin returns different versions of an object based on only one of the cookies, then CloudFront is forwarding more requests to your origin than necessary and is needlessly caching multiple identical versions of the object.

- Create separate cache behaviors for static and dynamic content, and configure CloudFront to forward cookies to your origin only for dynamic content.

For example, suppose you have just one cache behavior for your distribution and that you're using the distribution both for dynamic content, such as `.js` files, and for `.css` files that rarely change. CloudFront caches separate versions of your `.css` files based on cookie values, so each CloudFront edge location forwards a request to your origin for every new cookie value or combination of cookie values.

If you create a cache behavior for which the path pattern is `*.css` and for which CloudFront doesn't cache based on cookie values, then CloudFront forwards requests for `.css` files to your origin only for the first request that an edge location receives for a given `.css` file and for the first request after a `.css` file expires.

- If possible, create separate cache behaviors for dynamic content for which cookie values are unique for each user (such as a user ID) and dynamic content that varies based on a smaller number of unique values.

For more information, see [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#). If you want to review the cookies that CloudFront forwards to your origin, enable CloudFront access logs and see the values in the `cs(Cookie)` column of your log files. For more information, see [Access Logs \(p. 315\)](#).

Caching Based on Request Headers

If you configure CloudFront to cache based on request headers, you can improve caching if you do the following:

- Configure CloudFront to forward and cache based only specified headers instead of forwarding and caching based on all headers. For the headers that you specify, CloudFront forwards every combination of header name and value and separately caches the objects that your origin returns even if they're all identical.

Note

CloudFront always forwards to your origin the headers specified in the following topics:

- How CloudFront Processes and Forwards Requests to Your Amazon S3 Origin Server > [HTTP Request Headers That CloudFront Removes or Updates \(p. 186\)](#)
- How CloudFront Processes and Forwards Requests to Your Custom Origin Server > [HTTP Request Headers and CloudFront Behavior \(p. 194\)](#)

When you configure CloudFront to cache based on request headers, you don't change the headers that CloudFront forwards, only whether CloudFront caches objects based on the header values.

- Try to avoid caching based on request headers that have large numbers of unique values.

For example, if you want to serve different sizes of an image based on the user's device, then don't configure CloudFront to cache based on the `User-Agent` header, which has an enormous number of possible values. Instead, configure CloudFront to cache based on the CloudFront device-type headers `CloudFront-Is-Desktop-Viewer`, `CloudFront-Is-Mobile-Viewer`, `CloudFront-Is-SmartTV-Viewer`, and `CloudFront-Is-Tablet-Viewer`. In addition, if you're returning the same version of the image for tablets and desktops, then forward only the `CloudFront-Is-Tablet-Viewer` header, not the `CloudFront-Is-Desktop-Viewer` header.

For more information, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

Serving Media Content by Using HTTP

When you use HTTP to serve media content, we recommend that you use an HTTP-based dynamic streaming protocol such as Apple HTTP Dynamic Streaming (Apple HDS), Apple HTTP Live Streaming (Apple HLS), Microsoft Smooth Streaming, or MPEG-DASH. For dynamic-streaming protocols, a video is divided into a lot of small segments that are typically just a few seconds long each. If your users commonly stop watching before the end of a video (for example, because they close their viewer during the credits), CloudFront has still cached all of the small segments up to that point in the video. If you're using a protocol for which a video is served in a single, large file and a user stops watching before the end, CloudFront might not cache the entire video, and it might need to request the video from your origin again the next time that CloudFront receives a request for it.

Configuring CloudFront to Cache Based on Query String Parameters

Some web applications use query strings to send information to the origin. A query string is the part of a web request that appears after a ? character; the string can contain one or more parameters separated by & characters. In the following example, the query string includes two parameters, `color=red` and `size=large`:

`http://d111111abcdef8.cloudfront.net/images/image.jpg?color=red&size=large`

For web distributions, you can choose whether you want CloudFront to forward query strings to your origin and, if so, whether to cache your content based on all parameters or on selected parameters.

Suppose your website is available in five languages. The directory structure and file names for all five versions of the website are identical. As a user views your website, requests that are forwarded to CloudFront include a language query string parameter based on the language that the user chose. You can configure CloudFront to forward query strings to the origin and to cache based on the language parameter. If you configure your web server to return the version of a given page that corresponds with the selected language, CloudFront will cache each language version separately, based on the value of the language query string parameter.

In this example, if the main page for your website is main.html, the following five requests will cause CloudFront to cache main.html five times, once for each value of the language query string parameter:

- `http://d111111abcdef8.cloudfront.net/main.html?language=de`
- `http://d111111abcdef8.cloudfront.net/main.html?language=en`
- `http://d111111abcdef8.cloudfront.net/main.html?language=es`
- `http://d111111abcdef8.cloudfront.net/main.html?language=fr`
- `http://d111111abcdef8.cloudfront.net/main.html?language=jp`

Note the following:

- For RTMP distributions, you cannot configure CloudFront to forward query string parameters to your origin. Before CloudFront forwards a request to the origin server, it removes any query string parameters.
- Some HTTP servers don't process query string parameters and, therefore, don't return different versions of an object based on parameter values. For these origins, if you configure CloudFront to forward query string parameters to the origin, CloudFront will still cache based on the parameter values even though the origin returns identical versions of the object to CloudFront for every parameter value.
- You must use the & character as the delimiter between query string parameters. If you use a different delimiter, caching depends on which parameters you want CloudFront to use as a basis for caching and the order in which they appear in the query string. The following examples show what happens if you configure CloudFront to cache based only on the color parameter:
 - In the following request, CloudFront caches your content based on the value of the color parameter, but CloudFront interprets the value as `red;size=large`:
`http://d111111abcdef8.cloudfront.net/images/image.jpg?color=red;size=large`
 - In the following request, CloudFront caches your content but doesn't base caching on the query string parameters. This is because you configured CloudFront to cache based on the color parameter, but CloudFront interprets the following string as containing only a size parameter that has a value of `large;color=red`:
`http://d111111abcdef8.cloudfront.net/images/image.jpg?size=large;color=red`

You can configure CloudFront do to one of the following:

- Don't forward query strings to the origin at all. If you don't forward query strings, CloudFront doesn't cache based on query string parameters.
- Forward query strings to the origin, and cache based on all parameters in the query string.
- Forward query strings to the origin, and cache based on specified parameters in the query string.

For more information, see [Optimizing Caching \(p. 149\)](#).

Topics

- [Console and API Settings for Query String Forwarding and Caching \(p. 149\)](#)
- [Optimizing Caching \(p. 149\)](#)
- [Query String Parameters and CloudFront Access Logs \(p. 150\)](#)

Console and API Settings for Query String Forwarding and Caching

To configure query string forwarding and caching in the CloudFront console, see the following settings in [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#):

- [Query String Forwarding and Caching \(p. 76\)](#)
- [Query String Whitelist \(p. 76\)](#)

To configure query string forwarding and caching with the CloudFront API, see the following settings in [DistributionConfig Complex Type](#) and in [DistributionConfigWithTags Complex Type](#) in the [Amazon CloudFront API Reference](#):

- `QueryString`
- `QueryStringCacheKeys`

Optimizing Caching

When you configure CloudFront to cache based on query string parameters, here's how you can reduce the number of requests that CloudFront forwards to your origin, which reduces the load on your origin server and also reduces latency because more objects are served from CloudFront edge locations.

Cache Based Only on Parameters for Which Your Origin Returns Different Versions of an Object

For each query string parameter that your web application forwards to CloudFront, CloudFront forwards requests to your origin for every parameter value and caches a separate version of the object for every parameter value. This is true even if your origin always returns the same object regardless of the parameter value. For multiple parameters, the number of requests and the number of objects multiply: if requests for an object include two parameters that each have three different values, CloudFront will cache six versions of that object, assuming you follow the other recommendations in this section.

We recommend that you configure CloudFront to cache based only on the query string parameters for which your origin returns different versions, and that you carefully consider the merits of caching based on each parameter. For example, suppose you have a retail website. You have pictures of a jacket in six different colors, and the jacket comes in ten different sizes. The pictures that you have of the jacket show the different colors but not the different sizes. To optimize caching, you should configure CloudFront to cache based only on the color parameter, not on the size parameter. This increases the likelihood that CloudFront can serve a request from the cache, which improves performance and reduces the load on your origin.

Always List Parameters in the Same Order

The order of parameters matters in query strings. In the following example, the query strings are identical except that the parameters are in a different order. This will cause CloudFront to forward two separate requests for `image.jpg` to your origin and to cache two separate versions of the object:

- `http://d111111abcdef8.cloudfront.net/images/image.jpg?color=red&size=large`
- `http://d111111abcdef8.cloudfront.net/images/image.jpg?size=large&color=red`

We recommend that you always list parameter names in the same order, such as alphabetical order.

Always Use the Same Case for Parameter Names and Values

CloudFront considers the case of parameter names and values when caching based on query string parameters. In the following example, the query strings are identical except for the case of parameter names and values. This will cause CloudFront to forward four separate requests for image.jpg to your origin and to cache four separate versions of the object:

- `http://d111111abcdef8.cloudfront.net/images/image.jpg?color=red`
- `http://d111111abcdef8.cloudfront.net/images/image.jpg?color=Red`
- `http://d111111abcdef8.cloudfront.net/images/image.jpg?Color=red`
- `http://d111111abcdef8.cloudfront.net/images/image.jpg?Color=Red`

We recommend that you use case consistently for parameter names and values, such as all lowercase.

Don't Use Parameter Names that Conflict with Signed URLs

If you're using signed URLs to restrict access to your content (if you added trusted signers to your distribution), CloudFront removes the following query string parameters before forwarding the rest of the URL to your origin:

- `Expires`
- `Key-Pair-Id`
- `Policy`
- `Signature`

If you're using signed URLs and you want to configure CloudFront to forward query strings to your origin, your own query string parameters cannot be named `Expires`, `Key-Pair-Id`, `Policy`, or `Signature`.

Query String Parameters and CloudFront Access Logs

For web and RTMP distributions, if you enable logging, CloudFront logs the full URL, including query string parameters. For web distributions, this is true regardless of whether you have configured CloudFront to forward query strings to the origin. For more information about CloudFront logging, see [Access Logs \(p. 315\)](#).

Configuring CloudFront to Cache Objects Based on Cookies

For web distributions, you can choose whether you want CloudFront to forward cookies to your origin and to cache separate versions of your objects based on cookie values in viewer requests.

For Real Time Messaging Protocol (RTMP) distributions, you cannot configure CloudFront to process cookies. When CloudFront requests an object from the origin server, it removes any cookies before forwarding the request to your origin. If your origin returns any cookies along with the object, CloudFront removes them before returning the object to the viewer. For RTMP distributions, CloudFront does not cache cookies in edge caches.

Important

Amazon S3 and some HTTP servers do not process cookies. Do not configure CloudFront cache behaviors to forward cookies to an origin that doesn't process cookies, or you'll adversely affect cacheability and, therefore, performance. For more information about cache behaviors, see [Cache Behavior Settings \(p. 70\)](#).

For HTTP and HTTPS web distributions, you can choose whether you want CloudFront to forward cookies to your origin. For RTMP distributions, you cannot configure CloudFront to process cookies.

For web distributions, CloudFront by default doesn't consider cookies when caching your objects in edge locations. If your origin returns two objects and they differ only by the values in the Set-Cookie header, CloudFront caches only one version of the object.

You can configure CloudFront to forward to your origin some or all of the cookies in viewer requests. CloudFront uses the cookies in viewer requests to uniquely identify an object in the cache. For example, suppose that requests for `locations.html` contain a country cookie that has a value of either `uk` or `fr`. When you configure CloudFront to cache your objects based on the value of the country cookie, CloudFront forwards requests for `locations.html` to the origin and includes the country cookie and cookie values. Your origin returns `locations.html`, and CloudFront caches the object once for requests in which the value of the country cookie is `uk` and once for requests in which the value is `fr`.

Note

If you configure CloudFront to forward cookies to your origin, CloudFront caches based on cookie values. This is true even if your origin ignores the cookie values in the request and, in the previous example, always returns the same version of `locations.html` to CloudFront. As a result, CloudFront forwards more requests to your origin server for the same object, which slows performance and increases the load on your origin server. If your origin server does not vary its response based on the value of a given cookie, we recommend that you do not configure CloudFront to forward that cookie to your origin.

You can configure each cache behavior in a web distribution to do one of the following:

- **Forward all cookies to your origin** – CloudFront forwards viewer requests to your origin, including all cookies. When your origin returns a response, CloudFront caches the response, and the cookies and cookie values in the viewer request. (If your origin returns cookies that were not in the viewer request, CloudFront does not cache them.) CloudFront returns to the viewer the requested object and all cookies and cookie values, including cookies that were not in the viewer request.
- **Forward a whitelist of cookies that you specify** – CloudFront removes any cookies that aren't on the whitelist before forwarding requests to your origin. CloudFront caches the response from your origin as well as the specified cookies and their values. (If your origin returns both whitelisted cookies and cookies that aren't on your whitelist, CloudFront caches only the whitelisted cookies.) CloudFront also returns to the viewer the object, including the specified cookies and cookie values. If the response from the origin includes cookies that aren't on the whitelist, CloudFront returns those cookies to the viewer, too.

For information about specifying wildcards in cookie names, see [Whitelist Cookies \(Amazon EC2 and Other Custom Origins Only\) \(p. 75\)](#).

For the current limit on the number of cookie names that you can whitelist for each cache behavior, see [Amazon CloudFront Limits](#) in the *Amazon Web Services General Reference*. To request a higher limit, go to <https://console.aws.amazon.com/support/home#/case/create?issueType=service-limit-increase&limitType=service-code-cloudfront-distributions>.

- **Don't forward cookies to your origin** – CloudFront doesn't cache your objects based on cookie values. In addition, CloudFront removes the Cookie header from requests that it forwards to your origin and removes the Set-Cookie header from responses that it returns to your viewers.

Note the following about specifying the cookies that you want to forward:

Access Logs

If you configure CloudFront to log requests and to log cookies, CloudFront logs all cookies and all cookie attributes, even if you configure CloudFront not to forward cookies to your origin or if you configure CloudFront to forward only a specified list of cookies. For more information about CloudFront logging, see [Access Logs \(p. 315\)](#).

Case Sensitivity

Cookie names and values are both case sensitive. For example, if two cookies for the same object are identical except for case, CloudFront will cache the object twice.

CloudFront Sorts Cookies

CloudFront sorts the cookies in natural order by cookie name before forwarding the request to your origin.

If-Modified-Since and If-None-Match

If-Modified-Since and If-None-Match conditional requests are not supported.

Suspending Caching Based on Cookies

If you want CloudFront to temporarily stop caching cookies and cookie attributes, configure your origin server to add the following header in responses to CloudFront:

`no-cache="Set-Cookie"`

Total Length of Cookie Names

The total number of bytes in all of the cookie names that you configure CloudFront to forward to your origin can't exceed:

`512 - (the number of cookies that you're forwarding)`

For example, if you configure CloudFront to forward 10 cookies to your origin, the combined length of the names of the 10 cookies can't exceed 502 bytes (512 – 10). If you configure CloudFront to forward all cookies to your origin, the length of cookie names doesn't matter.

For information about using the CloudFront console to update a distribution so CloudFront forwards cookies to the origin, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#). For information about using the CloudFront API to update a distribution, see [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

Configuring CloudFront to Cache Objects Based on Request Headers

For web distributions, CloudFront lets you choose whether you want CloudFront to forward headers to your origin and to cache separate versions of a specified object based on the header values in viewer requests. This allows you to serve different versions of your content based on the device the user is using, the location of the viewer, the language the viewer is using, and a variety of other criteria. For RTMP distributions, you cannot configure CloudFront to cache based on header values.

Topics

- [Headers and Web Distributions \(p. 152\)](#)
- [Headers and RTMP Distributions \(p. 156\)](#)

Headers and Web Distributions

By default, CloudFront doesn't consider headers when caching your objects in edge locations. If your origin returns two objects and they differ only by the values in the request headers, CloudFront caches only one version of the object.

You can configure CloudFront to forward headers to the origin, which causes CloudFront to cache multiple versions of an object based on the values in one or more request headers. To configure CloudFront to cache objects based on the values of specific headers, you specify cache behavior settings for your distribution. For more information, see [Cache Based on Selected Request Headers](#).

For example, suppose viewer requests for `logo.jpg` contain a custom `Product` header that has a value of either `Acme` or `Apex`. When you configure CloudFront to cache your objects based on the value of the `Product` header, CloudFront forwards requests for `logo.jpg` to the origin and includes the `Product` header and header values. CloudFront caches `logo.jpg` once for requests in which the value of the `Product` header is `Acme` and once for requests in which the value is `Apex`.

You can configure each cache behavior in a web distribution to do one of the following:

- Forward all headers to your origin

Important

If you configure CloudFront to forward all headers to your origin, CloudFront doesn't cache the objects associated with this cache behavior. Instead, it sends every request to the origin.

- Forward a whitelist of headers that you specify. CloudFront caches your objects based on the values in all of the specified headers. CloudFront also forwards the headers that it forwards by default, but it caches your objects based only on the headers that you specify.
- Forward only the default headers. In this configuration, CloudFront doesn't cache your objects based on the values in the request headers.

For the current limit on the number of headers that you can whitelist for each cache behavior, see [Amazon CloudFront Limits](#) in the *Amazon Web Services General Reference*. To request a higher limit, go to <https://console.aws.amazon.com/support/home#/case/create?issueType=service-limit-increase&limitType=service-code-cloudfront-distributions>.

For information about using the CloudFront console to update a distribution so CloudFront forwards headers to the origin, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#). For information about using the CloudFront API to update an existing web distribution, see [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

Topics

- [Selecting the Headers on Which You Want CloudFront to Base Caching \(p. 153\)](#)
- [Configuring CloudFront to Respect Cross-Origin Resource Sharing \(CORS\) Settings \(p. 154\)](#)
- [Configuring CloudFront to Cache Objects Based on the Device Type \(p. 154\)](#)
- [Configuring CloudFront to Cache Objects Based on the Language of the Viewer \(p. 155\)](#)
- [Configuring CloudFront to Cache Objects Based on the Location of the Viewer \(p. 155\)](#)
- [Configuring CloudFront to Cache Objects Based on the Protocol of the Request \(p. 155\)](#)
- [How Caching Based on Headers Affects Performance \(p. 155\)](#)
- [How the Case of Headers and Header Values Affects Caching \(p. 155\)](#)
- [Headers that CloudFront Returns to the Viewer \(p. 156\)](#)

Selecting the Headers on Which You Want CloudFront to Base Caching

The headers that you can forward to the origin and that CloudFront bases caching on depend on whether you're using an Amazon S3 bucket or a custom origin.

- **Amazon S3** – You can configure CloudFront to forward and to cache your objects based only on three headers: `Access-Control-Request-Headers`, `Access-Control-Request-Method`, and `Origin`.

Forwarding these headers allows CloudFront to distribute content for websites that are enabled for cross-origin resource sharing (CORS). You can't configure CloudFront to forward custom headers to Amazon S3. For more information, see [Configuring CloudFront to Respect Cross-Origin Resource Sharing \(CORS\) Settings \(p. 154\)](#).

- **Custom origin** – You can configure CloudFront to cache based on the value of any request header except the following:
 - Accept-Encoding
 - Connection
 - Cookie – If you want to forward and cache based on cookies, you use a separate setting in your distribution. For more information, see [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).
 - Proxy-Authorization
 - TE
 - Upgrade

You can configure CloudFront to cache objects based on values in the Date and User-Agent headers, but we don't recommend it. These headers have a lot of possible values, and caching based on their values would cause CloudFront to forward significantly more requests to your origin.

For a full list of HTTP request headers and how CloudFront processes them, see the applicable topic:

- [Amazon S3 origin – HTTP Request Headers That CloudFront Removes or Updates \(p. 186\)](#)
- [Custom origin – HTTP Request Headers and CloudFront Behavior \(p. 194\)](#)

Configuring CloudFront to Respect Cross-Origin Resource Sharing (CORS) Settings

If you enabled cross-origin resource sharing (CORS) on an Amazon S3 bucket or a custom origin, you can configure CloudFront to respect the CORS settings. Configure CloudFront to forward a whitelist of headers and include the applicable headers in the list of headers to forward, depending on whether you're using Amazon S3 or a custom origin:

Amazon S3

- When you want OPTIONS responses to be cached, configure CloudFront to forward the following headers: Origin, Access-Control-Request-Headers, and Access-Control-Request-Method.
- When you don't want OPTIONS responses to be cached, configure CloudFront to forward the Origin header. You can also configure CloudFront to forward Access-Control-Request-Headers and Access-Control-Request-Method headers, but it's not required.

Custom origins – Forward the Origin header along with any other headers required by your origin.

For more information about CORS and Amazon S3, see [Enabling Cross-Origin Resource Sharing](#) in the [Amazon Simple Storage Service Developer Guide](#).

Configuring CloudFront to Cache Objects Based on the Device Type

If you want CloudFront to cache different versions of your objects based on the device a user is using to view your content, configure CloudFront to forward the applicable headers to your custom origin:

- CloudFront-Is-Desktop-Viewer

- `CloudFront-Is-Mobile-Viewer`
- `CloudFront-Is-SmartTV-Viewer`
- `CloudFront-Is-Tablet-Viewer`

Based on the value of the `User-Agent` header, CloudFront sets the value of these headers to `true` or `false` before forwarding the request to your origin. If a device falls into more than one category, more than one value might be `true`. For example, for some tablet devices, CloudFront might set both `CloudFront-Is-Mobile-Viewer` and `CloudFront-Is-Tablet-Viewer` to `true`.

Configuring CloudFront to Cache Objects Based on the Language of the Viewer

If you want CloudFront to cache different versions of your objects based on the language specified in the request, program your application to include the language in the `Accept-Language` header, and configure CloudFront to forward the `Accept-Language` header to your origin.

Configuring CloudFront to Cache Objects Based on the Location of the Viewer

If you want CloudFront to cache different versions of your objects based on the country that the request came from, configure CloudFront to forward the `CloudFront-Viewer-Country` header to your origin. CloudFront automatically converts the IP address that the request came from into a two-letter country code. For an easy-to-use list of country codes, sortable by code and by country name, see the Wikipedia entry [ISO 3166-1 alpha-2](#).

Configuring CloudFront to Cache Objects Based on the Protocol of the Request

If you want CloudFront to cache different versions of your objects based on the protocol of the request, HTTP or HTTPS, configure CloudFront to forward the `CloudFront-Forwarded-Proto` header to your origin.

How Caching Based on Headers Affects Performance

When you configure CloudFront to cache based on one or more headers and the headers have more than one possible value, CloudFront forwards more requests to your origin server for the same object. This slows performance and increases the load on your origin server. If your origin server returns the same object regardless of the value of a given header, we recommend that you don't configure CloudFront to cache based on that header.

If you configure CloudFront to forward more than one header, the order of the headers in viewer requests doesn't affect caching as long as the values are the same. For example, if one request contains the headers `A:1,B:2` and another request contains `B:2,A:1`, CloudFront caches just one copy of the object.

How the Case of Headers and Header Values Affects Caching

When CloudFront caches based on header values, it doesn't consider the case of the header name, but it does consider the case of the header value:

- If viewer requests include both `Product:Acme` and `product:Acme`, CloudFront caches an object only once. The only difference between them is the case of the header name, which doesn't affect caching.
- If viewer requests include both `Product:Acme` and `Product:acme`, CloudFront caches an object twice, because the value is `Acme` in some requests and `acme` in others.

Headers that CloudFront Returns to the Viewer

Configuring CloudFront to forward and cache headers does not affect which headers CloudFront returns to the viewer. CloudFront returns all of the headers that it gets from the origin with a few exceptions. For more information, see the applicable topic:

- **Amazon S3 origins** – See [HTTP Response Headers That CloudFront Removes or Updates \(p. 189\)](#).
- **Custom origins** – See [HTTP Response Headers that CloudFront Removes or Updates \(p. 201\)](#).

Headers and RTMP Distributions

For RTMP distributions, you cannot configure CloudFront to cache your content based on the headers in viewer requests.

Forwarding Custom Headers to Your Origin (Web Distributions Only)

You can configure CloudFront to include custom headers whenever it forwards a request to your origin. You can specify the names and values of custom headers for each origin, both for custom origins and for Amazon S3 buckets. Custom headers have a variety of uses, such as the following:

- You can identify the requests that are forwarded to your custom origin by CloudFront. This is useful if you want to know whether users are bypassing CloudFront or if you're using more than one CDN and you want information about which requests are coming from each CDN. (If you're using an Amazon S3 origin and you enable [Amazon S3 server access logging](#), the logs don't include header information.)
- If you've configured more than one CloudFront distribution to use the same origin, you can specify different custom headers for the origins in each distribution and use the logs for your web server to distinguish between the requests that CloudFront forwards for each distribution.
- If some of your users use viewers that don't support cross-origin resource sharing (CORS), you can configure CloudFront to forward the `Origin` header to your origin. That will cause your origin to return the `Access-Control-Allow-Origin` header for every request.
- You can use custom headers together and, optionally, signed URLs or signed cookies, to control access to content on a custom origin. If you configure your custom origin to respond to requests only if they include a custom header, you can prevent users from bypassing CloudFront and submitting requests directly to your origin.

Topics

- [Configuring CloudFront to Forward Custom Headers to Your Origin \(p. 156\)](#)
- [Custom Headers that CloudFront Can't Forward to Your Origin \(p. 157\)](#)
- [Using Custom Headers for Cross-Origin Resource Sharing \(CORS\) \(p. 157\)](#)
- [Using Custom Headers to Restrict Access to Your Content on a Custom Origin \(p. 158\)](#)

Configuring CloudFront to Forward Custom Headers to Your Origin

To configure a web distribution to forward custom headers to your origin, you update the configuration of the applicable origins by using one of the following methods:

CloudFront console

When you create or update a distribution, specify header names and values in the **Origin Custom Headers** settings. For more information, see [Creating or Updating a Web Distribution Using the CloudFront Console \(p. 60\)](#).

CloudFront API

For each origin that you want to forward custom headers to, add header names and values to the `CustomHeaders` section of the `DistributionConfig` complex type. For more information, see [Create Distribution](#) (to create a new distribution) or [Update Distribution](#) (to update an existing distribution).

If the header names and values that you specify are not already present in the viewer request, CloudFront adds them. If a header is present, CloudFront overwrites the header value before forwarding the request to the origin.

For the current limits related to forwarding custom headers to the origin, see [Limits \(p. 359\)](#).

Custom Headers that CloudFront Can't Forward to Your Origin

You can't configure CloudFront to forward the following custom headers to your origin.

| | |
|---------------------|----------------------------------|
| Accept-Encoding | Proxy-Authenticate |
| Cache-Control | Proxy-Authorization |
| Connection | Proxy-Connection |
| Content-Length | Range |
| Cookie | Request-Range |
| Host | TE |
| If-Match | Trailer |
| If-Modified-Since | Transfer-Encoding |
| If-None-Match | Upgrade |
| If-Range | Via |
| If-Unmodified-Since | Headers that begin with X-Amz-* |
| Max-Forwards | Headers that begin with X-Edge-* |
| Pragma | X-Real-Ip |

Using Custom Headers for Cross-Origin Resource Sharing (CORS)

You can configure CloudFront to always forward the applicable headers to your origin to accommodate viewers that don't automatically include those headers in requests. You also need to configure

CloudFront to respect CORS settings. For more information, see [Configuring CloudFront to Respect Cross-Origin Resource Sharing \(CORS\) Settings \(p. 154\)](#).

Using Custom Headers to Restrict Access to Your Content on a Custom Origin

If you're using a custom origin, you can use custom headers to prevent users from bypassing CloudFront and requesting content directly from your origin. You can also optionally restrict access to your content by requiring that your users access your objects by using either signed URLs or signed cookies. For more information about private content, see [Serving Private Content through CloudFront \(p. 207\)](#).

To require that users access your content through CloudFront, change the following settings in your CloudFront distributions:

Origin Custom Headers

Configure CloudFront to forward custom headers to your origin. See [Configuring CloudFront to Forward Custom Headers to Your Origin \(p. 156\)](#).

Viewer Protocol Policy

Configure your distribution to require viewers to use HTTPS to access CloudFront. See [Viewer Protocol Policy \(p. 72\)](#).

Origin Protocol Policy

Configure your distribution to require CloudFront to use the same protocol as viewers to forward requests to the origin. See [Origin Protocol Policy \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 68\)](#).

The combination of **Viewer Protocol Policy** and **Origin Protocol Policy** ensure that your custom headers are encrypted between the viewer and your origin. However, we recommend that you periodically perform the following tasks to rotate the custom headers that CloudFront forwards to your origin:

1. Update your CloudFront distribution to begin forwarding a new header to your custom origin.
2. Update your application to accept the new header as confirmation that the request is coming from CloudFront.
3. When viewer requests no longer include the header that you're replacing, update your application to no longer accept the old header as confirmation that the request is coming from CloudFront.

Adding, Removing, or Replacing Objects in a Distribution

For information about adding objects to a distribution, see [Adding Objects that You Want CloudFront to Distribute \(p. 159\)](#).

When you replace objects in your distribution, we recommend that you use versioned object names. For more information, see [Updating Existing Objects Using Versioned Object Names \(p. 159\)](#). You can also replace objects with objects that have the same name. See [Updating Existing Objects Using the Same Object Names \(p. 159\)](#). Regardless of how you choose to replace objects in your distribution, we recommend that you specify when objects should be removed from the CloudFront cache. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

If you need to quickly remove objects from a distribution, you can invalidate them. For more information, see [Invalidate Objects \(Web Distributions Only\) \(p. 165\)](#).

Adding Objects that You Want CloudFront to Distribute

When you want CloudFront to start distributing additional objects, you add the objects to one of the origins that you specified for the distribution, and you expose a CloudFront link to the objects. A CloudFront edge location doesn't fetch the new objects from an origin until the edge location receives viewer requests for the objects. For more information, see [How CloudFront Delivers Content \(p. 4\)](#).

When you add an object that you want CloudFront to distribute, ensure that you add it to one of the Amazon S3 buckets specified in your distribution or, for a custom origin, to a directory in the specified domain. In addition, confirm that the path pattern in the applicable cache behavior sends requests to the correct origin. For example, suppose the path pattern for a cache behavior is *.html. If no other cache behaviors are configured to forward requests to that origin, CloudFront will never distribute jpg files that you upload to the origin.

CloudFront servers don't determine the MIME type for the objects they serve. When you upload an object to your origin, you should set the `Content-Type` header field for the object.

Updating Existing Objects Using Versioned Object Names

When you update existing objects in a CloudFront distribution, we recommend that you include some sort of version identifier either in your object names or in your directory names to give yourself better control over your content. This identifier might be a date-time stamp, a sequential number, or some other method of distinguishing two versions of the same object.

For example, instead of naming a graphic file image.jpg, you might call it image_1.jpg. When you want to start serving a new version of the file, you'd name the new file image_2.jpg, and you'd update the links in your web application or website to point to image_2.jpg. Alternatively, you might put all graphics in an images_v1 directory and, when you want to start serving new versions of one or more graphics, you'd create a new images_v2 directory, and you'd update your links to point to that directory. With versioning, you don't have to wait for an object to expire before CloudFront begins to serve a new version of it, and you don't have to pay for object invalidation.

Even if you version your objects, we still recommend that you set an expiration date. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

Note

Specifying versioned object names or directory names is not related to Amazon S3 object versioning.

Updating Existing Objects Using the Same Object Names

Although you can update existing objects in a CloudFront distribution and use the same object names, we don't recommend it. CloudFront distributes objects to edge locations only when the objects are requested, not when you put new or updated objects in your origin. If you update an existing object in your origin with a newer version that has the same name, an edge location won't get that new version from your origin until both of the following occur:

- The old version of the object in the cache expires. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).
- There's an end user request for the object at that edge location.

If you use the same names when you replace objects, you can't control when CloudFront starts to serve the new files. By default, CloudFront caches objects in edge locations for 24 hours. (For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).) For example, if you're replacing all of the objects on an entire website:

- Objects for the less popular pages may not be in any edge locations. The new versions of these objects will start being served on the next request.
- Objects for some pages may be in some edge locations and not in others, so your end users will see different versions depending on which edge location they're served from.
- New versions of the objects for the most popular pages might not be served for up to 24 hours because CloudFront might have retrieved the objects for those pages just before you replaced the objects with new versions.

Specifying How Long Objects Stay in a CloudFront Edge Cache (Expiration)

You can control how long your objects stay in a CloudFront cache before CloudFront forwards another request to your origin. Reducing the duration allows you to serve dynamic content. Increasing the duration means your users get better performance because your objects are more likely to be served directly from the edge cache. A longer duration also reduces the load on your origin.

Typically, CloudFront serves an object from an edge location until the cache duration that you specified passes—that is, until the object expires. After it expires, the next time the edge location gets a user request for the object, CloudFront forwards the request to the origin server to verify that the cache contains the latest version of the object. The response from the origin depends on whether the object has changed:

- If the CloudFront cache already has the latest version, the origin returns a 304 status code (Not Modified).
- If the CloudFront cache does not have the latest version, the origin returns a 200 status code (OK) and the latest version of the object.

If an object in an edge location isn't frequently requested, CloudFront might evict the object—remove the object before its expiration date—to make room for objects that have been requested more recently.

By default, each object automatically expires after 24 hours. For web distributions, you can change the default behavior in two ways:

- To change the cache duration for all objects that match the same path pattern, you can change the CloudFront settings for **Minimum TTL**, **Maximum TTL**, and **Default TTL** for a cache behavior. For information about the individual settings, see [Minimum TTL](#), [Maximum TTL](#), and [Default TTL](#). To use these settings, you must choose the **Customize** option for the **Object Caching** setting.
- To change the cache duration for an individual object, you can configure your origin to add a `Cache-Control max-age` or `Cache-Control s-maxage` directive, or an `Expires` header field to the object. For more information, see [Using Headers to Control Cache Duration for Individual Objects \(p. 161\)](#).

For more information about how **Minimum TTL**, **Default TTL**, and **Maximum TTL** interact with `Cache-Control max-age` and `Cache-Control s-maxage` directives and the `Expires` header field, see [Specifying the Amount of Time that CloudFront Caches Objects for Web Distributions \(p. 162\)](#).

You can also control how long errors (for example, 404, Not Found) stay in a CloudFront cache before CloudFront tries again to get the requested object by forwarding another request to your origin. For

more information, see [How CloudFront Processes and Caches HTTP 4xx and 5xx Status Codes from Your Origin \(p. 203\)](#).

Topics

- [Using Headers to Control Cache Duration for Individual Objects \(p. 161\)](#)
- [Specifying the Amount of Time that CloudFront Caches Objects for Web Distributions \(p. 162\)](#)
- [Specifying the Minimum Time that CloudFront Caches Objects for RTMP Distributions \(p. 164\)](#)
- [Adding Headers to Your Objects Using the Amazon S3 Console \(p. 165\)](#)

Using Headers to Control Cache Duration for Individual Objects

You can use the `Cache-Control` and `Expires` headers to control how long objects stay in the cache. Settings for **Minimum TTL**, **Default TTL**, and **Maximum TTL** also affect cache duration, but here's an overview of how headers can affect cache duration:

- The `Cache-Control max-age` directive lets you specify how long (in seconds) that you want an object to remain in the cache before CloudFront gets the object again from the origin server. The minimum expiration time CloudFront supports is 0 seconds for web distributions and 3600 seconds for RTMP distributions. The maximum value is 100 years. Specify the value in the following format:

`Cache-Control: max-age=seconds`

For example, the following directive tells CloudFront to keep the associated object in the cache for 3600 seconds (one hour):

`Cache-Control: max-age=3600`

If you want objects to stay in CloudFront edge caches for a different duration than they stay in browser caches, you can use the `Cache-Control max-age` and `Cache-Control s-maxage` directives together. For more information, see [Specifying the Amount of Time that CloudFront Caches Objects for Web Distributions \(p. 162\)](#).

- The `Expires` header field lets you specify an expiration date and time using the format specified in [RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1 Section 3.3.1, Full Date](#), for example:

`Sat, 27 Jun 2015 23:59:59 GMT`

We recommend that you use the `Cache-Control max-age` directive instead of the `Expires` header field to control object caching. If you specify values both for `Cache-Control max-age` and for `Expires`, CloudFront uses only the value of `Cache-Control max-age`.

For more information, see [Specifying the Amount of Time that CloudFront Caches Objects for Web Distributions \(p. 162\)](#).

You cannot use the `HTTP Cache-Control` or `Pragma` header fields in a `GET` request from a viewer to force CloudFront to go back to the origin server for the object. CloudFront ignores those header fields in viewer requests.

For more information about the `Cache-Control` and `Expires` header fields, see the following sections in [RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1](#):

- [Section 14.9 Cache Control](#)
- [Section 14.21 Expires](#)

For an example of how to add Cache-Control and Expires header fields using the AWS SDK for PHP, see [Upload an Object Using the AWS SDK for PHP](#) in the *Amazon Simple Storage Service Developer Guide*. Some third-party tools are also able to add these fields.

Specifying the Amount of Time that CloudFront Caches Objects for Web Distributions

For web distributions, you can use Cache-Control or Expires headers, and CloudFront minimum, maximum, and default TTL values to control the amount of time in seconds that CloudFront keeps an object in the cache before forwarding another request to the origin. Headers values also determine how long a browser keeps an object in the cache before forwarding another request to CloudFront.

Important

If you configure CloudFront to forward all headers to your origin for a cache behavior, CloudFront never caches the associated objects. Instead, CloudFront forwards all requests for those objects to the origin. In that configuration, the value of minimum TTL must be 0. For more information, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

To specify values for Minimum TTL, Maximum TTL, and Default TTL, you must choose the **Customize** option for the [Object Caching](#) setting.

| Origin Configuration | Minimum TTL = 0 Seconds | Minimum TTL > 0 Seconds |
|---|--|--|
| The origin adds a Cache-Control max-age directive to objects | <p>CloudFront caching</p> <p>CloudFront caches objects for the lesser of the value of the Cache-Control max-age directive or the value of the CloudFront maximum TTL.</p> <p>Browser caching</p> <p>Browsers cache objects for the value of the Cache-Control max-age directive.</p> | <p>CloudFront caching</p> <p>CloudFront caching depends on the values of the CloudFront minimum TTL and maximum TTL and the Cache-Control max-age directive:</p> <ul style="list-style-type: none"> • Minimum TTL < max-age < maximum TTL CloudFront caches objects for the value of the Cache-Control max-age directive. • max-age < minimum TTL CloudFront caches objects for the value of the CloudFront minimum TTL. • max-age > maximum TTL CloudFront caches objects for the value of the CloudFront maximum TTL. <p>Browser caching</p> <p>Browsers cache objects for the value of the Cache-Control max-age directive.</p> |
| The origin does not add a Cache-Control max-age directive to objects | CloudFront caching | CloudFront caching |

| Origin Configuration | Minimum TTL = 0 Seconds | Minimum TTL > 0 Seconds |
|---|---|--|
| | <p>CloudFront caches objects for the value of the CloudFront default TTL.</p> <p>Browser caching Depends on the browser.</p> | <p>CloudFront caches objects for the greater of the value of the CloudFront minimum TTL or default TTL.</p> <p>Browser caching Depends on the browser.</p> |
| The origin adds Cache-Control max-age and Cache-Control s-maxage directives to objects | <p>CloudFront caching CloudFront caches objects for the lesser of the value of the Cache-Control s-maxage directive or the value of the CloudFront maximum TTL.</p> <p>Browser caching Browsers cache objects for the value of the Cache-Control max-age directive.</p> | <p>CloudFront caching CloudFront caching depends on the values of the CloudFront minimum TTL and maximum TTL and the Cache-Control s-maxage directive:</p> <ul style="list-style-type: none"> • Minimum TTL < s-maxage < maximum TTL CloudFront caches objects for the value of the Cache-Control s-maxage directive. • s-maxage < minimum TTL CloudFront caches objects for the value of the CloudFront minimum TTL. • s-maxage > maximum TTL CloudFront caches objects for the value of the CloudFront maximum TTL. <p>Browser caching Browsers cache objects for the value of the Cache-Control max-age directive.</p> |

| Origin Configuration | Minimum TTL = 0 Seconds | Minimum TTL > 0 Seconds |
|---|---|--|
| The origin adds an <code>Expires</code> header to objects | <p>CloudFront caching</p> <p>CloudFront caches objects until the date in the <code>Expires</code> header or for the value of the CloudFront maximum TTL, whichever is sooner.</p> <p>Browser caching</p> <p>Browsers cache objects until the date in the <code>Expires</code> header.</p> | <p>CloudFront caching</p> <p>CloudFront caching depends on the values of the CloudFront minimum TTL and maximum TTL and the <code>Expires</code> header:</p> <ul style="list-style-type: none"> • <code>Minimum TTL < Expires < maximum TTL</code> CloudFront caches objects until the date and time in the <code>Expires</code> header. • <code>Expires < minimum TTL</code> CloudFront caches objects for the value of the CloudFront minimum TTL. • <code>Expires > maximum TTL</code> CloudFront caches objects for the value of the CloudFront maximum TTL. <p>Browser caching</p> <p>Browsers cache objects until the date and time in the <code>Expires</code> header.</p> |
| Origin adds <code>Cache-Control: no-cache, no-store, and/or private</code> directives to objects | <p>CloudFront and browsers respect the headers.</p> <p>For an exception to how CloudFront handles the <code>Cache-Control: no-cache</code> header, see Simultaneous Requests for the Same Object (Traffic Spikes) (p. 199).</p> | <p>CloudFront caching</p> <p>CloudFront caches objects for the value of the CloudFront minimum TTL.</p> <p>Browser caching</p> <p>Browsers respect the headers.</p> |

For information about how to change settings for web distributions using the CloudFront console, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#). For information about how to change settings for web distributions using the CloudFront API, see [PUT Config](#).

Specifying the Minimum Time that CloudFront Caches Objects for RTMP Distributions

For RTMP distributions, CloudFront keeps objects in edge caches for 24 hours by default. You can add `Cache-Control` or `Expires` headers to your objects to change the amount of time that CloudFront keeps objects in edge caches before it forwards another request to the origin. The minimum duration is 3600 seconds (one hour). If you specify a lower value, CloudFront uses 3600 seconds.

Adding Headers to Your Objects Using the Amazon S3 Console

Note

Using the Amazon S3 console, you can only add headers to one object at a time, but with some third-party tools, you can add headers to multiple Amazon S3 objects at a time. For more information about third-party tools that support Amazon S3, perform a web search on [AWS s3 third party tools](#).

To add a Cache-Control or Expires header field to Amazon S3 objects using the Amazon S3 console

1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3>.
2. In the Amazon S3 console, in the buckets list, choose the name of the bucket that contains the files.
3. In the list of objects, select the check box for one or more objects that you want to add a header to.
4. Choose **More** and choose **Change metadata**.
5. In the **Key** list, choose **Cache-Control** or **Expires**, as applicable.
6. In the **Value** field, type the applicable value:
 - For a Cache-Control field, type:
max-age=number of seconds that you want objects to stay in a CloudFront edge cache
 - For an Expires field, type a date and time in HTML format.
7. Choose **Save**.

Invalidating Objects (Web Distributions Only)

If you need to remove an object from CloudFront edge caches before it expires, you can do one of the following:

- Invalidate the object from edge caches. The next time a viewer requests the object, CloudFront returns to the origin to fetch the latest version of the object.
- Use object versioning to serve a different version of the object that has a different name. For more information, see [Updating Existing Objects Using Versioned Object Names \(p. 159\)](#).

Important

You can invalidate most types of objects that are served by a web distribution, but you cannot invalidate media files in the Microsoft Smooth Streaming format when you have enabled Smooth Streaming for the corresponding cache behavior. In addition, you cannot invalidate objects that are served by an RTMP distribution.

To invalidate objects, you can specify either the path for individual objects or a path that ends with the * wildcard, which might apply to one object or to many, as shown in the following examples:

- /images/image1.jpg
- /images/image*
- /images/*

Note

If you use the AWS command line interface (CLI) for invalidating objects and you specify a path that includes the * wildcard, you must use quotes ("") around the path.

For example: `aws cloudfront create-invalidation --distribution-id $CDN_DISTRIBUTION_ID --paths "/*"`

You can submit a specified number of invalidation paths each month for free. If you submit more than the allotted number of invalidation paths in a month, you pay a fee for each invalidation path that you submit. For more information about the charges for invalidation, see [Paying for Object Invalidation \(p. 171\)](#).

Topics

- [Choosing Between Invalidating Objects and Using Versioned Object Names \(p. 166\)](#)
- [Determining Which Objects to Invalidate \(p. 166\)](#)
- [Specifying the Objects to Invalidate \(p. 166\)](#)
- [Invalidating Objects and Displaying Information about Invalidations \(p. 169\)](#)
- [Third-Party Tools for Invalidating Objects \(p. 171\)](#)
- [Invalidation Limits \(p. 171\)](#)
- [Paying for Object Invalidation \(p. 171\)](#)

Choosing Between Invalidating Objects and Using Versioned Object Names

To control the versions of objects that are served from your distribution, you can either invalidate objects or give them versioned file names. If you'll want to update your objects frequently, we recommend that you primarily use object versioning for the following reasons:

- Versioning enables you to control which object a request returns even when the user has a version cached either locally or behind a corporate caching proxy. If you invalidate the object, the user might continue to see the old version until it expires from those caches.
- CloudFront access logs include the names of your objects, so versioning makes it easier to analyze the results of object changes.
- Versioning provides a way to serve different versions of objects to different users.
- Versioning simplifies rolling forward and back between object revisions.
- Versioning is less expensive. You still have to pay for CloudFront to transfer new versions of your objects to edge locations, but you don't have to pay for invalidating objects.

For more information about object versioning, see [Updating Existing Objects Using Versioned Object Names \(p. 159\)](#).

Determining Which Objects to Invalidate

If you want to invalidate multiple objects such as all of the objects in a directory or all of the objects whose names begin with the same characters, you can include the `*` wildcard at the end of the invalidation path. For more information about using the `*` wildcard, see [Invalidation paths](#).

If you want to invalidate selected objects but your users don't necessarily access every object on your origin, you can determine which objects viewers have requested from CloudFront and invalidate only those objects. To determine which objects viewers have requested, enable CloudFront access logging. For more information about access logs, see [Access Logs \(p. 315\)](#).

Specifying the Objects to Invalidate

Whether you invalidate objects by using the CloudFront console or the CloudFront API, the requirements and limitations for specifying objects are the same. Note the following about specifying the objects that you want to invalidate.

Case sensitivity

Invalidation paths are case sensitive, so `/images/image.jpg` and `/images/Image.jpg` specify two different objects.

Changing the URI Using a Lambda Function

If your CloudFront distribution triggers a Lambda function on viewer request events, and if the function changes the URI of the requested object, you must invalidate both URLs to remove the object from CloudFront edge caches:

- The URI in the viewer request
- The URI after the function changed it

For example, suppose your Lambda function changes the URI for an object from this:

`http://d111111abcdef8.cloudfront.net/index.html`

to a URI that includes a language directory:

`http://d111111abcdef8.cloudfront.net/en/index.html`

To invalidate the object, you must specify the following paths:

- `index.html`
- `en/index.html`

For more information, see [Invalidation paths](#).

Default root object

To invalidate the default root object, specify the path the same way that you specify the path for any other object.

Distribution types

You can invalidate only objects that are associated with a web distribution.

Forwarding cookies

If you configured CloudFront to forward cookies to your origin, CloudFront edge caches might contain several versions of the object. When you invalidate an object, CloudFront invalidates every cached version of the object regardless of its associated cookies. You can't selectively invalidate some versions and not others based on the associated cookies. For more information, see [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).

Forwarding headers

If you configured CloudFront to forward a whitelist of headers to your origin and to cache based on the values of the headers, CloudFront edge caches might contain several versions of the object. When you invalidate an object, CloudFront invalidates every cached version of the object regardless of the header values. You can't selectively invalidate some versions and not others based on header values. (If you configure CloudFront to forward all headers to your origin, CloudFront doesn't cache your objects.) For more information, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

Forwarding query strings

If you configured CloudFront to forward query strings to your origin, you must include the query strings when invalidating objects, as shown in the following examples:

- `images/image.jpg?parameter1=a`
- `images/image.jpg?parameter1=b`

If client requests include five different query strings for the same object, you can either invalidate the object five times, once for each query string, or you can use the `*` wildcard in the invalidation path, as shown in the following example:

/images/image.jpg*

For more information about using wildcards in the invalidation path, see [Invalidation paths](#). For more information about query strings, see [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#). To determine which query strings are in use, you can enable CloudFront logging. For more information, see [Access Logs \(p. 315\)](#).

Limits

For information about limits on invalidations, see [Invalidation Limits \(p. 171\)](#).

Microsoft Smooth Streaming files

You cannot invalidate media files in the Microsoft Smooth Streaming format when you have enabled Smooth Streaming for the corresponding cache behavior.

Non-ASCII or unsafe characters in the path

If the path includes non-ASCII characters or unsafe characters as defined in RFC 1783 (<http://www.ietf.org/rfc/rfc1738.txt>), URL-encode those characters. Do not URL-encode any other characters in the path, or CloudFront will not invalidate the old version of the updated object.

Invalidation paths

The path is relative to the distribution. A leading / is optional. For example, to invalidate the object at `http://d111111abcdef8.cloudfront.net/images/image2.jpg`, you would specify the following:

/images/image2.jpg

or

images/image2.jpg

You can also invalidate multiple objects simultaneously by using the * wildcard. The *, which replaces 0 or more characters, must be the last character in the invalidation path. Also, if you use the AWS command line interface (CLI) for invalidating objects and you specify a path that includes the * wildcard, you must use quotes ("") around the path (like "/*").

The following are some examples:

- To invalidate all of the objects in a directory:

/directory-path/*

- To invalidate a directory, all of its subdirectories, and all of the objects in the directory and subdirectories:

/directory-path*

- To invalidate all files that have the same name but different file name extensions, such as logo.jpg, logo.png, and logo.gif:

/directory-path/file-name.*

- To invalidate all of the files in a directory for which the file name starts with the same characters (such as all of the files for a video in HLS format), regardless of the file name extension:

/directory-path/initial-characters-in-file-name*

- When you configure CloudFront to cache based on query string parameters and you want to invalidate every version of an object:

/directory-path/file-name.file-name-extension*

- To invalidate all of the objects in a distribution:

/*

The maximum length of a path is 4,000 characters.

For information about invalidating objects if you use a Lambda function to change the URI, see [Changing the URI Using a Lambda Function](#).

The charge to submit an invalidation path is the same regardless of the number of objects you're invalidating: a single object (/images/logo.jpg) or all of the objects that are associated with a distribution /*). For more information, see [Amazon CloudFront Pricing](#).

If the invalidation path is a directory and if you have not standardized on a method for specifying directories—with or without a trailing slash (/)—we recommend that you invalidate the directory both with and without a trailing slash, for example, /images and /images/.

Signed URLs

If you are using signed URLs, invalidate an object by including only the portion of the URL before the question mark (?).

Invalidating Objects and Displaying Information about Invalidations

You can use the CloudFront console or CloudFront API actions to create and run an invalidation, display a list of the invalidations that you submitted previously, and display detailed information about an individual invalidation. You can also copy an existing invalidation, edit the list of object paths, and run the edited invalidation.

See the applicable topic:

- [Invalidating Objects Using the CloudFront Console \(p. 169\)](#)
- [Copying, Editing, and Rerunning an Existing Invalidiation Using the CloudFront Console \(p. 170\)](#)
- [Canceling Invalidations \(p. 170\)](#)
- [Listing Invalidations Using the CloudFront Console \(p. 170\)](#)
- [Displaying Information about an Invalidiation Using the CloudFront Console \(p. 171\)](#)
- [Invalidating Objects and Displaying Information about Invalidations Using the CloudFront API \(p. 171\)](#)

Invalidating Objects Using the CloudFront Console

To invalidate objects using the CloudFront console, do the following procedure.

To invalidate objects using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the distribution for which you want to invalidate objects.
3. Choose **Distribution Settings**.
4. Choose the **Invalidations** tab.
5. Choose **Create Invalidation**.
6. For the objects that you want to invalidate, enter one invalidation path per line. For information about specifying invalidation paths, see [Specifying the Objects to Invalidate \(p. 166\)](#).

Important

Specify object paths carefully. You can't cancel an invalidation request after you start it.

7. Choose **Invalidate**.

Copying, Editing, and Rerunning an Existing Invalidations Using the CloudFront Console

You can copy an invalidation that you created previously, update the list of invalidation paths, and run the updated invalidation. You cannot copy an existing invalidation, update the invalidation paths, and save the updated invalidation without running it.

Important

If you copy an invalidation that is still in progress, update the list of invalidation paths, and run the updated invalidation, CloudFront will not stop or delete the invalidation that you copied. If any invalidation paths appear in the original and in the copy, CloudFront will try to invalidate the objects twice, and both invalidations will count against your maximum number of free invalidations for the month. If you've already reached the maximum number of free invalidations, you'll be charged for both invalidations of each object. For more information, see [Invalidation Limits \(p. 171\)](#).

To copy, edit, and rerun an existing invalidation using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the distribution that contains the invalidation that you want to copy.
3. Choose **Distribution Settings**.
4. Choose the **Invalidations** tab.
5. Choose the invalidation that you want to copy.

If you aren't sure which invalidation you want to copy, you can choose an invalidation and choose **Details** to display detailed information about that invalidation.

6. Choose **Copy**.
7. Update the list of invalidation paths if applicable.
8. Choose **Invalidate**.

Canceling Invalidations

When you submit an invalidation request to CloudFront, CloudFront forwards the request to all edge locations within a few seconds, and each edge location starts processing the invalidation immediately. As a result, you can't cancel an invalidation after you submit it.

Listing Invalidations Using the CloudFront Console

Using the console, you can display a list of the last 100 invalidations that you've created and run for a distribution. If you want to get a list of more than 100 invalidations, use the GET [Invalidation List API](#) action. For more information, see [GET Invalidations List](#) in the *Amazon CloudFront API Reference*.

To list invalidations using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the distribution for which you want to display a list of invalidations.
3. Choose **Distribution Settings**.

4. Choose the **Invalidations** tab.

Displaying Information about an Invalidations Using the CloudFront Console

You can display detailed information about an invalidation, including distribution ID, invalidation ID, the status of the invalidation, the date and time that the invalidation was created, and a complete list of the invalidation paths.

To display information about an invalidation using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the distribution that contains the invalidation that you want to display detailed information for.
3. Choose **Distribution Settings**.
4. Choose the **Invalidations** tab.
5. Choose the applicable invalidation.
6. Choose **Details**.

Invalidating Objects and Displaying Information about Invalidations Using the CloudFront API

For information about invalidating objects and about displaying information about invalidations using the CloudFront API, see the applicable topic in the *Amazon CloudFront API Reference*:

- Invalidating objects: [POST Invalidations](#)
- Getting a list of your invalidations: [GET Invalidations](#)
- Getting information about a specific invalidation: [GET Invalidations](#)

Third-Party Tools for Invalidating Objects

In addition to the invalidation methods provided by CloudFront, several third-party tools provide ways to invalidate objects. For a list of tools, see [Invalidate Objects \(p. 365\)](#).

Invalidation Limits

If you're invalidating objects individually, you can have invalidation requests for up to 3,000 objects per distribution in progress at one time. This can be one invalidation request for up to 3,000 objects, up to 3,000 requests for one object each, or any other combination that doesn't exceed 3,000 objects. For example, you can submit 30 invalidation requests that invalidate 100 objects each. As long as all 30 invalidation requests are still in progress, you can't submit any more invalidation requests. If you exceed the limit, CloudFront returns an error message.

If you're using the * wildcard, you can have requests for up to 15 invalidation paths in progress at one time. You can also have invalidation requests for up to 3,000 individual objects per distribution in progress at the same time; the limit on wildcard invalidation requests is independent of the limit on invalidating objects individually.

Paying for Object Invalidation

The first 1,000 invalidation paths that you submit per month are free; you pay for each invalidation path over 1,000 in a month. An invalidation path can be for a single object (such as /images/logo.jpg) or

for multiple objects (such as `/images/*`). A path that includes the `*` wildcard counts as one path even if it causes CloudFront to invalidate thousands of objects.

This limit of 1000 invalidation paths per month applies to the total number of invalidation paths across all of the distributions that you create with one AWS account. For example, if you use the AWS account `john@example.com` to create three distributions, and you submit 600 invalidation paths for each distribution in a given month (for a total of 1,800 invalidation paths), AWS will charge you for 800 invalidation paths in that month. For specific information about invalidation pricing, see [Amazon CloudFront Pricing](#). For more information about invalidation paths, see [Invalidation paths](#).

Customizing Error Responses

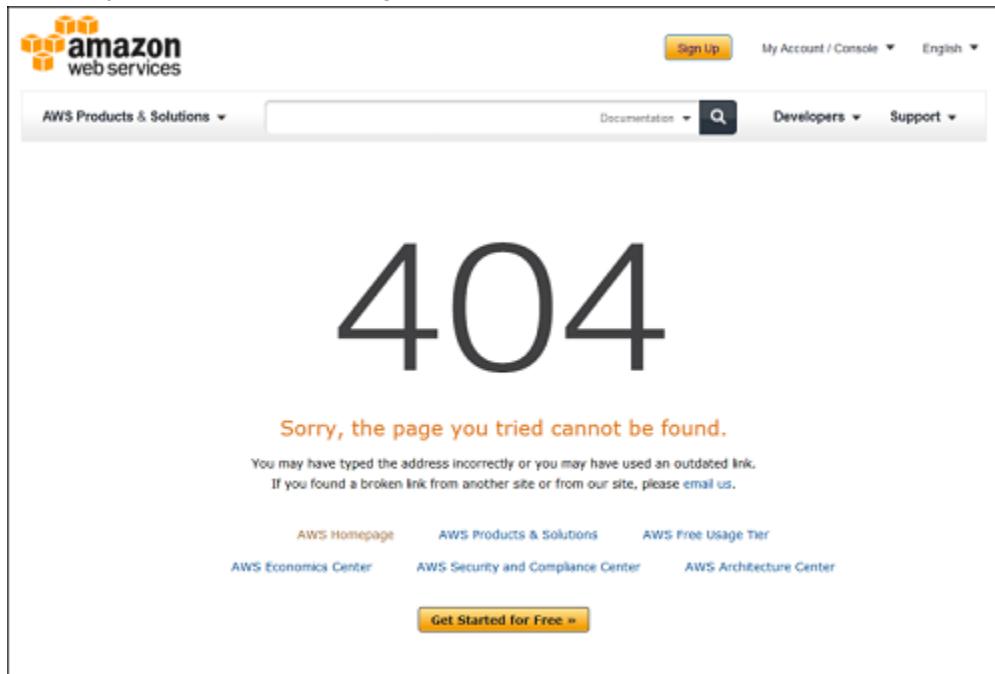
Topics

- [Creating or Updating a Cache Behavior for Custom Error Pages \(p. 173\)](#)
- [Changing Response Codes \(p. 173\)](#)
- [Controlling How Long CloudFront Caches Errors \(p. 174\)](#)
- [How CloudFront Responds When a Custom Error Page Is Unavailable \(p. 175\)](#)
- [Pricing for Custom Error Pages \(p. 175\)](#)
- [Configuring Error Response Behavior \(p. 175\)](#)

If the objects that you're serving through CloudFront are unavailable for some reason, your web server typically returns an HTTP status code to CloudFront. For example, if a viewer specifies an invalid URL, your web server returns a 404 status code to CloudFront, and CloudFront returns that status code to the viewer. The viewer displays a brief and sparsely formatted default message similar to this:

`Not Found: The requested URL /myfile.html was not found on this server.`

If you'd rather display a custom error message, possibly using the same formatting as the rest of your website, you can have CloudFront return to the viewer an object (for example, an HTML file) that contains your custom error message.



You can specify a different object for each supported HTTP status code, or you can use the same object for all of the supported status codes. You can also choose to specify objects for some status codes and not for others.

The objects that you're serving through CloudFront can be unavailable for a variety of reasons. These fall into two broad categories:

- **Client errors** indicate a problem with the request. For example, an object with the specified name isn't available, or the user doesn't have the permissions required to get an object in your Amazon S3 bucket. When a client error occurs, the origin returns an HTTP status code in the 400 range to CloudFront.
- **Server errors** indicate a problem with the origin server. For example, the HTTP server is busy or unavailable. When a server error occurs, either your origin server returns an HTTP status code in the 500 range to CloudFront, or CloudFront doesn't get a response from your origin server for a certain period of time and assumes a 504 status code (gateway timeout).

The HTTP status codes for which CloudFront can return a custom error page include the following:

- 400, 403, 404, 405, 414, 416
- 500, 501, 502, 503, 504

Note

You can create a custom error page for HTTP status code 416 (Requested Range Not Satisfiable), and you can change the HTTP status code that CloudFront returns to viewers when your origin returns a status code 416 to CloudFront. (For more information, see [Changing Response Codes \(p. 173\)](#).) However, CloudFront doesn't cache status code 416 responses, so you can specify a value for **Error Caching Minimum TTL** for status code 416, but CloudFront doesn't use it.

For a detailed explanation of how CloudFront handles error responses from your origin, see [How CloudFront Processes and Caches HTTP 4xx and 5xx Status Codes from Your Origin \(p. 203\)](#).

Creating or Updating a Cache Behavior for Custom Error Pages

If you want to store your objects and your custom error pages in different locations, your distribution must include a cache behavior for which the following is true:

- The value of **Path Pattern** matches the path to your custom error messages. For example, suppose you saved custom error pages for 4xx errors in an Amazon S3 bucket in a directory named `/4xx-errors`. Your distribution must include a cache behavior for which the path pattern routes requests for your custom error pages to that location, for example, `/4xx-errors/*`.
- The value of **Origin** specifies the value of **Origin ID** for the origin that contains your custom error pages.

For more information, see [Cache Behavior Settings \(p. 70\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

Changing Response Codes

You can choose the HTTP status code CloudFront returns along with a custom error page for a given HTTP status code. For example, if your origin returns a 500 status code to CloudFront, you might want CloudFront to return a custom error page and a 200 status code (OK) to the viewer. There are a variety of reasons that you might want CloudFront to return a status code to the viewer that is different from the one that your origin returned to CloudFront:

- Some internet devices (some firewalls and corporate proxies, for example) intercept HTTP 4xx and 5xx and prevent the response from being returned to the viewer. If you substitute 200, the response typically won't be intercepted.
- If you don't care about distinguishing among different client errors or server errors, you can specify **400** or **500** as the value that CloudFront returns for all 4xx or 5xx status codes.
- You might want to return a 200 status code (OK) and static website so your customers don't know that your website is down.

If you enable CloudFront access logs and you configure CloudFront to change the HTTP status code in the response, the value of the `sc-status` column in access logs will contain the status code that you specify. However, the value of the `x-edge-result-type` column will not be affected; it will still contain the result type of the response from the origin. For example, suppose you configure CloudFront to return a status code of 200 to the viewer when the origin returns 404 (Not Found) to CloudFront. When the origin responds to a request with a 404 status code, the value in the `sc-status` column in the access log will be 200, but the value in the `x-edge-result-type` column will be `Error`.

You can configure CloudFront to return any of the following HTTP status codes along with a custom error page:

- 200
- 400, 403, 404, 405, 414, 416
- 500, 501, 502, 503, 504

Controlling How Long CloudFront Caches Errors

By default, when your origin returns an HTTP 4xx or 5xx status code, CloudFront caches these error responses for five minutes and then submits the next request for the object to your origin to see whether the problem that caused the error has been resolved and the requested object is now available.

Note

You can create a custom error page for HTTP status code 416 (Requested Range Not Satisfiable), and you can change the HTTP status code that CloudFront returns to viewers when your origin returns a status code 416 to CloudFront. (For more information, see [Changing Response Codes \(p. 173\)](#).) However, CloudFront doesn't cache status code 416 responses, so you can specify a value for **Error Caching Minimum TTL** for status code 416, but CloudFront doesn't use it.

You can specify the error-caching duration—the **Error Caching Minimum TTL**—for each 4xx and 5xx status code that CloudFront caches. For a procedure, see [Configuring Error Response Behavior \(p. 175\)](#). When you specify a duration, note the following:

- If you specify a short error-caching duration, CloudFront forwards more requests to your origin than if you specify a longer duration. For 5xx errors, this may aggravate the problem that originally caused your origin to return an error.
- When your origin returns an error for an object, CloudFront responds to requests for the object either with the error response or with your custom error page until the error-caching duration elapses. If you specify a long error-caching duration, CloudFront might continue to respond to requests with an error response or your custom error page for a long time after the object becomes available again.

If you want to control how long CloudFront caches errors for individual objects, you can configure your origin server to add the applicable header to the error response for that object:

- **If the origin adds a `Cache-Control max-age` or `Cache-Control s-maxage` directive, or an `Expires` header:** CloudFront caches error responses for the greater of the value in the header or the value of **Error Caching Minimum TTL**.

- **If the origin adds other Cache-Control directives or adds no headers:** CloudFront caches error responses for the value of **Error Caching Minimum TTL**.

If the expiration time for a 4xx or 5xx status code for an object is longer than you want to wait, you can invalidate the status code by using the URL of the requested object. If your origin is returning an error response for multiple objects, you need to invalidate each object separately. For more information about invalidating objects, see [Invalidating Objects \(Web Distributions Only\) \(p. 165\)](#).

How CloudFront Responds When a Custom Error Page Is Unavailable

If you configure CloudFront to return a custom error page for an HTTP status code but the custom error page isn't available, CloudFront returns to the viewer the status code that CloudFront received from the origin that contains the custom error pages. For example, suppose your custom origin returns a 500 status code and you have configured CloudFront to get a custom error page for a 500 status code from an Amazon S3 bucket. However, someone accidentally deleted the custom error page from your bucket. CloudFront will return an HTTP 404 status code (not found) to the viewer that requested the object.

Pricing for Custom Error Pages

When CloudFront returns a custom error page to a viewer, you pay the standard CloudFront charges for the custom error page, not the charges for the requested object. For more information about CloudFront charges, see [Amazon CloudFront Pricing](#).

Configuring Error Response Behavior

You can use either the CloudFront API or console to configure CloudFront error responses. For information about using the CloudFront API to configure error responses, go to [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*, and see the `CustomErrorResponse` element.

To configure CloudFront error responses using the console

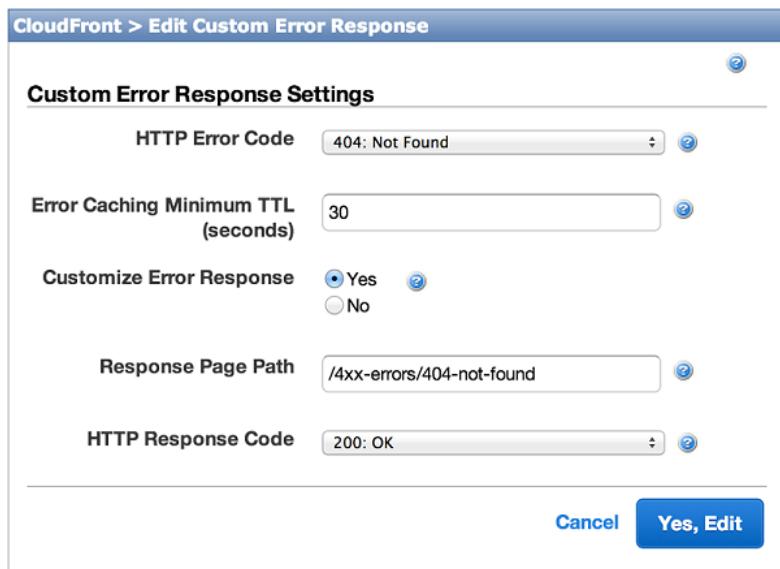
1. Create the custom error pages that you want CloudFront to return to viewers when your origin returns HTTP 4xx or 5xx errors. Save the pages in a location that is accessible to CloudFront.

We recommend that you store custom error pages in an Amazon S3 bucket even if you're using a custom origin. If you store custom error pages on an HTTP server and the server starts to return 5xx errors, CloudFront can't get the files that you want to return to viewers because the origin server is unavailable.

2. Confirm that you have granted CloudFront at least `read` permission to your custom error page objects.

For more information about Amazon S3 permissions, see [Access Control](#) in the *Amazon Simple Storage Service Developer Guide*. For information on using the Amazon S3 console to update permissions, go to the [Amazon Simple Storage Service Console User Guide](#).

3. (Optional) Configure your origin server to add Cache-Control directives or an `Expires` header along with the error response for specific objects, if applicable. For more information, see [Controlling How Long CloudFront Caches Errors \(p. 174\)](#).
4. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
5. In the list of distributions, select the distribution to update and choose **Distribution Settings**.
6. Choose the **Error Pages** tab. Then either choose **Create Custom Error Response**, or choose an existing error code and choose **Edit**.



7. Enter the applicable values. For more information, see [Custom Error Pages and Error Caching \(p. 82\)](#).
8. If you configured CloudFront to return custom error pages, add or update the applicable cache behaviors. For more information, see [Creating or Updating a Cache Behavior for Custom Error Pages \(p. 173\)](#).
9. To save your changes, choose **Yes, Edit**.

How CloudFront Processes Partial Requests for an Object (Range GETs)

For a large object, an end user's browser or client might make multiple GET requests and use the Range request header to download the object in smaller units. These requests for ranges of bytes, sometimes known as Range GET requests, improve the efficiency of partial downloads and the recovery from partially failed transfers.

When CloudFront receives a Range GET request, it checks the cache in the edge location that received the request. If the cache in that edge location already contains the entire object or the requested portion of the object, CloudFront immediately serves the requested range from the cache.

If the cache doesn't contain the requested range, CloudFront forwards the request to the origin. (To optimize performance, CloudFront may request a larger range than the client requested in the Range GET.) What happens next depends on whether the origin supports Range GET requests:

- **If the origin supports Range GET requests:** It returns the requested range. CloudFront serves the requested range and also caches it for future requests. (Amazon S3 supports Range GET requests, as do some HTTP servers, for example, Apache and IIS. For information about whether your HTTP server does, see the documentation for your HTTP server.)
- **If the origin doesn't support Range GET requests:** It returns the entire object. CloudFront serves the entire object and also caches the entire object for future requests. After CloudFront caches the entire object in an edge cache, it responds to Range GET requests by serving the requested range.

In either case, CloudFront begins to serve the requested range or object to the end user as soon as the first byte arrives from the origin.

Note

If the viewer makes a Range GET request and the origin returns Transfer-Encoding: chunked, CloudFront returns the entire object to the viewer instead of the requested range.

CloudFront generally follows the RFC specification for the Range header. However, if your Range headers don't adhere to the following requirements, CloudFront will return HTTP status code 200 with the full object instead of status code 206 with the specified ranges:

- The ranges must be listed in ascending order. For example, 100–200, 300–400 is valid, 300–400, 100–200 is not valid.
- The ranges must not overlap. For example, 100–200, 150–250 is not valid.
- All of the ranges specifications must be valid. For example, you can't specify a negative value as part of a range.

For more information about the Range request header, see "Section 14.35 Range" in *Hypertext Transfer Protocol -- HTTP/1.1* at <http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.35>.

Specifying a Default Root Object (Web Distributions Only)

You can configure CloudFront to return a specific object (the default root object) when an end user requests the root URL for your distribution instead of an object in your distribution. Specifying a default root object avoids exposing the contents of your distribution.

For example, the following request points to the object `image.jpg`:

```
http://d111111abcdef8.cloudfront.net/image.jpg
```

The following request points to the root URL of the same distribution instead of to a specific object:

```
http://d111111abcdef8.cloudfront.net/
```

When you define a default root object, an end-user request that calls the root of your distribution returns the default root object. For example, if you designate the file `index.html` as your default root object, a request for:

```
http://d111111abcdef8.cloudfront.net/
```

returns:

```
http://d111111abcdef8.cloudfront.net/index.html
```

However, if you define a default root object, an end-user request for a subdirectory of your distribution does not return the default root object. For example, suppose `index.html` is your default root object and that CloudFront receives an end-user request for the `install` directory under your CloudFront distribution:

```
http://d111111abcdef8.cloudfront.net/install/
```

CloudFront will not return the default root object even if a copy of `index.html` appears in the `install` directory.

If you configure your distribution to allow all of the HTTP methods that CloudFront supports, the default root object applies to all methods. For example, if your default root object is `index.php` and you write your application to submit a `POST` request to the root of your domain (`http://example.com`), CloudFront will send the request to `http://example.com/index.php`.

The behavior of CloudFront default root objects is different from the behavior of Amazon S3 index documents. When you configure an Amazon S3 bucket as a website and specify the index document, Amazon S3 returns the index document even if a user requests a subdirectory in the bucket. (A copy of the index document must appear in every subdirectory.) For more information about configuring Amazon S3 buckets as websites and about index documents, see the [Hosting Websites on Amazon S3](#) chapter in the *Amazon Simple Storage Service Developer Guide*.

Important

Remember that a default root object applies only to your CloudFront distribution. You still need to manage security for your origin. For example, if you are using an Amazon S3 origin, you still need to set your Amazon S3 bucket ACLs appropriately to ensure the level of access you want on your bucket.

If you don't define a default root object, requests for the root of your distribution pass to your origin server. If you are using an Amazon S3 origin, any of the following might be returned:

- **A list of the contents of your Amazon S3 bucket**—Under any of the following conditions, the contents of your origin are visible to anyone who uses CloudFront to access your distribution:
 - Your bucket is not properly configured.
 - The Amazon S3 permissions on the bucket associated with your distribution and on the objects in the bucket grant access to *everyone*.
 - An end user accesses your origin using your origin root URL.
- **A list of the private contents of your origin**—If you configure your origin as a private distribution (only you and CloudFront have access), the contents of the Amazon S3 bucket associated with your distribution are visible to anyone who has the credentials to access your distribution through CloudFront. In this case, users are not able to access your content through your origin root URL. For more information about distributing private content, see [Serving Private Content through CloudFront \(p. 207\)](#).
- **Error 403 Forbidden**—CloudFront returns this error if the permissions on the Amazon S3 bucket associated with your distribution or the permissions on the objects in that bucket deny access to CloudFront and to everyone.

To avoid exposing the contents of your web distribution or returning an error, perform the following procedure to specify a default root object for your distribution.

To specify a default root object for your distribution

1. Upload the default root object to the origin that your distribution points to.

The file can be any type supported by CloudFront. For a list of constraints on the file name, see the description of the `DefaultRootObject` element in [DistributionConfig Complex Type](#).

Note

If the file name of the default root object is too long or contains an invalid character, CloudFront returns the error `HTTP 400 Bad Request - InvalidDefaultRootObject`. In addition, CloudFront caches the code for five minutes and writes the results to the access logs.

2. Confirm that the permissions for the object grant CloudFront at least `read` access.

For more information about Amazon S3 permissions, see [Access Control](#) in the *Amazon Simple Storage Service Developer Guide*. For information on using the Amazon S3 console to update permissions, go to the [Amazon Simple Storage Service Console User Guide](#).

3. Update your distribution to refer to the default root object using the CloudFront console or the CloudFront API.

To specify a default root object using the CloudFront console:

- a. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
- b. In the list of distributions in the top pane, select the distribution to update.
- c. In the **Distribution Details** pane, on the **General** tab, choose **Edit**.
- d. In the **Edit Distribution** dialog box, in the **Default Root Object** field, enter the file name of the default root object.

Enter only the object name, for example, `index.html`. Do not add a / before the object name.
- e. To save your changes, choose **Yes, Edit**.

To update your configuration using the CloudFront API, you specify a value for the `DefaultRootObject` element in your distribution. For information about using the CloudFront API to specify a default root object, see [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

4. Confirm that you have enabled the default root object by requesting your root URL. If your browser doesn't display the default root object, perform the following steps:
 - a. Confirm that your distribution is fully deployed by viewing the status of your distribution in the CloudFront console.
 - b. Repeat Steps 2 and 3 to verify that you granted the correct permissions and that you correctly updated the configuration of your distribution to specify the default root object.

Serving Compressed Files

You can configure CloudFront to automatically compress files of certain types and serve the compressed files when viewer requests include `Accept-Encoding: gzip` in the request header. When content is compressed, downloads are faster because the files are smaller—in some cases, less than a quarter the size of the original. Especially for JavaScript and CSS files, faster downloads translates into faster rendering of web pages for your users. In addition, because the cost of CloudFront data transfer is based on the total amount of data served, serving compressed files is less expensive than serving uncompressed files.

Important

A viewer request must include `Accept-Encoding: gzip` in the request header, or CloudFront won't compress the requested file.

If you're using a custom origin, you can configure your origin to compress files with or without CloudFront compression. Your origin can compress file types that CloudFront doesn't compress. (See [File Types that CloudFront Compresses \(p. 181\)](#).) If your origin returns a compressed file to CloudFront, CloudFront detects that the file is compressed based on the value of the `Content-Encoding` header and doesn't compress the file again.

Topics

- [Using CloudFront to Compress Your Content \(p. 179\)](#)
- [Using a Custom Origin to Compress Your Content \(p. 182\)](#)

Using CloudFront to Compress Your Content

CloudFront can compress files both for Amazon S3 origins and for custom origins. When you configure CloudFront to compress your content, you specify the setting in your cache behaviors.

When you configure CloudFront to compress your content, here's how CloudFront serves your content:

1. You create or update a CloudFront distribution and configure CloudFront to compress content.
2. A viewer requests a file. The viewer adds the `Accept-Encoding: gzip` header to the request. This indicates that the viewer supports compressed content.
3. At the edge location, CloudFront checks the cache for a compressed version of the file that is referenced in the request.
4. If the compressed file is already in the cache, CloudFront returns the file to the viewer and skips the remaining steps.
5. If the compressed file is not in the cache, CloudFront forwards the request to the origin server, which can be either an Amazon S3 bucket or a custom origin.

Note

If CloudFront has an uncompressed version of the file in the cache, it still forwards a request to the origin.

6. The origin server returns an uncompressed version of the requested file to CloudFront.
7. CloudFront determines whether the file is compressible:
 - The file must be of a type that CloudFront compresses.
 - The file size must be between 1,000 and 10,000,000 bytes.
 - The response must include a `Content-Length` header so CloudFront can determine whether the size of the file is in the range that CloudFront compresses. If the `Content-Length` header is missing, CloudFront won't compress the file.
 - The response must not include a `Content-Encoding` header.
8. If the file is compressible, CloudFront compresses it, returns the compressed file to the viewer, and adds it to the cache.
9. The viewer uncompresses the file.

Note the following:

File types that CloudFront compresses

CloudFront compresses files in a large number of file types. For a complete list, see [File Types that CloudFront Compresses \(p. 181\)](#).

Size of files that CloudFront compresses

CloudFront compresses files that are between 1,000 bytes and 10,000,000 bytes in size.

Content-Length header

The origin must include a `Content-Length` header in the response so CloudFront can determine whether the size of the file is in the range that CloudFront compresses. If the `Content-Length` header is missing, CloudFront won't compress the file.

Etag header

If you configure CloudFront to compress content, CloudFront removes the `ETag` response header from the files that it compresses. When the `ETag` header is present, CloudFront and your origin can use it to determine whether the version of a file in a CloudFront edge cache is identical to the version on the origin server. However, after compression the two versions are no longer identical. As a result, when a compressed file expires and CloudFront forwards another request to your origin, your origin always returns the file to CloudFront instead of an HTTP status code 304 (Not Modified).

Content already in edge locations when you configure CloudFront to compress files

CloudFront compresses files in each edge location when it gets the files from your origin. When you configure CloudFront to compress your content, it doesn't compress files that are already in edge locations. In addition, when a file expires in an edge location and CloudFront forwards another request for the file to your origin, CloudFront doesn't compress the file if your origin returns an HTTP status code 304, which means that the edge location already has the latest version of

the file. If you want CloudFront to compress the files that are already in edge locations, you'll need to invalidate those files. For more information, see [Invalidate Objects \(Web Distributions Only\) \(p. 165\)](#).

Custom origin is already configured to compress files

If you configure CloudFront to compress files and CloudFront is forwarding requests to a custom origin that is also configured to compress files, the custom origin will include a `Content-Encoding: gzip` header, which indicates that the file that the origin returned to CloudFront has already been compressed. CloudFront returns the cached file to the viewer and caches it in the edge location.

Note

CloudFront does not compress a file if the response includes a `Content-Encoding` header, regardless of the value.

Request doesn't include Accept-Encoding: gzip

If the `Accept-Encoding` header is missing from the request, CloudFront serves uncompressed content. If the `Accept-Encoding` header includes additional values such as `deflate` or `sdch`, CloudFront removes them before forwarding the request to the origin server.

CloudFront is busy

In rare cases, when a CloudFront edge location is unusually busy, some files might not be compressed.

Configuring a CloudFront Distribution to Compress Content

To configure a web distribution to compress your content, you update the applicable cache behaviors by using one of the following methods:

- **CloudFront console** – Update the **Compress objects automatically** setting. For more information, see [Creating or Updating a Web Distribution Using the CloudFront Console \(p. 60\)](#).
- **CloudFront API** – Change the value of the `Compress` element to `true`. For more information, see [POST Distribution](#) (to create a new distribution) or [PUT Distribution Config](#) (to update an existing distribution).
- **One of the AWS SDKs** – See the applicable SDK documentation on the [AWS Documentation](#) page.
- **The AWS CLI** – For more information, see `create-distribution` or `update-distribution` in the [AWS Command Line Interface Reference](#).

File Types that CloudFront Compresses

If you configure CloudFront to compress your content, CloudFront compresses files that have the following values in the `Content-Type` header:

| | |
|------------------------|--------------------|
| application/eot | application/x-otf |
| application/font | application/x-perl |
| application/font-sfnt | application/x-ttf |
| application/javascript | font/eot |
| application/json | font/ttf |
| application/opentype | font/otf |
| application/otf | font/opentype |

| | |
|-------------------------------|---------------------------|
| application/pkcs7-mime | image/svg+xml |
| application/truetype | text/css |
| application/ttf | text/csv |
| application/vnd.ms-fontobject | text/html |
| application/xhtml+xml | text/javascript |
| application/xml | text/js |
| application/xml+rss | text/plain |
| application/x-font-opentype | text/richtext |
| application/x-font-truetype | text/tab-separated-values |
| application/x-font-ttf | text/xml |
| application/x-httplib-cgi | text/x-script |
| application/x-javascript | text/x-component |
| application/x-mpegurl | text/x-java-source |
| application/x-opentype | |

Using a Custom Origin to Compress Your Content

If you want to compress file types that CloudFront doesn't compress, you can configure your custom origin to compress files of those types using gzip. CloudFront doesn't support other compression algorithms. When your origin returns the compressed file to CloudFront, it will include a `Content-Encoding: gzip` header, which indicates to CloudFront that the file is already compressed.

Note

CloudFront does not compress a file if the response includes a `Content-Encoding` header, regardless of the value.

Serving Compressed Files When Your Origin Server Is Running IIS

By default, IIS does not serve compressed content for requests that come through proxy servers such as CloudFront. If you're using IIS and if you configured IIS to compress content by using the `httpCompressionElement`, change the value of the `noCompressionForProxies` attribute to `false` so IIS will return compressed content to CloudFront.

In addition, if you have compressed objects that are requested less frequently than every few seconds, you might have to change the values of `frequentHitThreshold` and `frequentHitTimePeriod`.

For more information, refer to the IIS documentation on the Microsoft website.

Serving Compressed Files When Your Origin Server Is Running NGINX

When CloudFront forwards a request to the origin server, it includes a `Via` header. This causes NGINX to interpret the request as proxied and, by default, NGINX disables compression for proxied requests. If your

version of NGINX includes the `gzip_proxied` setting, change the value to `any` so that NGINX will return compressed content to CloudFront. For more information, see the NGINX documentation for the module `ngx_http_gzip_module`.

Request and Response Behavior

The following sections explain how CloudFront processes viewer requests and forwards the requests to your Amazon S3 or custom origin, and how CloudFront processes responses from your origin, including how CloudFront processes and caches 4xx and 5xx HTTP status codes.

Topics

- [Request and Response Behavior for Amazon S3 Origins \(p. 184\)](#)
- [Request and Response Behavior for Custom Origins \(p. 190\)](#)
- [How CloudFront Processes HTTP 3xx Status Codes from Your Origin \(p. 202\)](#)
- [How CloudFront Processes and Caches HTTP 4xx and 5xx Status Codes from Your Origin \(p. 203\)](#)

Request and Response Behavior for Amazon S3 Origins

Topics

- [How CloudFront Processes and Forwards Requests to Your Amazon S3 Origin Server \(p. 184\)](#)
- [How CloudFront Processes Responses from Your Amazon S3 Origin Server \(p. 189\)](#)

How CloudFront Processes and Forwards Requests to Your Amazon S3 Origin Server

For information about how CloudFront processes viewer requests and forwards the requests to your Amazon S3 origin, see the applicable topic:

Topics

- [Caching Duration and Minimum TTL \(p. 185\)](#)
- [Client IP Addresses \(p. 185\)](#)
- [Conditional GETs \(p. 185\)](#)
- [Cookies \(p. 185\)](#)
- [Cross-Origin Resource Sharing \(CORS\) \(p. 186\)](#)
- [GET Requests That Include a Body \(p. 186\)](#)
- [HTTP Methods \(p. 186\)](#)
- [HTTP Request Headers That CloudFront Removes or Updates \(p. 186\)](#)
- [Maximum Length of a Request and Maximum Length of a URL \(p. 187\)](#)
- [OCSP Stapling \(p. 187\)](#)
- [Protocols \(p. 188\)](#)
- [Query Strings \(p. 188\)](#)
- [Origin Response Timeout \(p. 188\)](#)
- [Simultaneous Requests for the Same Object \(Traffic Spikes\) \(p. 188\)](#)

Caching Duration and Minimum TTL

For web distributions, to control how long your objects stay in a CloudFront cache before CloudFront forwards another request to your origin, you can:

- Configure your origin to add a `Cache-Control` or an `Expires` header field to each object.
- Specify a value for Minimum TTL in CloudFront cache behaviors.
- Use the default value of 24 hours.

For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

Client IP Addresses

If a viewer sends a request to CloudFront and does not include an `X-Forwarded-For` request header, CloudFront gets the IP address of the viewer from the TCP connection, adds an `X-Forwarded-For` header that includes the IP address, and forwards the request to the origin. For example, if CloudFront gets the IP address 192.0.2.2 from the TCP connection, it forwards the following header to the origin:

`X-Forwarded-For: 192.0.2.2`

If a viewer sends a request to CloudFront and includes an `X-Forwarded-For` request header, CloudFront gets the IP address of the viewer from the TCP connection, appends it to the end of the `x-Forwarded-For` header, and forwards the request to the origin. For example, if the viewer request includes `X-Forwarded-For: 192.0.2.4,192.0.2.3` and CloudFront gets the IP address 192.0.2.2 from the TCP connection, it forwards the following header to the origin:

`X-Forwarded-For: 192.0.2.4,192.0.2.3,192.0.2.2`

Note

The `X-Forwarded-For` header contains IPv4 addresses (such as 192.0.2.44) and IPv6 addresses (such as 2001:0db8:85a3:0000:0000:8a2e:0370:7334), as applicable.

Conditional GETs

When CloudFront receives a request for an object that has expired from an edge cache, it forwards the request to the Amazon S3 origin either to get the latest version of the object or to get confirmation from Amazon S3 that the CloudFront edge cache already has the latest version. When Amazon S3 originally sent the object to CloudFront, it included an `ETag` value and a `LastModified` value in the response. In the new request that CloudFront forwards to Amazon S3, CloudFront adds one or both of the following:

- An `If-Match` or `If-None-Match` header that contains the `ETag` value for the expired version of the object.
- An `If-Modified-Since` header that contains the `LastModified` value for the expired version of the object.

Amazon S3 uses this information to determine whether the object has been updated and, therefore, whether to return the entire object to CloudFront or to return only an HTTP 304 status code (not modified).

Cookies

Amazon S3 doesn't process cookies. If you configure a cache behavior to forward cookies to an Amazon S3 origin, CloudFront forwards the cookies, but Amazon S3 ignores them. All future requests for the same object, regardless if you vary the cookie, are served from the existing object in the cache.

Cross-Origin Resource Sharing (CORS)

If you want CloudFront to respect Amazon S3 cross-origin resource sharing settings, configure CloudFront to forward selected headers to Amazon S3. For more information, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

GET Requests That Include a Body

If a viewer GET request includes a body, CloudFront returns an HTTP status code 403 (Forbidden) to the viewer.

HTTP Methods

If you configure CloudFront to process all of the HTTP methods that it supports, CloudFront accepts the following requests from viewers and forwards them to your Amazon S3 origin:

- `DELETE`
- `GET`
- `HEAD`
- `OPTIONS`
- `PATCH`
- `POST`
- `PUT`

CloudFront always caches responses to `GET` and `HEAD` requests. You can also configure CloudFront to cache responses to `OPTIONS` requests. CloudFront does not cache responses to requests that use the other methods.

If you use an Amazon S3 bucket as the origin for your distribution and if you use CloudFront origin access identities, `POST` requests aren't supported in some Amazon S3 regions and `PUT` requests in those regions require an additional header. For more information, see [Using an Origin Access Identity in Amazon S3 Regions that Support Only Signature Version 4 Authentication \(p. 215\)](#).

If you want to use multi-part uploads to add objects to an Amazon S3 bucket, you must add a CloudFront origin access identity to your distribution and grant the origin access identity the applicable permissions. For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

Important

If you configure CloudFront to accept and forward to Amazon S3 all of the HTTP methods that CloudFront supports, you must create a CloudFront origin access identity to restrict access to your Amazon S3 content and grant the origin access identity the applicable permissions. For example, if you configure CloudFront to accept and forward these methods because you want to use `PUT`, you must configure Amazon S3 bucket policies or ACLs to handle `DELETE` requests appropriately so viewers can't delete resources that you don't want them to. For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

For information about the operations supported by Amazon S3, see the [Amazon S3 documentation](#).

HTTP Request Headers That CloudFront Removes or Updates

CloudFront removes or updates the following header fields before forwarding requests to your Amazon S3 origin:

- `Accept`
- `Accept-Charset`
- `Accept-Encoding` – If the value contains `gzip`, CloudFront forwards `Accept-Encoding: gzip` to your Amazon S3 origin. If the value does not contain `gzip`, CloudFront removes the `Accept-Encoding` header field before forwarding the request to your origin.
- `Accept-Language`
- `Authorization`:
 - `GET` and `HEAD` requests: CloudFront removes the `Authorization` header field before forwarding the request to your origin.
 - `OPTIONS` requests: CloudFront removes the `Authorization` header field before forwarding the request to your origin if you configure CloudFront to cache responses to `OPTIONS` requests.
- CloudFront forwards the `Authorization` header field to your origin if you do not configure CloudFront to cache responses to `OPTIONS` requests.
- `DELETE`, `PATCH`, `POST`, and `PUT` requests: CloudFront does not remove the header field before forwarding the request to your origin.
- `Connection` – CloudFront replaces this header with `Connection: Keep-Alive` before forwarding the request to your Amazon S3 origin.
- `Cookie` – If you configure CloudFront to forward cookies, it will forward the `Cookie` header field to your Amazon S3 origin. If you don't, CloudFront removes the `Cookie` header field. For more information, see [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).
- `Expect`
- `Host` – CloudFront sets the value to the name of the Amazon S3 bucket that is associated with the requested object.
- `Proxy-Authorization`
- `Referer`
- `TE`
- `Upgrade`
- `User-Agent` – CloudFront replaces the value of this header field with `Amazon CloudFront`.

Maximum Length of a Request and Maximum Length of a URL

The maximum length of a request, including the path, the query string (if any), and headers, is 20,480 bytes.

CloudFront constructs a URL from the request. The maximum length of this URL is 8192 bytes.

If a request or a URL exceeds these limits, CloudFront returns HTTP status code 413, Request Header Fields Too Large, to the viewer, and then terminates the TCP connection to the viewer.

OCSP Stapling

When a viewer submits an HTTPS request for an object, either CloudFront or the viewer must confirm with the certificate authority (CA) that the SSL certificate for the domain has not been revoked. OCSP stapling speeds up certificate validation by allowing CloudFront to validate the certificate and to cache the response from the CA, so the client doesn't need to validate the certificate directly with the CA.

The performance improvement of OCSP stapling is more pronounced when CloudFront receives a lot of HTTPS requests for objects in the same domain. Each server in a CloudFront edge location must submit a separate validation request. When CloudFront receives a lot of HTTPS requests for the same domain, every server in the edge location soon has a response from the CA that it can "staple" to a

packet in the SSL handshake; when the viewer is satisfied that the certificate is valid, CloudFront can serve the requested object. If your distribution doesn't get much traffic in a CloudFront edge location, new requests are more likely to be directed to a server that hasn't validated the certificate with the CA yet. In that case, the viewer separately performs the validation step and the CloudFront server serves the object. That CloudFront server also submits a validation request to the CA, so the next time it receives a request that includes the same domain name, it has a validation response from the CA.

Protocols

CloudFront forwards HTTP or HTTPS requests to the origin server based on the protocol of the viewer request, either HTTP or HTTPS.

Important

If your Amazon S3 bucket is configured as a website endpoint, you cannot configure CloudFront to use HTTPS to communicate with your origin because Amazon S3 doesn't support HTTPS connections in that configuration.

Query Strings

For web distributions, you can configure whether CloudFront forwards query string parameters to your Amazon S3 origin. For RTMP distributions, CloudFront does not forward query string parameters. For more information, see [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#).

Origin Response Timeout

The origin response timeout, also known as the origin read timeout or origin request timeout, applies to both of the following values:

- The amount of time, in seconds, that CloudFront waits for a response after forwarding a request to Amazon S3
- The amount of time, in seconds, that CloudFront waits after receiving a packet of a response from S3 and before receiving the next packet

CloudFront behavior depends on the HTTP method:

- **GET and HEAD requests** – If Amazon S3 doesn't respond within 30 seconds or stops responding for 30 seconds, CloudFront drops the connection and makes two additional attempts to contact the origin. If the origin doesn't reply during the third attempt, CloudFront doesn't try again until it receives another request for content on the same CloudFront origin.
- **DELETE, OPTIONS, PATCH, PUT, and POST requests** – If Amazon S3 doesn't respond within 30 seconds, CloudFront drops the connection and doesn't try again to contact the origin. The client can resubmit the request if necessary.

For all requests, CloudFront attempts to establish a connection with S3. If the connection fails within 10 seconds, CloudFront drops the connection and makes two additional attempts to contact S3. If the origin doesn't reply during the third attempt, CloudFront doesn't try again until it receives another request for content on the same origin.

The response timeout for S3 can't be changed.

Simultaneous Requests for the Same Object (Traffic Spikes)

When a CloudFront edge location receives a request for an object and either the object isn't currently in the cache or the object has expired, CloudFront immediately sends the request to your Amazon S3 origin. If there's a traffic spike—if additional requests for the same object arrive at the edge location

before Amazon S3 responds to the first request—CloudFront pauses briefly before forwarding additional requests for the object to your origin. Typically, the response to the first request will arrive at the CloudFront edge location before the response to subsequent requests. This brief pause helps to reduce unnecessary load on Amazon S3. If additional requests are not identical because, for example, you configured CloudFront to cache based on request headers or query strings, CloudFront forwards all of the unique requests to your origin.

When the response from the origin includes a `Cache-Control: no-cache` header, CloudFront typically forwards the next request for the same object to the origin to determine whether the object has been updated. However, when there's a traffic spike and CloudFront pauses after forwarding the first request to your origin, multiple viewer requests might arrive before CloudFront receives a response from the origin. When CloudFront receives a response that contains a `Cache-Control: no-cache` header, it sends the object in the response to the viewer that made the original request and to all of the viewers that requested the object during the pause. After the response arrives from the origin, CloudFront forwards the next viewer request for the same object to the origin. In CloudFront access logs, the first request is identified as a `Miss` in the `x-edge-result-type` column, and all subsequent requests that CloudFront received during the pause are identified as a `Hit`. For more information about access log file format, see [Web Distribution Log File Format \(p. 319\)](#).

How CloudFront Processes Responses from Your Amazon S3 Origin Server

Topics

- [Canceled Requests \(p. 189\)](#)
- [HTTP Response Headers That CloudFront Removes or Updates \(p. 189\)](#)
- [Maximum File Size \(p. 190\)](#)
- [Redirects \(p. 190\)](#)

Canceled Requests

If an object is not in the edge cache, and if a viewer terminates a session (for example, closes a browser) after CloudFront gets the object from your origin but before it can deliver the requested object, CloudFront does not cache the object in the edge location.

HTTP Response Headers That CloudFront Removes or Updates

CloudFront removes or updates the following header fields before forwarding the response from your Amazon S3 origin to the viewer:

- **Set-Cookie** – If you configure CloudFront to forward cookies, it will forward the `Set-Cookie` header field to clients. For more information, see [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).
- **Trailer**
- **Transfer-Encoding** – If your Amazon S3 origin returns this header field, CloudFront sets the value to `chunked` before returning the response to the viewer.
- **Upgrade**
- **Via** – CloudFront sets the value to:

`Via: 1.1 alphanumeric-string.cloudfront.net (CloudFront)`

before returning the response to the viewer. For example:

`Via: 1.1 1026589cc7887e7a0dc7827b4example.cloudfront.net (CloudFront)`

Maximum File Size

The maximum size of a response body that CloudFront will return to the viewer is 20 GB. This includes chunked transfer responses that don't specify the Content-Length header value.

Redirects

You can configure an Amazon S3 bucket to redirect all requests to another host name; this can be another Amazon S3 bucket or an HTTP server. If you configure a bucket to redirect all requests and if the bucket is the origin for a CloudFront distribution, we recommend that you configure the bucket to redirect all requests to a CloudFront distribution using either the domain name for the distribution (for example, d111111abcdef8.cloudfront.net) or an alternate domain name (a CNAME) that is associated with a distribution (for example, example.com). Otherwise, viewer requests bypass CloudFront, and the objects are served directly from the new origin.

Note

If you redirect requests to an alternate domain name, you must also update the DNS service for your domain by adding a CNAME record. For more information, see [Using Alternate Domain Names \(CNAMEs\) \(p. 50\)](#).

Here's what happens when you configure a bucket to redirect all requests:

1. A viewer (for example, a browser) requests an object from CloudFront.
2. CloudFront forwards the request to the Amazon S3 bucket that is the origin for your distribution.
3. Amazon S3 returns an HTTP status code 301 (Moved Permanently) as well as the new location.
4. CloudFront caches the redirect status code and the new location, and returns the values to the viewer. CloudFront does not follow the redirect to get the object from the new location.
5. The viewer sends another request for the object, but this time the viewer specifies the new location that it got from CloudFront:
 - If the Amazon S3 bucket is redirecting all requests to a CloudFront distribution, using either the domain name for the distribution or an alternate domain name, CloudFront requests the object from the Amazon S3 bucket or the HTTP server in the new location. When the new location returns the object, CloudFront returns it to the viewer and caches it in an edge location.
 - If the Amazon S3 bucket is redirecting requests to another location, the second request bypasses CloudFront. The Amazon S3 bucket or the HTTP server in the new location returns the object directly to the viewer, so the object is never cached in a CloudFront edge cache.

Request and Response Behavior for Custom Origins

Topics

- [How CloudFront Processes and Forwards Requests to Your Custom Origin Server \(p. 190\)](#)
- [How CloudFront Processes Responses from Your Custom Origin Server \(p. 199\)](#)

How CloudFront Processes and Forwards Requests to Your Custom Origin Server

For information about how CloudFront processes viewer requests and forwards the requests to your custom origin, see the applicable topic:

Topics

- [Authentication \(p. 191\)](#)
- [Caching Duration and Minimum TTL \(p. 191\)](#)
- [Client IP Addresses \(p. 192\)](#)
- [Client-Side SSL Authentication \(p. 192\)](#)
- [Compression \(p. 192\)](#)
- [Conditional Requests \(p. 192\)](#)
- [Cookies \(p. 193\)](#)
- [Cross-Origin Resource Sharing \(CORS\) \(p. 193\)](#)
- [Encryption \(p. 193\)](#)
- [GET Requests That Include a Body \(p. 193\)](#)
- [HTTP Methods \(p. 193\)](#)
- [HTTP Request Headers and CloudFront Behavior \(p. 194\)](#)
- [HTTP Version \(p. 197\)](#)
- [Maximum Length of a Request and Maximum Length of a URL \(p. 197\)](#)
- [OCSP Stapling \(p. 197\)](#)
- [Persistent Connections \(p. 198\)](#)
- [Protocols \(p. 198\)](#)
- [Query Strings \(p. 198\)](#)
- [Origin Response Timeout \(p. 198\)](#)
- [Simultaneous Requests for the Same Object \(Traffic Spikes\) \(p. 199\)](#)
- [User-Agent Header \(p. 199\)](#)

Authentication

For `DELETE`, `GET`, `HEAD`, `PATCH`, `POST`, and `PUT` requests, if you configure CloudFront to forward the `Authorization` header to your origin, you can configure your origin server to request client authentication.

For `OPTIONS` requests, you can configure your origin server to request client authentication only if you use the following CloudFront settings:

- Configure CloudFront to forward the `Authorization` header to your origin
- Configure CloudFront to *not* cache the response to `OPTIONS` requests

You can configure CloudFront to forward requests to your origin using either HTTP or HTTPS; for more information, see [Using HTTPS with CloudFront \(p. 278\)](#).

Caching Duration and Minimum TTL

For web distributions, to control how long your objects stay in a CloudFront cache before CloudFront forwards another request to your origin, you can:

- Configure your origin to add a `Cache-Control` or an `Expires` header field to each object.
- Specify a value for Minimum TTL in CloudFront cache behaviors.
- Use the default value of 24 hours.

For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

Client IP Addresses

If a viewer sends a request to CloudFront and does not include an `X-Forwarded-For` request header, CloudFront gets the IP address of the viewer from the TCP connection, adds an `X-Forwarded-For` header that includes the IP address, and forwards the request to the origin. For example, if CloudFront gets the IP address 192.0.2.2 from the TCP connection, it forwards the following header to the origin:

`X-Forwarded-For: 192.0.2.2`

If a viewer sends a request to CloudFront and includes an `X-Forwarded-For` request header, CloudFront gets the IP address of the viewer from the TCP connection, appends it to the end of the `x-Forwarded-For` header, and forwards the request to the origin. For example, if the viewer request includes `X-Forwarded-For: 192.0.2.4,192.0.2.3` and CloudFront gets the IP address 192.0.2.2 from the TCP connection, it forwards the following header to the origin:

`X-Forwarded-For: 192.0.2.4,192.0.2.3,192.0.2.2`

Some applications, such as load balancers (including Elastic Load Balancing), web application firewalls, reverse proxies, intrusion prevention systems, and API Gateway, append the IP address of the CloudFront edge server that forwarded the request onto the end of the `x-Forwarded-For` header. For example, if CloudFront includes `X-Forwarded-For: 192.0.2.2` in a request that it forwards to ELB and if the IP address of the CloudFront edge server is 192.0.2.199, the request that your EC2 instance receives contains the following header:

`X-Forwarded-For: 192.0.2.2,192.0.2.199`

Note

The `x-Forwarded-For` header contains IPv4 addresses (such as 192.0.2.44) and IPv6 addresses (such as 2001:0db8:85a3:0000:0000:8a2e:0370:7334), as applicable.

Client-Side SSL Authentication

CloudFront does not support client authentication with client-side SSL certificates. If an origin requests a client-side certificate, CloudFront drops the request.

Compression

CloudFront forwards requests that have the `Accept-Encoding` field values "identity" and "gzip". For more information, see [Serving Compressed Files \(p. 179\)](#).

Conditional Requests

When CloudFront receives a request for an object that has expired from an edge cache, it forwards the request to the origin either to get the latest version of the object or to get confirmation from the origin that the CloudFront edge cache already has the latest version. Typically, when the origin last sent the object to CloudFront, it included an `ETag` value, a `LastModified` value, or both values in the response. In the new request that CloudFront forwards to the origin, CloudFront adds one or both of the following:

- An `If-Match` or `If-None-Match` header that contains the `ETag` value for the expired version of the object.
- An `If-Modified-Since` header that contains the `LastModified` value for the expired version of the object.

The origin uses this information to determine whether the object has been updated and, therefore, whether to return the entire object to CloudFront or to return only an HTTP 304 status code (not modified).

Cookies

You can configure CloudFront to forward cookies to your origin. For more information, see [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).

Cross-Origin Resource Sharing (CORS)

If you want CloudFront to respect cross-origin resource sharing settings, configure CloudFront to forward the `Origin` header to your origin. For more information, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

Encryption

You can require viewers to use HTTPS to send requests to CloudFront and require CloudFront to forward requests to your custom origin by using the protocol that is used by the viewer. For more information, see the following distribution settings:

- [Viewer Protocol Policy \(p. 72\)](#)
- [Origin Protocol Policy \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 68\)](#)

CloudFront forwards HTTPS requests to the origin server using the SSLv3, TLSv1.0, TLSv1.1, and TLSv1.2 protocols. For custom origins, you can choose the SSL protocols that you want CloudFront to use when communicating with your origin:

- If you're using the CloudFront console, choose protocols by using the **Origin SSL Protocols** check boxes. For more information, see [Creating or Updating a Web Distribution Using the CloudFront Console \(p. 60\)](#).
- If you're using the CloudFront API, specify protocols by using the `OriginSslProtocols` element. For more information, see [DistributionConfig Complex Type](#) in the *Amazon CloudFront API Reference*.

If the origin is an Amazon S3 bucket, CloudFront always uses TLSv1.2.

Important

Other versions of SSL and TLS are not supported.

For more information about using HTTPS with CloudFront, see [Using HTTPS with CloudFront \(p. 278\)](#).

For lists of the ciphers that CloudFront supports for HTTPS communication between viewers and CloudFront, and between CloudFront and your origin, see [Supported SSL/TLS Protocols and Ciphers for Communication Between Viewers and CloudFront \(p. 296\)](#).

GET Requests That Include a Body

If a viewer GET request includes a body, CloudFront returns an HTTP status code 403 (Forbidden) to the viewer.

HTTP Methods

If you configure CloudFront to process all of the HTTP methods that it supports, CloudFront accepts the following requests from viewers and forwards them to your custom origin:

- `DELETE`
- `GET`
- `HEAD`
- `OPTIONS`
- `PATCH`

- POST
- PUT

CloudFront always caches responses to `GET` and `HEAD` requests. You can also configure CloudFront to cache responses to `OPTIONS` requests. CloudFront does not cache responses to requests that use the other methods.

For information about configuring whether your custom origin processes these methods, see the documentation for your origin.

Important

If you configure CloudFront to accept and forward to your origin all of the HTTP methods that CloudFront supports, configure your origin server to handle all methods. For example, if you configure CloudFront to accept and forward these methods because you want to use `POST`, you must configure your origin server to handle `DELETE` requests appropriately so viewers can't delete resources that you don't want them to. For more information, see the documentation for your HTTP server.

HTTP Request Headers and CloudFront Behavior

The following table lists HTTP request headers and, for each header, explains the following:

- CloudFront behavior if you don't configure CloudFront to forward the header to your origin, which causes CloudFront to cache your objects based on header values.
- Whether you can configure CloudFront to cache objects based on header values for that header.

You can configure CloudFront to cache objects based on values in the `Date` and `User-Agent` headers, but we don't recommend it. These headers have a lot of possible values, and caching based on their values would cause CloudFront to forward significantly more requests to your origin.

For more information about caching based on header values, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

| Header | Behavior If You Don't Configure CloudFront to Cache Based on Header Values | Caching Based on Header Values Is Supported |
|------------------------------|--|---|
| Other-defined headers | CloudFront forwards the headers to your origin. | Yes |
| <code>Accept</code> | CloudFront removes the header. | Yes |
| <code>Accept-Charset</code> | CloudFront removes the header. | Yes |
| <code>Accept-Encoding</code> | If the value contains <code>gzip</code> , CloudFront forwards <code>Accept-Encoding: gzip</code> to your origin. If the value does not contain <code>gzip</code> , CloudFront removes the <code>Accept-Encoding</code> header field before forwarding the request to your origin. | No |
| <code>Accept-Language</code> | CloudFront removes the header. | Yes |
| <code>Authorization</code> | <ul style="list-style-type: none">• <code>GET</code> and <code>HEAD</code> requests – CloudFront removes the <code>Authorization</code> header field before forwarding the request to your origin. | Yes |

| Header | Behavior If You Don't Configure CloudFront to Cache Based on Header Values | Caching Based on Header Values Is Supported |
|------------------------------|---|---|
| | <ul style="list-style-type: none"> OPTIONS requests – CloudFront removes the Authorization header field before forwarding the request to your origin if you configure CloudFront to cache responses to OPTIONS requests. <p>CloudFront forwards the Authorization header field to your origin if you do not configure CloudFront to cache responses to OPTIONS requests.</p> <ul style="list-style-type: none"> DELETE, PATCH, POST, and PUT requests – CloudFront does not remove the header field before forwarding the request to your origin. | |
| Cache-Control | CloudFront forwards the header to your origin. | No |
| CloudFront-Forwarded-Proto | <p>CloudFront does not add the header before forwarding the request to your origin.</p> <p>For more information, see Configuring CloudFront to Cache Objects Based on the Protocol of the Request (p. 155).</p> | Yes |
| CloudFront-Is-Desktop-Viewer | <p>CloudFront does not add the header before forwarding the request to your origin.</p> <p>For more information, see Configuring CloudFront to Cache Objects Based on the Device Type (p. 154).</p> | Yes |
| CloudFront-Is-Mobile-Viewer | <p>CloudFront does not add the header before forwarding the request to your origin.</p> <p>For more information, see Configuring CloudFront to Cache Objects Based on the Device Type (p. 154).</p> | Yes |
| CloudFront-Is-Tablet-Viewer | <p>CloudFront does not add the header before forwarding the request to your origin.</p> <p>For more information, see Configuring CloudFront to Cache Objects Based on the Device Type (p. 154).</p> | Yes |
| CloudFront-Viewer-Country | CloudFront does not add the header before forwarding the request to your origin. | Yes |
| Connection | CloudFront replaces this header with Connection: Keep-Alive before forwarding the request to your origin. | No |
| Content-Length | CloudFront forwards the header to your origin. | No |
| Content-MD5 | CloudFront forwards the header to your origin. | Yes |
| Content-Type | CloudFront forwards the header to your origin. | Yes |

| Header | Behavior If You Don't Configure CloudFront to Cache Based on Header Values | Caching Based on Header Values Is Supported |
|----------------------------------|---|--|
| <code>Cookie</code> | If you configure CloudFront to forward cookies, it will forward the <code>Cookie</code> header field to your origin. If you don't, CloudFront removes the <code>Cookie</code> header field. For more information, see Configuring CloudFront to Cache Objects Based on Cookies (p. 150) . | No |
| <code>Date</code> | CloudFront forwards the header to your origin. | Yes, but not recommended |
| <code>Expect</code> | CloudFront removes the header. | Yes |
| <code>From</code> | CloudFront forwards the header to your origin. | Yes |
| <code>Host</code> | CloudFront sets the value to the domain name of the origin that is associated with the requested object. | Yes |
| <code>If-Match</code> | CloudFront forwards the header to your origin. | Yes |
| <code>If-Modified-Since</code> | CloudFront forwards the header to your origin. | Yes |
| <code>If-None-Match</code> | CloudFront forwards the header to your origin. | Yes |
| <code>If-Range</code> | CloudFront forwards the header to your origin. | Yes |
| <code>If-Unmodified-Since</code> | CloudFront forwards the header to your origin. | Yes |
| <code>Max-Forwards</code> | CloudFront forwards the header to your origin. | No |
| <code>Origin</code> | CloudFront forwards the header to your origin. | Yes |
| <code>Pragma</code> | CloudFront forwards the header to your origin. | No |
| <code>Proxy-Authenticate</code> | CloudFront removes the header. | No |
| <code>Proxy-Authorization</code> | CloudFront removes the header. | No |
| <code>Proxy-Connection</code> | CloudFront removes the header. | No |
| <code>Range</code> | CloudFront forwards the header to your origin. For more information, see How CloudFront Processes Partial Requests for an Object (Range GETs) (p. 176) . | Yes, by default |
| <code>Referer</code> | CloudFront removes the header. | Yes |
| <code>Request-Range</code> | CloudFront forwards the header to your origin. | No |
| <code>TE</code> | CloudFront removes the header. | No |
| <code>Trailer</code> | CloudFront removes the header. | No |
| <code>Transfer-Encoding</code> | CloudFront forwards the header to your origin. | No |
| <code>Upgrade</code> | CloudFront removes the header. | No |

| Header | Behavior If You Don't Configure CloudFront to Cache Based on Header Values | Caching Based on Header Values Is Supported |
|-------------------|--|---|
| User-Agent | CloudFront replaces the value of this header field with <code>Amazon CloudFront</code> . If you want CloudFront to cache your content based on the device the user is using, see Configuring CloudFront to Cache Objects Based on the Device Type (p. 154) . | Yes, but not recommended |
| Via | CloudFront forwards the header to your origin. | Yes |
| Warning | CloudFront forwards the header to your origin. | Yes |
| X-Amz-Cf-Id | CloudFront adds the header to the viewer request before forwarding the request to your origin. The header value contains an encrypted string that uniquely identifies the request. | No |
| X-Edge-* | CloudFront removes all X-Edge-* headers. | No |
| X-Forwarded-For | CloudFront forwards the header to your origin. For more information, see Client IP Addresses (p. 192) . | Yes |
| X-Forwarded-Proto | CloudFront removes the header. | Yes |
| X-Real-IP | CloudFront removes the header. | No |

HTTP Version

CloudFront forwards requests to your custom origin using HTTP/1.1.

Maximum Length of a Request and Maximum Length of a URL

The maximum length of a request, including the path, the query string (if any), and headers, is 20,480 bytes.

CloudFront constructs a URL from the request. The maximum length of this URL is 8192 bytes.

If a request or a URL exceeds these limits, CloudFront returns HTTP status code 413, Request Header Fields Too Large, to the viewer, and then terminates the TCP connection to the viewer.

OCSP Stapling

When a viewer submits an HTTPS request for an object, either CloudFront or the viewer must confirm with the certificate authority (CA) that the SSL certificate for the domain has not been revoked. OCSP stapling speeds up certificate validation by allowing CloudFront to validate the certificate and to cache the response from the CA, so the client doesn't need to validate the certificate directly with the CA.

The performance improvement of OCSP stapling is more pronounced when CloudFront receives a lot of HTTPS requests for objects in the same domain. Each server in a CloudFront edge location must submit a separate validation request. When CloudFront receives a lot of HTTPS requests for the same domain, every server in the edge location soon has a response from the CA that it can "staple" to a packet in the SSL handshake; when the viewer is satisfied that the certificate is valid, CloudFront can

serve the requested object. If your distribution doesn't get much traffic in a CloudFront edge location, new requests are more likely to be directed to a server that hasn't validated the certificate with the CA yet. In that case, the viewer separately performs the validation step and the CloudFront server serves the object. That CloudFront server also submits a validation request to the CA, so the next time it receives a request that includes the same domain name, it has a validation response from the CA.

Persistent Connections

When CloudFront gets a response from your origin, it tries to maintain the connection for several seconds in case another request arrives during that period. Maintaining a persistent connection saves the time required to re-establish the TCP connection and perform another TLS handshake for subsequent requests.

For more information, including how to configure the duration of persistent connections, see [Origin Keep-alive Timeout \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 69\)](#) in the section [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

Protocols

CloudFront forwards HTTP or HTTPS requests to the origin server based on the following:

- The protocol of the request that the viewer sends to CloudFront, either HTTP or HTTPS.
- The value of the **Origin Protocol Policy** field in the CloudFront console or, if you're using the CloudFront API, the `OriginProtocolPolicy` element in the `DistributionConfig` complex type. In the CloudFront console, the options are **HTTP Only**, **HTTPS Only**, and **Match Viewer**.

If you specify **HTTP Only** or **HTTPS Only**, CloudFront forwards requests to the origin server using the specified protocol, regardless of the protocol in the viewer request.

If you specify **Match Viewer**, CloudFront forwards requests to the origin server using the protocol in the viewer request. Note that CloudFront caches the object only once even if viewers make requests using both HTTP and HTTPS protocols.

Important

If CloudFront forwards a request to the origin using the HTTPS protocol, and if the origin server returns an invalid certificate or a self-signed certificate, CloudFront drops the TCP connection.

For information about how to update a distribution using the CloudFront console, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#). For information about how to update a distribution using the CloudFront API, go to [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

Query Strings

You can configure whether CloudFront forwards query string parameters to your origin. For more information, see [Configuring CloudFront to Cache Based on Query String Parameters \(p. 147\)](#).

Origin Response Timeout

The origin response timeout, also known as the origin request timeout or origin read timeout, applies to both of the following values:

- The amount of time, in seconds, that CloudFront waits for a response after forwarding a request to a custom origin
- The amount of time, in seconds, that CloudFront waits after receiving a packet of a response from the origin and before receiving the next packet

For more information, including how to configure the origin response timeout, see [Origin Response Timeout \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 69\)](#) in the section [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

Simultaneous Requests for the Same Object (Traffic Spikes)

When a CloudFront edge location receives a request for an object and either the object isn't currently in the cache or the object has expired, CloudFront immediately sends the request to your origin. If there's a traffic spike—if additional requests for the same object arrive at the edge location before your origin responds to the first request—CloudFront pauses briefly before forwarding additional requests for the object to your origin. Typically, the response to the first request will arrive at the CloudFront edge location before the response to subsequent requests. This brief pause helps to reduce unnecessary load on your origin server. If additional requests are not identical because, for example, you configured CloudFront to cache based on request headers or cookies, CloudFront forwards all of the unique requests to your origin.

User-Agent Header

If you want CloudFront to cache different versions of your objects based on the device a user is using to view your content, we recommend that you configure CloudFront to forward the applicable headers to your custom origin:

- `CloudFront-Is-Desktop-Viewer`
- `CloudFront-Is-Mobile-Viewer`
- `CloudFront-Is-SmartTV-Viewer`
- `CloudFront-Is-Tablet-Viewer`

Based on the value of the `User-Agent` header, CloudFront sets the value of these headers to `true` or `false` before forwarding the request to your origin. If a device falls into more than one category, more than one value might be `true`. For example, for some tablet devices, CloudFront might set both `CloudFront-Is-Mobile-Viewer` and `CloudFront-Is-Tablet-Viewer` to `true`. For more information about configuring CloudFront to cache based on request headers, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

You can configure CloudFront to cache objects based on values in the `User-Agent` header, but we don't recommend it. The `User-Agent` header has a lot of possible values, and caching based on those values would cause CloudFront to forward significantly more requests to your origin.

If you do not configure CloudFront to cache objects based on values in the `User-Agent` header, CloudFront adds a `User-Agent` header with the following value before it forwards a request to your origin:

```
User-Agent = Amazon CloudFront
```

CloudFront adds this header regardless of whether the request from the viewer includes a `User-Agent` header. If the request from the viewer includes a `User-Agent` header, CloudFront removes it.

How CloudFront Processes Responses from Your Custom Origin Server

For information about how CloudFront processes responses from custom origin servers, see the applicable topic:

Topics

- [Caching \(p. 200\)](#)
- [Canceled Requests \(p. 200\)](#)
- [Content Negotiation \(p. 200\)](#)
- [Cookies \(p. 200\)](#)
- [Dropped TCP Connections \(p. 200\)](#)
- [HTTP Response Headers that CloudFront Removes or Updates \(p. 201\)](#)
- [Maximum File Size \(p. 201\)](#)
- [Origin Unavailable \(p. 202\)](#)
- [Redirects \(p. 202\)](#)
- [Transfer Encoding \(p. 202\)](#)

Caching

- Ensure that the origin server sets valid and accurate values for the `Date` and `Last-Modified` header fields.
- If requests from viewers include the `If-Match` or `If-None-Match` request header fields, set the `ETag` response header field. If you do not specify an `ETag` value, CloudFront ignores subsequent `If-Match` or `If-None-Match` headers.
- CloudFront normally respects a `Cache-Control: no-cache` header in the response from the origin. For an exception, see [Simultaneous Requests for the Same Object \(Traffic Spikes\) \(p. 199\)](#).

Canceled Requests

If an object is not in the edge cache, and if a viewer terminates a session (for example, closes a browser) after CloudFront gets the object from your origin but before it can deliver the requested object, CloudFront does not cache the object in the edge location.

Content Negotiation

If your origin returns `Vary: *` in the response, and if the value of **Minimum TTL** for the corresponding cache behavior is **0**, CloudFront caches the object but still forwards every subsequent request for the object to the origin to confirm that the cache contains the latest version of the object. CloudFront doesn't include any conditional headers, such as `If-None-Match` or `If-Modified-Since`. As a result, your origin returns the object to CloudFront in response to every request.

If your origin returns `Vary: *` in the response, and if the value of **Minimum TTL** for the corresponding cache behavior is any other value, CloudFront processes the `Vary` header as described in [HTTP Response Headers that CloudFront Removes or Updates \(p. 201\)](#).

Cookies

If you enable cookies for a cache behavior, and if the origin returns cookies with an object, CloudFront caches both the object and the cookies. Note that this reduces cacheability for an object. For more information, see [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).

Dropped TCP Connections

If the TCP connection between CloudFront and your origin drops while your origin is returning an object to CloudFront, CloudFront behavior depends on whether your origin included a `Content-Length` header in the response:

- **Content-Length header** – CloudFront returns the object to the viewer as it gets the object from your origin. However, if the value of the Content-Length header doesn't match the size of the object, CloudFront doesn't cache the object.
- **Transfer-Encoding: Chunked** – CloudFront returns the object to the viewer as it gets the object from your origin. However, if the chunked response is not complete, CloudFront does not cache the object.
- **No Content-Length header** – CloudFront returns the object to the viewer and caches it, but the object may not be complete. Without a Content-Length header, CloudFront cannot determine whether the TCP connection was dropped accidentally or on purpose.

We recommend that you configure your HTTP server to add a Content-Length header to prevent CloudFront from caching partial objects.

HTTP Response Headers that CloudFront Removes or Updates

CloudFront removes or updates the following header fields before forwarding the response from your origin to the viewer:

- Set-Cookie – If you configure CloudFront to forward cookies, it will forward the Set-Cookie header field to clients. For more information, see [Configuring CloudFront to Cache Objects Based on Cookies \(p. 150\)](#).
- Trailer
- Transfer-Encoding – If your origin returns this header field, CloudFront sets the value to chunked before returning the response to the viewer.
- Upgrade
- Vary – Note the following:
 - If you configure CloudFront to forward any of the device-specific headers to your origin (CloudFront-Is-Desktop-Viewer, CloudFront-Is-Mobile-Viewer, CloudFront-Is-SmartTV-Viewer, CloudFront-Is-Tablet-Viewer) and you configure your origin to return Vary:User-Agent to CloudFront, CloudFront returns Vary:User-Agent to the viewer. For more information, see [Configuring CloudFront to Cache Objects Based on the Device Type \(p. 154\)](#).
 - If you configure your origin to include either Accept-Encoding or Cookie in the Vary header, CloudFront includes the values in the response to the viewer.
 - If you configure CloudFront to forward a whitelist of headers to your origin, and if you configure your origin to return the header names to CloudFront in the Vary header (for example, Vary:Accept-Charset, Accept-Language), CloudFront returns the Vary header with those values to the viewer.
 - For information about how CloudFront processes a value of * in the Vary header, see [Content Negotiation \(p. 200\)](#).
 - If you configure your origin to include any other values in the Vary header, CloudFront removes the values before returning the response to the viewer.
- Via – Regardless of whether your origin returns this header field to CloudFront, CloudFront sets the value to:

`Via: 1.1 alphanumeric-string.cloudfront.net (CloudFront)`

before returning the response to the viewer. For example:

```
Via: 1.1 1026589cc7887e7a0dc7827b4example.cloudfront.net (CloudFront)
```

Maximum File Size

The maximum size of a response body that CloudFront will return to the viewer is 20 GB. This includes chunked transfer responses that don't specify the Content-Length header value.

Origin Unavailable

If your origin server is unavailable and CloudFront gets a request for an object that is in the edge cache but that has expired (for example, because the period of time specified in the Cache-Control max-age directive has passed), CloudFront either serves the expired version of the object or serves a custom error page. For more information, see [How CloudFront Processes and Caches HTTP 4xx and 5xx Status Codes from Your Origin \(p. 203\)](#).

In some cases, an object that is seldom requested is evicted and is no longer available in the edge cache. CloudFront can't serve an object that has been evicted.

Redirects

If you change the location of an object on the origin server, you can configure your web server to redirect requests to the new location. After you configure the redirect, the first time a viewer submits a request for the object, CloudFront sends the request to the origin, and the origin responds with a redirect (for example, 302 Moved Temporarily). CloudFront caches the redirect and returns it to the viewer. CloudFront does not follow the redirect.

You can configure your web server to redirect requests to one of the following locations:

- The new URL of the object on the origin server. When the viewer follows the redirect to the new URL, the viewer bypasses CloudFront and goes straight to the origin. As a result, we recommend that you not redirect requests to the new URL of the object on the origin.
- The new CloudFront URL for the object. When the viewer submits the request that contains the new CloudFront URL, CloudFront gets the object from the new location on your origin, caches it at the edge location, and returns the object to the viewer. Subsequent requests for the object will be served by the edge location. This avoids the latency and load associated with viewers requesting the object from the origin. However, every new request for the object will incur charges for two requests to CloudFront.

Transfer Encoding

CloudFront supports only the chunked value of the Transfer-Encoding header. If your origin returns Transfer-Encoding: chunked, CloudFront returns the object to the client as the object is received at the edge location, and caches the object in chunked format for subsequent requests.

If the viewer makes a Range GET request and the origin returns Transfer-Encoding: chunked, CloudFront returns the entire object to the viewer instead of the requested range.

We recommend that you use chunked encoding if the content length of your response cannot be predetermined. For more information, see [Dropped TCP Connections \(p. 200\)](#).

How CloudFront Processes HTTP 3xx Status Codes from Your Origin

When CloudFront requests an object from your Amazon S3 bucket or custom origin server, your origin sometimes returns an HTTP 3xx status code, which typically indicates either that the URL has changed (301, Moved Permanently, or 307, Temporary Redirect) or that the object hasn't changed since the last time CloudFront requested it (304, Not Modified). CloudFront caches 3xx responses for the duration specified by the settings in your CloudFront distribution and by the header fields that your origin returns along with an object. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

If your origin returns a 301 or 307 status code, CloudFront doesn't follow the redirect to the new location.

How CloudFront Processes and Caches HTTP 4xx and 5xx Status Codes from Your Origin

Topics

- [How CloudFront Processes Errors When You Have Configured Custom Error Pages \(p. 203\)](#)
- [How CloudFront Processes Errors When You Have Not Configured Custom Error Pages \(p. 204\)](#)
- [HTTP 4xx and 5xx Status Codes that CloudFront Caches \(p. 205\)](#)

When CloudFront requests an object from your Amazon S3 bucket or custom origin server, your origin sometimes returns an HTTP 4xx or 5xx status code, which indicates an error has occurred. CloudFront behavior depends on:

- Whether you have configured custom error pages.
- Whether you have configured how long you want CloudFront to cache error responses from your origin (error caching minimum TTL).
- The status code.
- For 5xx status codes, whether the requested object is currently in the CloudFront edge cache.

CloudFront always caches responses to GET and HEAD requests. You can also configure CloudFront to cache responses to OPTIONS requests. CloudFront does not cache responses to requests that use the other methods.

For information about settings for custom error pages in the CloudFront console, see [Custom Error Pages and Error Caching \(p. 82\)](#). For information about the error caching minimum TTL in the CloudFront console, see [Error Caching Minimum TTL \(p. 83\)](#).

For a list of the HTTP status codes that CloudFront caches, see [HTTP 4xx and 5xx Status Codes that CloudFront Caches \(p. 205\)](#).

If you have enabled logging, CloudFront writes the results to the logs regardless of the HTTP status code.

How CloudFront Processes Errors When You Have Configured Custom Error Pages

If you have configured custom error pages, CloudFront behavior depends on whether the requested object is in the edge cache.

The Requested Object Is Not in the Edge Cache

CloudFront continues to try to get the requested object from your origin when all of the following are true:

- A viewer requests an object
- The object isn't in the edge cache
- Your origin returns an HTTP 4xx or 5xx status code instead of returning a 304 status code (Not Modified) or an updated version of the object

CloudFront does the following:

1. In the CloudFront edge cache that received the viewer request, CloudFront checks your distribution configuration and gets the path of the custom error page that corresponds with the status code that your origin returned.
2. CloudFront finds the first cache behavior in your distribution that has a path pattern that matches the path of the custom error page.
3. The CloudFront edge location sends a request for the custom error page to the origin that is specified in the cache behavior.
4. The origin returns the custom error page to the edge location.
5. CloudFront returns the custom error page to the viewer that made the request, and also caches the custom error page for the amount of time specified by the error caching minimum TTL (five minutes by default).
6. After the error caching minimum TTL has elapsed, CloudFront tries again to get the requested object by forwarding another request to your origin. CloudFront continues to retry at intervals specified by the error caching minimum TTL.

The Requested Object Is in the Edge Cache

CloudFront continues to serve the object that is currently in the edge cache when all of the following are true:

- A viewer requests an object
- The object is in the edge cache but it has expired
- Your origin returns an HTTP 5xx status code instead of returning a 304 status code (Not Modified) or an updated version of the object

CloudFront does the following:

1. If your origin returns a 5xx status code, CloudFront serves the object even though it has expired. For the duration of the error caching minimum TTL, CloudFront continues to respond to viewer requests by serving the object from the edge cache.

If your origin returns a 4xx status code, CloudFront returns the status code, not the requested object, to the viewer.

2. After the error caching minimum TTL has elapsed, CloudFront tries again to get the requested object by forwarding another request to your origin. Note that if the object is not requested frequently, CloudFront might evict it from the edge cache while your origin server is still returning 5xx responses. For information about how long objects stay in CloudFront edge caches, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\)](#) (p. 160).

How CloudFront Processes Errors When You Have Not Configured Custom Error Pages

If you have not configured custom error pages, CloudFront behavior depends on whether the requested object is in the edge cache.

The Requested Object Is Not in the Edge Cache

CloudFront continues to try to get the requested object from your origin when all of the following are true:

- A viewer requests an object
- The object isn't in the edge cache
- Your origin returns an HTTP 4xx or 5xx status code instead of returning a 304 status code (Not Modified) or an updated version of the object

CloudFront does the following:

1. CloudFront returns the 4xx or 5xx status code to the viewer.
2. CloudFront also caches the status code in the edge cache that received the request.
3. For the duration of the error caching minimum TTL (five minutes by default), CloudFront responds to subsequent viewer requests for the same object with the cached 4xx or 5xx status code.
4. After the error caching minimum TTL has elapsed, CloudFront tries again to get the requested object by forwarding another request to your origin.

The Requested Object Is in the Edge Cache

CloudFront continues to serve the object that is currently in the edge cache when all of the following are true:

- A viewer requests an object
- The object is in the edge cache but it has expired
- Your origin returns an HTTP 5xx status code instead of returning a 304 status code (Not Modified) or an updated version of the object

CloudFront does the following:

1. If your origin returns a 5xx error code, CloudFront serves the object even though it has expired. For the duration of the error caching minimum TTL (five minutes by default), CloudFront continues to respond to viewer requests by serving the object from the edge cache.

If your origin returns a 4xx status code, CloudFront returns the status code, not the requested object, to the viewer.

2. After the error caching minimum TTL has elapsed, CloudFront tries again to get the requested object by forwarding another request to your origin. Note that if the object is not requested frequently, CloudFront might evict it from the edge cache while your origin server is still returning 5xx responses. For information about how long objects stay in CloudFront edge caches, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Expiration\) \(p. 160\)](#).

HTTP 4xx and 5xx Status Codes that CloudFront Caches

CloudFront caches the following HTTP 4xx and 5xx status codes returned by Amazon S3 or your custom origin server. If you have configured a custom error page for an HTTP status code, CloudFront caches the custom error page.

| | |
|-----|-------------|
| 400 | Bad Request |
| 403 | Forbidden |
| 404 | Not Found |

| | |
|-----|-----------------------|
| 405 | Method Not Allowed |
| 414 | Request-URI Too Large |
| 500 | Internal Server Error |
| 501 | Not Implemented |
| 502 | Bad Gateway |
| 503 | Service Unavailable |
| 504 | Gateway Time-out |

Serving Private Content through CloudFront

Many companies that distribute content via the internet want to restrict access to documents, business data, media streams, or content that is intended for selected users, for example, users who have paid a fee. To securely serve this private content using CloudFront, you can do the following:

- Require that your users access your private content by using special CloudFront signed URLs or signed cookies.
- Require that your users access your Amazon S3 content using CloudFront URLs, not Amazon S3 URLs. Requiring CloudFront URLs isn't required, but we recommend it to prevent users from bypassing the restrictions that you specify in signed URLs or signed cookies.

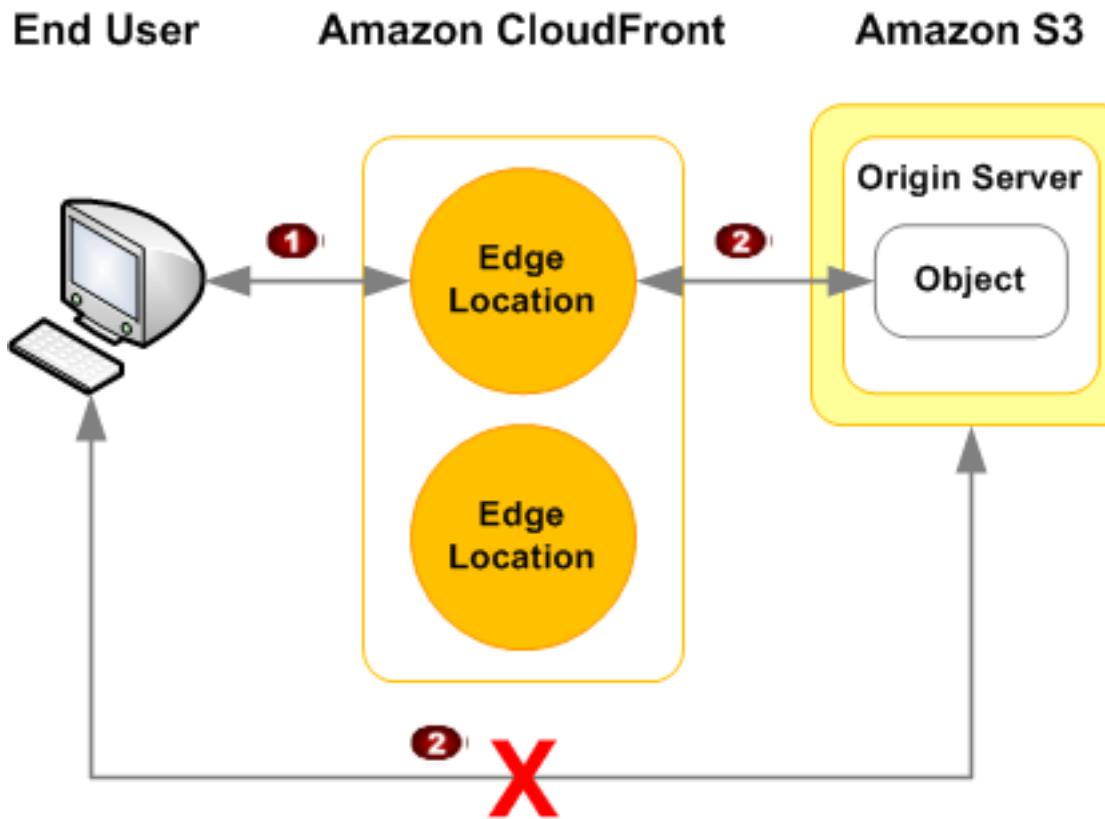
Topics

- [Overview of Private Content \(p. 207\)](#)
- [Using an HTTP Server for Private Content \(p. 209\)](#)
- [Task List: Serving Private Content \(p. 210\)](#)
- [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#)
- [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#)
- [Choosing Between Signed URLs and Signed Cookies \(p. 223\)](#)
- [Using Signed URLs \(p. 224\)](#)
- [Using Signed Cookies \(p. 244\)](#)
- [Using a Linux Command and OpenSSL for Base64-Encoding and Encryption \(p. 259\)](#)
- [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#)

Overview of Private Content

You can control user access to your private content in two ways, as shown in the following illustration:

1. Restrict access to objects in CloudFront edge caches
2. Restrict access to objects in your Amazon S3 bucket



Restricting Access to Objects in CloudFront Edge Caches

You can configure CloudFront to require that users access your objects using either **signed URLs** or **signed cookies**. You then develop your application either to create and distribute signed URLs to authenticated users or to send Set-Cookie headers that set signed cookies on the viewers for authenticated users. (To give a few users long-term access to a limited number of objects, you can also create signed URLs manually.)

When you create signed URLs or signed cookies to control access to your objects, you can specify the following restrictions:

- An ending date and time, after which the URL is no longer valid.
- (Optional) The date and time that the URL becomes valid.
- (Optional) The IP address or range of addresses of the computers that can be used to access your content.

One part of a signed URL or a signed cookie is hashed and signed using the private key from a public/private key pair. When someone uses a signed URL or signed cookie to access an object, CloudFront compares the signed and unsigned portions of the URL or cookie. If they don't match, CloudFront doesn't serve the object.

Restricting Access to Objects in Amazon S3 Buckets

You can optionally secure the content in your Amazon S3 bucket so users can access it through CloudFront but cannot access it directly by using Amazon S3 URLs. This prevents anyone from bypassing CloudFront and using the Amazon S3 URL to get content that you want to restrict access to. This step isn't required to use signed URLs, but we recommend it.

To require that users access your content through CloudFront URLs, you perform the following tasks:

- Create a special CloudFront user called an **origin access identity**.
- Give the origin access identity permission to read the objects in your bucket.
- Remove permission for anyone else to use Amazon S3 URLs to read the objects.

Using an HTTP Server for Private Content

You can use signed URLs or signed cookies for any CloudFront distribution, regardless of whether the origin is an Amazon S3 bucket or an HTTP server. However, for CloudFront to get your objects from an HTTP server, the objects must remain publicly accessible. When the objects are publicly accessible, anyone who has the URL for an object on your HTTP server can access the object without logging in or paying for your content. If you use signed URLs or signed cookies and your origin is an HTTP server, do not give the URLs for the objects on your HTTP server to your customers or to others outside your organization.

Task List: Serving Private Content

To configure CloudFront to serve private content, perform the following tasks:

1. (Optional but recommended) Require your users to access your content only through CloudFront. The method that you use depends on whether you're using Amazon S3 or custom origins:
 - **Amazon S3** – See [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).
 - **Custom origin** – See [Using Custom Headers to Restrict Access to Your Content on a Custom Origin \(p. 158\)](#).
2. Specify the AWS accounts that you want to use to create signed URLs or signed cookies. For more information, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#).
3. Write your application to respond to requests from authorized users either with signed URLs or with Set-Cookie headers that set signed cookies. For more information, see [Choosing Between Signed URLs and Signed Cookies \(p. 223\)](#).

Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content

Topics

- [Creating a CloudFront Origin Access Identity and Adding it to Your Distribution \(p. 212\)](#)
- [Granting the Origin Access Identity Permission to Read Objects in Your Amazon S3 Bucket \(p. 213\)](#)
- [Using an Origin Access Identity in Amazon S3 Regions that Support Only Signature Version 4 Authentication \(p. 215\)](#)

Typically, if you're using an Amazon S3 bucket as the origin for a CloudFront distribution, you grant everyone permission to read the objects in your bucket. This allows anyone to access your objects either through CloudFront or using the Amazon S3 URL. CloudFront doesn't expose Amazon S3 URLs, but your users might have those URLs if your application serves any objects directly from Amazon S3 or if anyone gives out direct links to specific objects in Amazon S3.

Note

You can also restrict access to content on a custom origin by using custom headers. For more information, see [Using Custom Headers to Restrict Access to Your Content on a Custom Origin \(p. 158\)](#).

If you want to use CloudFront signed URLs or signed cookies to provide access to objects in your Amazon S3 bucket, you probably also want to prevent users from accessing your Amazon S3 objects using Amazon S3 URLs. If users access your objects directly in Amazon S3, they bypass the controls provided by CloudFront signed URLs or signed cookies, for example, control over the date and time that a user can no longer access your content and control over which IP addresses can be used to access content. In addition, if users access objects both through CloudFront and directly by using Amazon S3 URLs, CloudFront access logs are less useful because they're incomplete.

Note

To create origin access identities, you must use the CloudFront console or CloudFront API version 2009-09-09 or later.

To ensure that your users access your objects using only CloudFront URLs, regardless of whether the URLs are signed, perform the following tasks:

1. Create an origin access identity, which is a special CloudFront user, and associate the origin access identity with your distribution. (For web distributions, you associate the origin access identity with origins, so you can secure all or just some of your Amazon S3 content.) You can also create an origin access identity and add it to your distribution when you create the distribution. For more information, see [Creating a CloudFront Origin Access Identity and Adding it to Your Distribution \(p. 212\)](#).
2. Change the permissions either on your Amazon S3 bucket or on the objects in your bucket so only the origin access identity has read permission (or read and download permission). When your users access your Amazon S3 objects through CloudFront, the CloudFront origin access identity gets the objects on behalf of your users. If your users request objects directly by using Amazon S3 URLs, they're denied access. The origin access identity has permission to access objects in your Amazon S3 bucket, but users don't. For more information, see [Granting the Origin Access Identity Permission to Read Objects in Your Amazon S3 Bucket \(p. 213\)](#).

Creating a CloudFront Origin Access Identity and Adding it to Your Distribution

An AWS account can have up to 100 CloudFront origin access identities. However, you can add an origin access identity to as many distributions as you want, so one origin access identity is usually sufficient.

If you didn't create an origin access identity and add it to your distribution when you created the distribution, you can create and add one now using either the CloudFront console or the CloudFront API:

- **If you're using the CloudFront console** – You can create an origin access identity and add it to your distribution at the same time. For more information, see [Creating an Origin Access Identity and Adding it to Your Distribution Using the CloudFront Console \(p. 212\)](#).
- **If you're using the CloudFront API** – You create an origin access identity and then you add it to your distribution. Perform the procedure in each of the following topics:
 - [Creating an Origin Access Identity Using the CloudFront API \(p. 213\)](#)
 - [Adding an Origin Access Identity to Your Distribution Using the CloudFront API \(p. 213\)](#)

Creating an Origin Access Identity and Adding it to Your Distribution Using the CloudFront Console

If you didn't create an origin access identity when you created your distribution, perform the following procedure.

To create a CloudFront origin access identity using the CloudFront console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click the ID of the distribution that you want to add an origin access identity to.
3. Change to edit mode:
 - **Web distributions** – Click the **Origins** tab, select the origin that you want to edit, and click **Edit**. You can only create an origin access identity for origins for which **Origin Type** is **S3 Origin**.
 - **RTMP distributions** – Click **Edit**.
4. For **Restrict Bucket Access**, click **Yes**.
5. If you already have an origin access identity that you want to use, click **Use an Existing Identity**. Then select the identity in the **Your Identities** list.

Note

If you already have an origin access identity, we recommend that you reuse it to simplify maintenance.

If you want to create an identity, click **Create a New Identity**. Then enter a description for the identity in the **Comment** field.

6. If you want CloudFront to automatically give the origin access identity permission to read the objects in the Amazon S3 bucket specified in **Origin Domain Name**, click **Yes, Update Bucket Policy**.

Important

If you click **Yes, Update Bucket Policy**, CloudFront updates bucket permissions to grant the specified origin access identity the permission to read objects in your bucket. However, CloudFront does not remove existing permissions. If users currently have permission to access the objects in your bucket using Amazon S3 URLs, they will still have that permission after CloudFront updates your bucket permissions. To view or remove existing bucket

permissions, use a method provided by Amazon S3. For more information, see [Granting the Origin Access Identity Permission to Read Objects in Your Amazon S3 Bucket \(p. 213\)](#).

If you want to manually update permissions on your Amazon S3 bucket, click **No, I Will Update Permissions**.

7. Click **Yes, Edit**.
8. If you're adding an origin access identity to a web distribution and you have more than one origin, repeat step 3 through step 7 as applicable.

Creating an Origin Access Identity Using the CloudFront API

If you already have an origin access identity and you want to reuse it instead of creating another one, skip to [Adding an Origin Access Identity to Your Distribution Using the CloudFront API \(p. 213\)](#).

To create a CloudFront origin access identity using the CloudFront API, use the `POST Origin Access Identity` API action. The response includes an `Id` and an `S3CanonicalUserId` for the new origin access identity. Make note of these values because you will use them later in the process:

- **Id element** – You use the value of the `Id` element to associate an origin access ID with your distribution.
- **S3CanonicalUserId element** – You use the value of the `S3CanonicalUserId` element when you give CloudFront access to your Amazon S3 bucket or objects.

For more information about the `POST Origin Access Identity` API action, go to [POST Origin Access Identity](#) in the *Amazon CloudFront API Reference*. For a list of other actions that you can perform on origin access identities, go to [Actions on Origin Access Identities](#), also in the *Amazon CloudFront API Reference*.

Adding an Origin Access Identity to Your Distribution Using the CloudFront API

You can use the CloudFront API to add a CloudFront origin access identity to an existing distribution or to create a new distribution that includes an origin access identity. In either case, include an `OriginAccessIdentity` element. This element contains the value of the `Id` element that the `POST Origin Access Identity` API action returned when you created the origin access identity. For web distributions, add the `OriginAccessIdentity` element to one or more origins. For RTMP distributions, add the `OriginAccessIdentity` element to the distribution.

See the applicable topic in the *Amazon CloudFront API Reference*:

- **Create a new web distribution** – [POST Distribution](#)
- **Update an existing web distribution** – [PUT Distribution Config](#)
- **Create a new RTMP distribution** – [POST Streaming Distribution](#)
- **Update an existing RTMP distribution** – [PUT Streaming Distribution Config](#)

Granting the Origin Access Identity Permission to Read Objects in Your Amazon S3 Bucket

When you create or update a distribution, you can add an origin access identity and automatically update the bucket policy to give the origin access identity permission to access your bucket. Alternatively, you can choose to manually change the bucket policy or change ACLs, which control permissions on individual objects in your bucket.

Whichever method you use, you should still review the bucket policy for your bucket and review the permissions on your objects to ensure that:

- CloudFront can access objects in the bucket on behalf of users who are requesting your objects through CloudFront.
- Users can't use Amazon S3 URLs to access your objects.

Important

If you configure CloudFront to accept and forward to Amazon S3 all of the HTTP methods that CloudFront supports, create a CloudFront origin access identity to restrict access to your Amazon S3 content, and grant the origin access identity the applicable permissions. For example, if you configure CloudFront to accept and forward these methods because you want to use the `PUT` method, you must configure Amazon S3 bucket policies or ACLs to handle `DELETE` requests appropriately so users can't delete resources that you don't want them to.

Note the following:

- You might find it easier to update Amazon S3 bucket policies than ACLs because you can add objects to the bucket without updating permissions. However, ACLs give you more fine-grained control because you're granting permissions on each object.
- By default, your Amazon S3 bucket and all of the objects in it are private—only the AWS account that created the bucket has permission to read or write the objects in it.
- If you're adding an origin access identity to an existing distribution, modify the bucket policy or any object ACLs as appropriate to ensure that the objects are not publicly available.
- Grant additional permissions to one or more secure administrator accounts so you can continue to update the contents of the Amazon S3 bucket.

Important

There might be a brief delay between when you save your changes to Amazon S3 permissions and when the changes take effect. Until the changes take effect, you can get permission-denied errors when you try to access objects in your bucket.

Updating Amazon S3 Bucket Policies

You can update the Amazon S3 bucket policy using either the AWS Management Console or the Amazon S3 API:

- Grant the CloudFront origin access identity the applicable permissions on the bucket.

To specify an origin access identity, use the value of **Amazon S3 Canonical User ID on the Origin Access Identity** page in the CloudFront console. If you're using the CloudFront API, use the value of the `S3CanonicalUserId` element that was returned when you created the origin access identity.

- Deny access to anyone that you don't want to have access using Amazon S3 URLs.

For more information, go to [Using Bucket Policies and User Policies](#) in the *Amazon Simple Storage Service Developer Guide*.

For an example, see "Granting Permission to an Amazon CloudFront Origin Identify" in the topic [Bucket Policy Examples](#), also in the *Amazon Simple Storage Service Developer Guide*.

Updating Amazon S3 ACLs

Using either the AWS Management Console or the Amazon S3 API, change the Amazon S3 ACL:

- Grant the CloudFront origin access identity the applicable permissions on each object that the CloudFront distribution serves.

To specify an origin access identity, use the value of **Amazon S3 Canonical User ID** on the **Origin Access Identity** page in the CloudFront console. If you're using the CloudFront API, use the value of the `S3CanonicalUserId` element that was returned when you created the origin access identity.

- Deny access to anyone that you don't want to have access using Amazon S3 URLs.

If another AWS account uploads objects to your bucket, that account is the owner of the objects. By default, the account that owns objects in a bucket is the only account that can grant permissions to those objects. However, the AWS account that owns the objects can make you an owner, too, which allows you to change permissions on the objects.

For more information, go to [Managing Access with ACLs](#) in the *Amazon Simple Storage Service Developer Guide*.

You can also change the ACLs using code and one of the AWS SDKs. For an example, see the downloadable sample code in [Create a URL Signature Using C# and the .NET Framework \(p. 270\)](#).

Using an Origin Access Identity in Amazon S3 Regions that Support Only Signature Version 4 Authentication

Newer Amazon S3 regions require that you use signature version 4 for authenticated requests. (For the versions of signature supported in each Amazon S3 region, see [Amazon Simple Storage Service \(S3\)](#) in the topic [Regions and Endpoints](#) in the *Amazon Web Services General Reference*.) However, when you create an origin access identity and add it to a CloudFront distribution, CloudFront typically uses signature version 2 for authentication when it requests objects in your Amazon S3 bucket. If you're using an origin access identity and if your bucket is in one of the regions that requires signature version 4 for authentication, note the following:

- `DELETE`, `GET`, `HEAD`, `OPTIONS`, and `PATCH` requests are supported without qualifications.
- If you want to submit `PUT` requests to CloudFront to upload objects to your Amazon S3 bucket, you must add an `x-amz-content-sha256` header to the request, and the header value must contain a SHA256 hash of the body of the request. For more information, see the documentation about the `x-amz-content-sha256` header on the [Common Request Headers](#) page in the *Amazon Simple Storage Service API Reference*.
- `POST` requests are not supported.

Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies (Trusted Signers)

Topics

- [Creating CloudFront Key Pairs for Your Trusted Signers \(p. 217\)](#)
- [Reformatting the CloudFront Private Key \(.NET and Java Only\) \(p. 218\)](#)
- [Adding Trusted Signers to Your Distribution \(p. 219\)](#)
- [Verifying that Trusted Signers Are Active \(Optional\) \(p. 221\)](#)
- [Rotating CloudFront Key Pairs \(p. 221\)](#)

To create signed URLs or signed cookies, you need at least one AWS account that has an active CloudFront key pair. This account is known as a trusted signer. The trusted signer has two purposes:

- As soon as you add the AWS account ID for your trusted signer to your distribution, CloudFront starts to require that users use signed URLs or signed cookies to access your objects.
- When you create signed URLs or signed cookies, you use the private key from the trusted signer's key pair to sign a portion of the URL or the cookie. When someone requests a restricted object, CloudFront compares the signed portion of the URL or cookie with the unsigned portion to verify that the URL or cookie hasn't been tampered with. CloudFront also verifies that the URL or cookie is valid, meaning, for example, that the expiration date and time hasn't passed.

When you specify trusted signers, you also indirectly specify the objects that require signed URLs or signed cookies:

- **Web distributions** – You add trusted signers to cache behaviors. If your distribution has only one cache behavior, users must use signed URLs or signed cookies to access any object associated with the distribution. If you create multiple cache behaviors and add trusted signers to some cache behaviors and not to others, you can require that users use signed URLs or signed cookies to access some objects and not others.
- **RTMP distributions (signed URLs only)** – You add trusted signers to a distribution. After you add trusted signers to an RTMP distribution, users must use signed URLs to access any object associated with the distribution.

Note

To specify trusted signers for a distribution, you must use the CloudFront console or CloudFront API version 2009-09-09 or later.

To specify the accounts that are allowed to create signed URLs or signed cookies and to add the accounts to your CloudFront distribution, perform the following tasks:

1. Decide which AWS accounts you want to use as trusted signers. Most CloudFront customers use the account that they used to create the distribution.
2. For each of the accounts that you selected in Step 1, create a CloudFront key pair. For more information, see [Creating CloudFront Key Pairs for Your Trusted Signers \(p. 217\)](#).
3. If you're using .NET or Java to create signed URLs or signed cookies, reformat the CloudFront private key. For more information, see [Reformatting the CloudFront Private Key \(.NET and Java Only\) \(p. 218\)](#).
4. In the distribution for which you're creating signed URLs or signed cookies, specify the AWS account IDs of your trusted signers. For more information, see [Adding Trusted Signers to Your Distribution \(p. 219\)](#).

5. (Optional) Verify that CloudFront recognizes that your trusted signers have active CloudFront key pairs. For more information, see [Verifying that Trusted Signers Are Active \(Optional\) \(p. 221\)](#).

Creating CloudFront Key Pairs for Your Trusted Signers

Each of the AWS accounts that you use to create CloudFront signed URLs or signed cookies—your trusted signers—must have its own CloudFront key pair, and the key pair must be active. Note that you can't substitute an Amazon EC2 key pair for a CloudFront key pair. When you create a CloudFront signed URL or signed cookie, you include the key pair ID for the trusted signer's key pair in the URL. Amazon EC2 does not make key pair IDs available.

To help secure your applications, we recommend that you change CloudFront key pairs every 90 days or more often. For more information, see [Rotating CloudFront Key Pairs \(p. 221\)](#).

You can create a key pair in the following ways:

- Create a key pair in the AWS Management Console and download the private key. See the procedure [To create CloudFront key pairs in the AWS Management Console \(p. 217\)](#).
- Create an RSA key pair by using an application such as OpenSSL, and upload the public key to the AWS Management Console. See the procedure [To create an RSA key pair and upload the public key in the AWS Management Console \(p. 217\)](#).

To create CloudFront key pairs in the AWS Management Console

1. Sign in to the AWS Management Console using the root credentials for an AWS account.

Important

IAM users can't create CloudFront key pairs. You must log in using root credentials to create key pairs.

2. On the *account-name* menu, click **Security Credentials**.
3. Expand **CloudFront Key Pairs**.
4. Confirm that you have no more than one active key pair. You can't create a key pair if you already have two active key pairs.
5. Click **Create New Key Pair**.
6. In the **Create Key Pair** dialog box, click **Download Private Key File**.
7. In the **Opening <filename>** dialog box, accept the default value of **Save File**, and click **OK** to download and save the private key for your CloudFront key pair.

Important

Save the private key for your CloudFront key pair in a secure location, and set permissions on the file so that only the desired administrator users can read it. If someone gets your private key, they can generate valid signed URLs and signed cookies and download your content. You cannot get the private key again, so if you lose or delete it, you must create a new CloudFront key pair.

8. Record the key pair ID for your key pair. (In the AWS Management Console, this is called the access key ID.) You'll use it when you create signed URLs or signed cookies.

To create an RSA key pair and upload the public key in the AWS Management Console

1. Use OpenSSL or another tool to create a key pair.

For example, if you're using OpenSSL, you can use the following command to generate a key pair with a length of 4096 bits and save it in the file `private_key.pem`:

```
$ openssl genrsa -out private_key.pem 4096
```

The resulting file contains both the public and the private key. To extract the public key from that file, run the following command:

```
$ openssl rsa -pubout -in private_key.pem -out public_key.pem
```

The public key is the file that you upload later in this procedure

Note the following requirements for the key:

- The key pair must be an SSH-2 RSA key pair.
- The key pair must be in base64 encoded PEM format.
- The supported key lengths are 1024, 2048, and 4096 bits.

2. Sign in to the AWS Management Console using the root credentials for an AWS account.

Important

IAM users can't create CloudFront key pairs. You must log in using root credentials to create key pairs.

3. On the `account-name` menu, click **Security Credentials**.
4. Expand **CloudFront Key Pairs**.
5. Confirm that you have no more than one active key pair. You can't upload your own key pair if you already have two active key pairs.
6. Click **Upload Your Own Key Pair**.
7. In the **Upload Your Own Key Pair** dialog box, click **Choose File** and choose the public key file that you created in step 1.
8. Click **Upload**.

The **Upload Key Pair** dialog box clears, and the new key pair appears at the top of the list of CloudFront key pairs.

9. Record the key pair ID for your key pair. (In the AWS Management Console, this is called the access key ID.) You'll use it when you create signed URLs or signed cookies.

Reformatting the CloudFront Private Key (.NET and Java Only)

If you're using .NET or Java to create signed URLs or signed cookies, you cannot use the private key from your key pair in the default .pem format to create the signature:

- **.NET framework** – Convert the private key to the XML format that the .NET framework uses. Several tools are available.
- **Java** – Convert the private key to DER format. To do this, you can use OpenSSL:

```
$ openssl pkcs8 -topk8 -nocrypt -in origin.pem -inform PEM -out new.der -  
outform DER
```

To ensure that the encoder works correctly, add the jar for the Bouncy Castle Java cryptography APIs to your project and then add the Bouncy Castle provider.

Adding Trusted Signers to Your Distribution

Trusted signers are the AWS accounts that can create signed URLs and signed cookies for a distribution. By default, no account, not even the account that created the distribution, is allowed to create signed URLs or signed cookies. To specify the AWS accounts that you want to use as trusted signers, add the accounts to your distribution:

- **Web distributions** – Trusted signers are associated with cache behaviors. This allows you to require signed URLs or signed cookies for some objects and not for others in the same distribution. Trusted signers can only create signed URLs or cookies for objects that are associated with the corresponding cache behaviors. For example, if you have one trusted signer for one cache behavior and a different trusted signer for a different cache behavior, neither trusted signer can create signed URLs or cookies for objects that are associated with the other cache behavior.
- **RTMP distributions (signed URLs only)** – Trusted signers are associated with the distribution. After you add trusted signers to an RTMP distribution, users must use signed URLs or signed cookies to access any of the objects associated with the distribution.

Important

Define path patterns and their sequence carefully so you don't either give users unintended access to your content or prevent them from accessing content that you want to be available to everyone. For example, suppose a request matches the path pattern for two cache behaviors. The first cache behavior does not require signed URLs or signed cookies and the second cache behavior does. Users will be able to access the objects without using signed URLs or signed cookies because CloudFront processes the cache behavior that is associated with the first match.

For more information about path patterns, see [Path Pattern \(p. 71\)](#).

Important

If you're updating a distribution that you're already using to distribute content, add trusted signers only when you're ready to start generating signed URLs or signed cookies for your objects, or CloudFront will reject the requests:

- **Web distributions** – After you add trusted signers to a cache behavior for a web distribution, users must use signed URLs or signed cookies to access the objects that are associated with the cache behavior.
- **RTMP distributions (signed URLs only)** – After you add trusted signers to an RTMP distribution, users must use signed URLs to access any of the objects associated with the distribution.

The maximum number of trusted signers depends on the type of distribution:

- **Web distributions** – A maximum of five for each cache behavior
- **RTMP distributions** – A maximum of five for the distribution

You can add trusted signers to your distribution using either the CloudFront console or the CloudFront API. See the applicable topic:

- [Adding Trusted Signers to Your Distribution Using the CloudFront Console \(p. 220\)](#)
- [Adding Trusted Signers to Your Distribution Using the CloudFront API \(p. 220\)](#)

Adding Trusted Signers to Your Distribution Using the CloudFront Console

To add trusted signers to your distribution using the CloudFront console

1. If you want to use only the AWS account that created the distribution as a trusted signer, skip to Step 2.

If you want to use other AWS accounts, get the AWS account ID for each account:

- a. Sign in to the AWS Management Console at <https://console.aws.amazon.com/console/home> using an account that you want to use as a trusted signer.
 - b. In the upper-right corner of the console, click the name associated with the account, and click **My Account**.
 - c. Under **Account Settings**, make note of the account ID.
 - d. Sign out of the AWS Management Console.
 - e. Repeat steps a through d for the other accounts that you want to use as trusted signers.
2. Open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>, and sign in using the account that you used to create the distribution that you want to add trusted signers to.
 3. Click the distribution ID.
 4. Change to edit mode:
 - **Web distributions** – Click the **Behaviors** tab, click the behavior that you want to edit, and click **Edit**.
 - **RTMP distributions** – Click **Edit**.
 5. For **Restrict Viewer Access (Use Signed URLs or Signed Cookies)**, click **Yes**.
 6. For **Trusted Signers**, check the applicable check boxes:
 - **Self** – Check this check box if you want to use the current account (the account that you used to create the distribution).
 - **Specify Accounts** – Check this check box if you want to use other AWS accounts.
 7. If you checked the **Specify Accounts** check box, enter AWS account IDs in the **AWS Account Number** field. These are the account IDs that you got in the first step of this procedure. Enter one account ID per line.
 8. Click **Yes, Edit**.
 9. If you're adding trusted signers to a web distribution and you have more than one cache behavior, repeat steps 4 through 8 as applicable.

Adding Trusted Signers to Your Distribution Using the CloudFront API

You can use the CloudFront API to add the AWS account IDs for trusted signers to an existing distribution or to create a new distribution that includes trusted signers. In either case, specify the applicable values in the `TrustedSigners` element. For web distributions, add the `TrustedSigners` element to one or more cache behaviors. For RTMP distributions, add the `TrustedSigners` element to the distribution.

See the applicable topic in the *Amazon CloudFront API Reference*:

- **Create a new web distribution** – [POST Distribution](#)
- **Update an existing web distribution** – [PUT Distribution Config](#)
- **Create a new RTMP distribution** – [POST Streaming Distribution](#)

- **Update an existing RTMP distribution – [PUT Streaming Distribution Config](#)**

Verifying that Trusted Signers Are Active (Optional)

After you add trusted signers to your distribution, you might want to verify that the signers are active. For a trusted signer to be active, the following must be true:

- The AWS account must have at least one active key pair. If you're rotating key pairs, the account will temporarily have two active key pairs, the old key pair and the new one.
- CloudFront must be aware of the active key pair. After you create a key pair, there can be a short period of time before CloudFront is aware that the key pair exists.

Note

To display a list of active trusted signers for a distribution, you currently must use the CloudFront API. A list of active trusted signers is not available in the CloudFront console.

Verifying that Trusted Signers Are Active Using the CloudFront API

To determine which trusted signers have active key pairs (are active trusted signers), you get the distribution and review the values in the `ActiveTrustedSigners` element. This element lists the AWS account ID of each account that the distribution identifies as a trusted signer. If the trusted signer has one or more active CloudFront key pairs, the `ActiveTrustedSigners` element also lists the key pair IDs. For more information, see the applicable topic in the *Amazon CloudFront API Reference*:

- **Web distributions – [GET Distribution](#)**
- **RTMP distributions – [GET Streaming Distribution](#)**

Rotating CloudFront Key Pairs

AWS recommends that you rotate (change) your active CloudFront key pairs every 90 days. To rotate CloudFront key pairs that you're using to create signed URLs or signed cookies without invalidating URLs or cookies that haven't expired yet, perform the following tasks:

1. Create a new key pair for each of the accounts that you're using to create signed URLs. For more information, see [Creating CloudFront Key Pairs for Your Trusted Signers \(p. 217\)](#).
2. Verify that CloudFront is aware of the new key pairs. For more information, see [Verifying that Trusted Signers Are Active \(Optional\) \(p. 221\)](#).
3. Update your application to create signatures using the private keys from the new key pairs.
4. Confirm that URLs or cookies that you're signing using the new private keys are working.
5. Wait until the expiration date has passed in URLs or cookies that were signed using the old CloudFront key pairs.
6. Change the old CloudFront key pairs to **Inactive**:
 - a. Sign in to the AWS Management Console using the root credentials for an AWS account for which you want to make key pairs inactive.
 - b. On the *account-name* menu, click **Security Credentials**.
 - c. Expand **CloudFront Key Pairs**.
 - d. For the applicable key pairs, click **Make Inactive**.
 - e. Repeat steps a through d for each of the AWS accounts for which you want to make key pairs inactive.

7. Reconfirm that URLs or cookies that you're signing using the new private keys are working.
8. Delete the old CloudFront key pairs:
 - a. Go to the [Your Security Credentials](#) page.
 - b. Expand **CloudFront Key Pairs**.
 - c. For the applicable key pairs, click **Delete**.
9. Delete the old private keys from the location where you stored them.

Choosing Between Signed URLs and Signed Cookies

CloudFront signed URLs and signed cookies provide the same basic functionality: they allow you to control who can access your content. If you want to serve private content through CloudFront and you're trying to decide whether to use signed URLs or signed cookies, consider the following.

Use signed URLs in the following cases:

- You want to use an RTMP distribution. Signed cookies aren't supported for RTMP distributions.
- You want to restrict access to individual files, for example, an installation download for your application.
- Your users are using a client (for example, a custom HTTP client) that doesn't support cookies.

Use signed cookies in the following cases:

- You want to provide access to multiple restricted files, for example, all of the files for a video in HLS format or all of the files in the subscribers' area of a website.
- You don't want to change your current URLs.

If you are not currently using signed URLs and if your URLs contain any of the following query string parameters, you cannot use either signed URLs or signed cookies:

- `Expires`
- `Policy`
- `Signature`
- `Key-Pair-Id`

CloudFront assumes that URLs that contain any of those query string parameters are signed URLs and therefore won't look at signed cookies.

Using Both Signed URLs and Signed Cookies

If you use both signed URLs and signed cookies to control access to the same objects and a viewer uses a signed URL to request an object, CloudFront determines whether to return the object to the viewer based only on the signed URL.

Using Signed URLs

Topics

- [Choosing Between Canned and Custom Policies for Signed URLs \(p. 224\)](#)
- [How Signed URLs Work \(p. 225\)](#)
- [Choosing How Long Signed URLs Are Valid \(p. 225\)](#)
- [When Does CloudFront Check the Expiration Date and Time in a Signed URL? \(p. 226\)](#)
- [Sample Code and Third-Party Tools \(p. 226\)](#)
- [Creating a Signed URL Using a Canned Policy \(p. 227\)](#)
- [Creating a Signed URL Using a Custom Policy \(p. 234\)](#)

A signed URL includes additional information, for example, an expiration date and time, that gives you more control over access to your content. This additional information appears in a policy statement, which is based on either a canned policy or a custom policy. The differences between canned and custom policies are explained in the next two sections.

Note

You can create some signed URLs using canned policies and create some signed URLs using custom policies for the same distribution.

Choosing Between Canned and Custom Policies for Signed URLs

When you create a signed URL, you write a policy statement in JSON format that specifies the restrictions on the signed URL, for example, how long the URL is valid. You can use either a canned policy or a custom policy. Here's how canned and custom policies compare:

| Description | Canned Policy | Custom Policy |
|--|---------------|----------------|
| You can reuse the policy statement for multiple objects. To reuse the policy statement, you must use wildcard characters in the <code>Resource</code> object. For more information, see Values that You Specify in the Policy Statement for a Signed URL That Uses a Custom Policy (p. 237) .) | No | Yes |
| You can specify the date and time that users can begin to access your content. | No | Yes (optional) |
| You can specify the date and time that users can no longer access your content. | Yes | Yes |
| You can specify the IP address or range of IP addresses of the users who can access your content. | No | Yes (optional) |
| The signed URL includes a base64-encoded version of the policy, which results in a longer URL. | No | Yes |

For information about creating signed URLs using a canned policy, see [Creating a Signed URL Using a Canned Policy \(p. 227\)](#).

For information about creating signed URLs using a custom policy, see [Creating a Signed URL Using a Custom Policy \(p. 234\)](#).

How Signed URLs Work

Here's an overview of how you configure CloudFront and Amazon S3 for signed URLs and how CloudFront responds when a user uses a signed URL to request an object.

1. In your CloudFront distribution, specify one or more trusted signers, which are the AWS accounts that you want to have permission to create signed URLs.

For more information, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#).
2. You develop your application to determine whether a user should have access to your content and to create signed URLs for the objects or parts of your application that you want to restrict access to. For more information, see the applicable topic:
 - [Creating a Signed URL Using a Canned Policy \(p. 227\)](#)
 - [Creating a Signed URL Using a Custom Policy \(p. 234\)](#)
3. A user requests an object for which you want to require signed URLs.
4. Your application verifies that the user is entitled to access the object: they've signed in, they've paid for access to the content, or they've met some other requirement for access.
5. Your application creates and returns a signed URL to the user.
6. The signed URL allows the user to download or stream the content.

This step is automatic; the user usually doesn't have to do anything additional to access the content. For example, if a user is accessing your content in a web browser, your application returns the signed URL to the browser. The browser immediately uses the signed URL to access the object in the CloudFront edge cache without any intervention from the user.

7. CloudFront uses the public key to validate the signature and confirm that the URL hasn't been tampered with. If the signature is invalid, the request is rejected.

If the signature is valid, CloudFront looks at the policy statement in the URL (or constructs one if you're using a canned policy) to confirm that the request is still valid. For example, if you specified a beginning and ending date and time for the URL, CloudFront confirms that the user is trying to access your content during the time period that you want to allow access.

If the request meets the requirements in the policy statement, CloudFront performs the standard operations: determines whether the object is already in the edge cache, forwards the request to the origin if necessary, and returns the object to the user.

Choosing How Long Signed URLs Are Valid

You can distribute private content using a signed URL that is valid for only a short time—possibly for as little as a few minutes. Signed URLs that are valid for such a short period are good for distributing content on-the-fly to a user for a limited purpose, such as distributing movie rentals or music downloads to customers on demand. If your signed URLs will be valid for just a short period, you'll probably want to generate them automatically using an application that you develop. When the user starts to download an object or starts to play a media file, CloudFront compares the expiration time in the URL with the current time to determine whether the URL is still valid.

You can also distribute private content using a signed URL that is valid for a longer time, possibly for years. Signed URLs that are valid for a longer period are useful for distributing private content to known users, such as distributing a business plan to investors or distributing training materials to employees. You can develop an application to generate these longer-term signed URLs for you, or you can use one of the third-party GUI tools listed in [Tools and Code Examples for Configuring Private Content \(p. 366\)](#).

When Does CloudFront Check the Expiration Date and Time in a Signed URL?

When CloudFront checks the expiration date and time in a signed URL to determine whether the URL is still valid depends on whether the URL is for a web distribution or an RTMP distribution:

- **Web distributions** – CloudFront checks the expiration date and time in a signed URL at the time of the HTTP request. If a client begins to download a large object immediately before the expiration time, the download should complete even if the expiration time passes during the download. If the TCP connection drops and the client tries to restart the download after the expiration time passes, the download will fail.

If a client uses Range GETs to get an object in smaller pieces, any GET request that occurs after the expiration time passes will fail. For more information about Range GETs, see [How CloudFront Processes Partial Requests for an Object \(Range GETs\) \(p. 176\)](#).

- **RTMP distributions** – CloudFront checks the expiration time in a signed URL at the start of a play event. If a client starts to play a media file before the expiration time passes, CloudFront allows the entire media file to play. However, depending on the media player, pausing and restarting might trigger another play event. Skipping to another position in the media file will trigger another play event. If the subsequent play event occurs after the expiration time passes, CloudFront won't serve the media file.

Sample Code and Third-Party Tools

For sample code that creates the hashed and signed part of signed URLs, see the following topics:

- [Create a URL Signature Using Perl \(p. 260\)](#)
- [Create a URL Signature Using PHP \(p. 268\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 270\)](#)
- [Create a URL Signature Using Java \(p. 276\)](#)

Additional sample code for creating signed URLs is available on the [Amazon CloudFront Sample Code & Libraries](#) page.

For information about third-party tools that support private content, including creating signed URLs, see [Tools and Code Examples for Configuring Private Content \(p. 366\)](#).

Creating a Signed URL Using a Canned Policy

To create a signed URL using a canned policy, perform the following procedure.

To create a signed URL using a canned policy

1. If you're using .NET or Java to create signed URLs, and if you haven't reformatted the private key for your key pair from the default .pem format to a format compatible with .NET or with Java, do so now. For more information, see [Reformatting the CloudFront Private Key \(.NET and Java Only\) \(p. 218\)](#).
2. Concatenate the following values in the specified order, and remove the whitespace (including tabs and newline characters) between the parts. You might have to include escape characters in the string in application code. All values have a type of String. Each part is keyed by number (1) to the two examples that follow.

1 Base URL for the object

The base URL is the CloudFront URL that you would use to access the object if you were not using signed URLs, including your own query string parameters, if any. For more information about the format of URLs for web distributions, see [Format of URLs for Objects \(p. 143\)](#).

The following examples show values that you specify for web distributions.

- The following CloudFront URL is for an object in a web distribution (using the CloudFront domain name). Note that `image.jpg` is in an `images` directory. The path to the object in the URL must match the path to the object on your HTTP server or in your Amazon S3 bucket.

`http://d111111abcdef8.cloudfront.net/images/image.jpg`

- The following CloudFront URL includes a query string:

`http://d111111abcdef8.cloudfront.net/images/image.jpg?size=large`

- The following CloudFront URLs are for objects in a web distribution. Both use an alternate domain name; the second one includes a query string:

`http://www.example.com/images/image.jpg`

`http://www.example.com/images/image.jpg?color=red`

- The following CloudFront URL is for an objects in a web distribution that uses an alternate domain name and the HTTPS protocol:

`https://www.example.com/images/image.jpg`

For RTMP distributions, the following examples are for objects in two different video formats, MP4 and FLV:

- **MP4** – `mp4:sydney-vacation.mp4`
- **FLV** – `sydney-vacation`
- **FLV** – `sydney-vacation.flv`

Note

For .flv files, whether you include the `.flv` filename extension depends on your player. To serve MP3 audio files or H.264/MPEG-4 video files, you might need to prefix the file name with `mp3:` or `mp4::`. Some media players can be configured to add the prefix automatically. The media player might also require you to specify the file name without the file extension (for example, `sydney-vacation` instead of `sydney-vacation.mp4`).

2 ?

The `?` indicates that query string parameters follow the base URL. Include the `?` even if you don't have any query string parameters of your own.

3 Your query string parameters, if any&

This value is optional. If you want to add your own query string parameters, for example:

```
color=red&size=medium
```

then add the parameters after the `?` (see **2**) and before the `Expires` parameter. In certain rare circumstances, you might need to put your query string parameters after `Key-Pair-Id`.

Important

Your parameters cannot be named `Expires`, `Signature`, or `Key-Pair-Id`.

If you add your own parameters, append an `&` after each one, including the last one.

4 Expires=date and time in Unix time format (in seconds) and Coordinated Universal Time (UTC)

Specify the expiration date and time in Unix time format and Coordinated Universal Time (UTC). For example, January 1, 2013 10:00 am UTC converts to 1357034400 in Unix time format.

For information about UTC, see *RFC 3339, Date and Time on the Internet: Timestamps*, <http://tools.ietf.org/html/rfc3339>.

Specify the date and time, in Unix time format (in seconds) and Coordinated Universal Time (UTC), that you want the URL to stop allowing access to the object. For example, January 1, 2013 10:00 am UTC converts to 1357034400 in Unix time format.

5 &Signature=hashed and signed version of the policy statement

A hashed, signed, and base64-encoded version of the JSON policy statement. For more information, see [Creating a Signature for a Signed URL That Uses a Canned Policy \(p. 229\)](#).

6 &Key-Pair-Id=active CloudFront key pair ID for the key pair that you're using to generate the signature

The ID for an active CloudFront key pair, for example, APKA9ONS7QCOWEXAMPLE. The CloudFront key pair ID tells CloudFront which public key to use to validate the signed URL. CloudFront compares the information in the signature with the information in the policy statement to verify that the URL has not been tampered with.

The key pair ID that you include in CloudFront signed URLs must be the ID of an active key pair for one of your trusted signers:

- **Web distributions** – The key pair must be associated with an AWS account that is one of the trusted signers for the applicable cache behavior.
- **RTMP distributions** – The key pair must be associated with an AWS account that is one of the trusted signers for the distribution.

For more information, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#).

If you make a key pair inactive while rotating CloudFront key pairs, and if you're generating signed URLs programmatically, you must update your application to use a new active key pair for one of your trusted signers. If you're generating signed URLs manually, you must create new signed URLs. For more information about rotating key pairs, see [Rotating CloudFront Key Pairs \(p. 221\)](#).

Example signed URL for a web distribution:

① `http://d111111abcdef8.cloudfront.net/image.jpg` ②
? ③ `color=red&size=medium&` ④ `Expires=1357034400`
⑤ `&Signature=nitfHRCrtziwO2HwPfWw-yYDhUF5EwRunQA-`
`j19DzZrvDh6hQ731Dx--ar3UocvvRQVw6EkC-GdpGQyyOSKQim-`
`TxAnW7d8F5Kkai9Hvx0FIu-5jcQb0UEmatEXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkylL6f3fVYNGQI6`
⑥ `&Key-Pair-Id=APKA90NS7QCOWEXAMPLE`

Example signed URL for an RTMP distribution:

① `videos/mediafile.flv` ② ? ③ `color=red&size=medium&` ④
④ `Expires=1357034400` ⑤ `&Signature=nitfHRCrtziwO2HwPfWw-yYDhUF5EwRunQA-`
`j19DzZrvDh6hQ731Dx--ar3UocvvRQVw6EkC-GdpGQyyOSKQim-`
`TxAnW7d8F5Kkai9Hvx0FIu-5jcQb0UEmatEXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkylL6f3fVYNGQI6`
⑥ `&Key-Pair-Id=APKA90NS7QCOWEXAMPLE`

Creating a Signature for a Signed URL That Uses a Canned Policy

To create the signature for a signed URL that uses a canned policy, you perform the following procedures:

1. Create a policy statement. See [Creating a Policy Statement for a Signed URL That Uses a Canned Policy \(p. 229\)](#).
2. Sign the policy statement to create a signature. See [Creating a Signature for a Signed URL That Uses a Canned Policy \(p. 231\)](#).

Creating a Policy Statement for a Signed URL That Uses a Canned Policy

When you create a signed URL using a canned policy, the `Signature` parameter is a hashed and signed version of a policy statement. For signed URLs that use a canned policy, you don't include the policy statement in the URL, as you do for signed URLs that use a custom policy. To create the policy statement, perform the following procedure.

To create the policy statement for a signed URL that uses a canned policy

1. Construct the policy statement using the following JSON format and using UTF-8 character encoding. Include all punctuation and other literal values exactly as specified. For information about the `Resource` and `DateLessThan` parameters, see [Values that You Specify in the Policy Statement for a Signed URL That Uses a Canned Policy \(p. 230\)](#).

```
{  
    "Statement": [  
        {  
            "Resource": "base URL or stream name",  
            "Condition": {  
                "DateLessThan": {  
                    "AWS:EpochTime": ending date and time in Unix time format and UTC  
                }  
            }  
        }  
    ]  
}
```

2. Remove all whitespace (including tabs and newline characters) from the policy statement. You might have to include escape characters in the string in application code.

Values that You Specify in the Policy Statement for a Signed URL That Uses a Canned Policy

When you create a policy statement for a canned policy, you specify the following values.

Resource

The value that you specify depends on whether you're creating the signed URL for a web distribution or an RTMP distribution.

Note

You can specify only one value for `Resource`.

Web distributions

The base URL including your query strings, if any, but excluding the CloudFront `Expires`, `Signature`, and `Key-Pair-Id` parameters, for example:

```
http://d111111abcdef8.cloudfront.net/images/horizon.jpg?  
size=large&license=yes
```

Note the following:

- **Protocol** – The value must begin with `http://` or `https://`.
- **Query string parameters** – If you have no query string parameters, omit the question mark.
- **Alternate domain names** – If you specify an alternate domain name (CNAME) in the URL, you must specify the alternate domain name when referencing the object in your web page or application. Do not specify the Amazon S3 URL for the object.

RTMP distributions

Include only the stream name. For example, if the full URL for a streaming video is:

```
rtmp://s5c39gqb8ow64r.cloudfront.net/videos/cfx/st/mp3_name.mp3
```

then use the following value for `Resource`:

```
videos/mp3_name
```

Do not include a prefix such as `mp3:` or `mp4:`. Also, depending on the player you're using, you might have to omit the file extension from the value of `Resource`. For example, you might need to use `sydney-vacation` instead of `sydney-vacation.flv`.

DateLessThan

The expiration date and time for the URL in Unix time format (in seconds) and Coordinated Universal Time (UTC). For example, January 1, 2013 10:00 am UTC converts to 1357034400 in Unix time format.

This value must match the value of the `Expires` query string parameter in the signed URL. Do not enclose the value in quotation marks.

For more information, see [When Does CloudFront Check the Expiration Date and Time in a Signed URL? \(p. 226\)](#).

Example Policy Statement for a Signed URL That Uses a Canned Policy

When you use the following example policy statement in a signed URL, a user can access the object `http://d111111abcdef8.cloudfront.net/horizon.jpg` until January 1, 2013 10:00 am UTC:

```
{
```

```

    "Statement": [
        {
            "Resource": "http://d111111abcdef8.cloudfront.net/horizon.jpg?
size=large&license=yes",
            "Condition": {
                "DateLessThan": {
                    "AWS:EpochTime": 1357034400
                }
            }
        }
    ]
}

```

Creating a Signature for a Signed URL That Uses a Canned Policy

To create the value for the `Signature` parameter in a signed URL, you hash and sign the policy statement that you created in [Creating a Policy Statement for a Signed URL That Uses a Canned Policy \(p. 229\)](#). There are two versions of this procedure. Perform the applicable procedure:

- [Option 1: To create a signature for a web distribution or for an RTMP distribution \(without Adobe Flash Player\) by using a canned policy \(p. 231\)](#)
- [Option 2: To create a signature for an RTMP distribution by using a canned policy \(Adobe Flash Player\) \(p. 232\)](#)

For additional information and examples of how to hash, sign, and encode the policy statement, see:

- [Using a Linux Command and OpenSSL for Base64-Encoding and Encryption \(p. 259\)](#)
- [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)

Option 1: To create a signature for a web distribution or for an RTMP distribution (without Adobe Flash Player) by using a canned policy

1. Use the SHA-1 hash function and RSA to hash and sign the policy statement that you created in the procedure [To create the policy statement for a signed URL that uses a canned policy \(p. 229\)](#). Use the version of the policy statement that no longer includes whitespace.

For the private key that is required by the hash function, use the private key that is associated with the applicable active trusted signer.

Note

The method that you use to hash and sign the policy statement depends on your programming language and platform. For sample code, see [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#).

2. Remove whitespace (including tabs and newline characters) from the hashed and signed string.
3. Base64-encode the string using MIME base64 encoding. For more information, see [Section 6.8, Base64 Content-Transfer-Encoding in RFC 2045, MIME \(Multipurpose Internet Mail Extensions\) Part One: Format of Internet Message Bodies](#).
4. Replace characters that are invalid in a URL query string with characters that are valid. The following table lists invalid and valid characters.

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| + | - (hyphen) |
| = | _ (underscore) |

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| / | ~ (tilde) |

- Append the resulting value to your signed URL after `&Signature=`, and return to [To create a signed URL using a canned policy \(p. 227\)](#) to finish concatenating the parts of your signed URL.

Option 2: To create a signature for an RTMP distribution by using a canned policy (Adobe Flash Player)

- Use the SHA-1 hash function and RSA to hash and sign the policy statement that you created in the [To create the policy statement for a signed URL that uses a canned policy \(p. 229\)](#) procedure. Use the version of the policy statement that no longer includes whitespace.

For the private key that is required by the hash function, use the private key that is associated with the applicable active trusted signer.

Note

The method that you use to hash and sign the policy statement depends on your programming language and platform. For sample code, see [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#).

- Remove whitespace (including tabs and newline characters) from the hashed and signed string.

Continue on to Step 3 if you're using Adobe Flash Player and the stream name is passed in from a web page.

If you're using Adobe Flash Player and if the stream name is not passed in from a web page, skip the rest of this procedure. For example, if you wrote your own player that fetches stream names from within the Adobe Flash .swf file, skip the rest of this procedure.

- Base64-encode the string using MIME base64 encoding. For more information, see [Section 6.8, Base64 Content-Transfer-Encoding in RFC 2045, MIME \(Multipurpose Internet Mail Extensions\) Part One: Format of Internet Message Bodies](#).
- Replace characters that are invalid in a URL query string with characters that are valid. The following table lists invalid and valid characters.

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| + | - (hyphen) |
| = | _ (underscore) |
| / | ~ (tilde) |

- Some versions of Adobe Flash Player require that you URL-encode the characters ?, =, and &. For information about whether your version of Adobe Flash Player requires this character substitution, refer to the Adobe website.

If your version of Flash does not require URL-encoding those character, skip to Step 6.

If your version of Flash requires URL-encoding those characters, replace them as indicated in the following table. (You already replaced = in the previous step.)

| Replace these invalid characters | With this URL encoding |
|----------------------------------|------------------------|
| ? | %3F |

| Replace these invalid characters | With this URL encoding |
|----------------------------------|------------------------|
| & | %26 |

6. Append the resulting value to your signed URL after &Signature=, and return to [To create a signed URL using a canned policy \(p. 227\)](#) to finish concatenating the parts of your signed URL.

Creating a Signed URL Using a Custom Policy

Topics

- [Creating a Policy Statement for a Signed URL That Uses a Custom Policy \(p. 236\)](#)
- [Example Policy Statements for a Signed URL That Uses a Custom Policy \(p. 239\)](#)
- [Creating a Signature for a Signed URL That Uses a Custom Policy \(p. 241\)](#)

To create a signed URL using a custom policy, perform the following procedure.

To create a signed URL using a custom policy

1. If you're using .NET or Java to create signed URLs, and if you haven't reformatted the private key for your key pair from the default .pem format to a format compatible with .NET or with Java, do so now. For more information, see [Reformatting the CloudFront Private Key \(.NET and Java Only\) \(p. 218\)](#).
2. Concatenate the following values in the specified order, and remove the whitespace (including tabs and newline characters) between the parts. You might have to include escape characters in the string in application code. All values have a type of String. Each part is keyed by number (1) to the two examples that follow.

1 *Base URL for the object*

The base URL is the CloudFront URL that you would use to access the object if you were not using signed URLs, including your own query string parameters, if any. For more information about the format of URLs for web distributions, see [Format of URLs for Objects \(p. 143\)](#).

The following examples show values that you specify for web distributions.

- The following CloudFront URL is for an object in a web distribution (using the CloudFront domain name). Note that `image.jpg` is in an `images` directory. The path to the object in the URL must match the path to the object on your HTTP server or in your Amazon S3 bucket.

`http://d111111abcdef8.cloudfront.net/images/image.jpg`

- The following CloudFront URL includes a query string:

`http://d111111abcdef8.cloudfront.net/images/image.jpg?size=large`

- The following CloudFront URLs are for objects in a web distribution. Both use an alternate domain name; the second one includes a query string:

`http://www.example.com/images/image.jpg`

`http://www.example.com/images/image.jpg?color=red`

- The following CloudFront URL is for an objects in a web distribution that uses an alternate domain name and the HTTPS protocol:

`https://www.example.com/images/image.jpg`

For RTMP distributions, the following examples are for objects in two different video formats, MP4 and FLV:

- **MP4** – `mp4:sydney-vacation.mp4`
- **FLV** – `sydney-vacation`
- **FLV** – `sydney-vacation.flv`

Note

For .flv files, whether you include the .flv filename extension depends on your player. To serve MP3 audio files or H.264/MPEG-4 video files, you might need to prefix the file name with mp3: or mp4:. Some media players can be configured to add the prefix automatically. The media player might also require you to specify the file name without the file extension (for example, sydney-vacation instead of sydney-vacation.mp4).

2 ?

The ? indicates that query string parameters follow the base URL. Include the ? even if you don't have any query string parameters of your own.

3 Your query string parameters, if any&

This value is optional. If you want to add your own query string parameters, for example:

color=red&size=medium

then add them after the ? (see **2**) and before the Policy parameter. In certain rare circumstances, you might need to put your query string parameters after Key-Pair-Id.

Important

Your parameters cannot be named Policy, Signature, or Key-Pair-Id.

If you add your own parameters, append an & after each one, including the last one.

4 Policy=base64 encoded version of policy statement

Your policy statement in JSON format, with white space removed, then base64 encoded. For more information, see [Creating a Policy Statement for a Signed URL That Uses a Custom Policy \(p. 236\)](#).

The policy statement controls the access that a signed URL grants to a user: the URL of the object (for web distributions) or the stream name (for RTMP distributions), an expiration date and time, an optional date and time that the URL becomes valid, and an optional IP address or range of IP addresses that are allowed to access the object.

5 &signature=hashed and signed version of the policy statement

A hashed, signed, and base64-encoded version of the JSON policy statement. For more information, see [Creating a Signature for a Signed URL That Uses a Custom Policy \(p. 241\)](#).

6 &Key-Pair-Id=active CloudFront key pair Id for the key pair that you're using to sign the policy statement

The ID for an active CloudFront key pair, for example, APKA9ONS7QCOWEXAMPLE. The CloudFront key pair ID tells CloudFront which public key to use to validate the signed URL. CloudFront compares the information in the signature with the information in the policy statement to verify that the URL has not been tampered with.

The key pair ID that you include in CloudFront signed URLs must be the ID of an active key pair for one of your trusted signers:

- **Web distributions** – The key pair must be associated with an AWS account that is one of the trusted signers for the applicable cache behavior.
- **RTMP distributions** – The key pair must be associated with an AWS account that is one of the trusted signers for the distribution.

For more information, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signer\)](#).
Version 2016-09-29

If you make a key pair inactive while rotating CloudFront key pairs, and if you're generating signed URLs programmatically, you must update your application to use a new active key pair for one of your trusted signers. If you're generating signed URLs manually, you must create new signed URLs. For more information about rotating key pairs, see [Rotating CloudFront Key Pairs \(p. 221\)](#).

Example signed URL for a web distribution:

```

① http://d111111abcdef8.cloudfront.net/image.jpg ② ?
③ color=red&size=medium&
④ Policy=eyANCiAgICEXAMPLEW1lbnQiOibbeyANCiAgICAgICJSZXNvdXJjZSI6Imh0dHA
6Ly9kemJlc3FtN3VuMW0wLmNsB3VkJnJvbnQubmV0L2RlbW8ucGhwIiwgDQogICAgICAIQ
29uZG1oaw9uIjp7IAOKICAgICAgICAgIk1wQWRkcmVzcyI6eyJBV1M6U291cmNlSXAiOii
yMDcuMTcxLjE4MC4xMDEvMzIifSwNCiAgICAgICAgICJEYXR1R3J1YXRlc1RoYW4iOnsiQ
vdTOkVwb2NoVGltZSI6MTI5Njg2MDE3Nn0sDQogICAgICAgICAIRGF0ZUxlC3NUaGFuIjp
7IkFXUzpFcG9jaFRpbWUiOjEyOTY4NjAyMjZ9DQogICAgICB9IAOKICAgfV0gDQp9DQo
⑤ &Signature=nitfHRCrtziwO2HwPfw-yYDhUF5EwRunQA-j19DzZrvDh6hQ731Dx-
-ar3UocvvRQVw6EkC-GdpGQyyOSKQim-TxAnW7d8F5Kkai9HVx0FIu-5jcQb0UEmat
EXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkyl6f3fVYNGQI6 ⑥ &Key-Pair-
Id=APKA90ONS7QCOWEXAMPLE

```

Example signed URL for an RTMP distribution:

```

① videos/mediafile.flv ② ? ③ color=red&size=medium&
④ Policy=eyANCiAgICEXAMPLEW1lbnQiOibbeyANCiAgICAgICJSZXNvdXJjZSI6Imh0dHA
6Ly9kemJlc3FtN3VuMW0wLmNsB3VkJnJvbnQubmV0L2RlbW8ucGhwIiwgDQogICAgICAIQ
29uZG1oaw9uIjp7IAOKICAgICAgICAgIk1wQWRkcmVzcyI6eyJBV1M6U291cmNlSXAiOii
yMDcuMTcxLjE4MC4xMDEvMzIifSwNCiAgICAgICAgICJEYXR1R3J1YXRlc1RoYW4iOnsiQ
vdTOkVwb2NoVGltZSI6MTI5Njg2MDE3Nn0sDQogICAgICAgICAIRGF0ZUxlC3NUaGFuIjp
7IkFXUzpFcG9jaFRpbWUiOjEyOTY4NjAyMjZ9DQogICAgICB9IAOKICAgfV0gDQp9DQo
⑤ &Signature=nitfHRCrtziwO2HwPfw-yYDhUF5EwRunQA-j19DzZrvDh6hQ731Dx-
-ar3UocvvRQVw6EkC-GdpGQyyOSKQim-TxAnW7d8F5Kkai9HVx0FIu-5jcQb0UEmat
EXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkyl6f3fVYNGQI6 ⑥ &Key-Pair-
Id=APKA90ONS7QCOWEXAMPLE

```

Creating a Policy Statement for a Signed URL That Uses a Custom Policy

To create a policy statement for a custom policy, perform the following procedure. For several example policy statements that control access to objects in a variety of ways, see [Example Policy Statements for a Signed URL That Uses a Custom Policy \(p. 239\)](#).

To create the policy statement for a signed URL that uses a custom policy

1. Construct the policy statement using the following JSON format. For more information, see [Values that You Specify in the Policy Statement for a Signed URL That Uses a Custom Policy \(p. 237\)](#).

```
{
  "Statement": [
    {
      "Resource": "URL or stream name of the object",

```

```

        "Condition": {
            "DateLessThan": {"AWS:EpochTime": required ending date and time in Unix time format and UTC},
            "DateGreaterThanOrEqual": {"AWS:EpochTime": optional beginning date and time in Unix time format and UTC},
            "IpAddress": {"AWS:SourceIp": optional IP address}
        }
    }
}

```

Note the following:

- You can include only one statement.
 - Use UTF-8 character encoding.
 - Include all punctuation and parameter names exactly as specified. Abbreviations for parameter names are not accepted.
 - The order of the parameters in the Condition section doesn't matter.
 - For information about the values for Resource, DateLessThan, DateGreaterThanOrEqual, and IpAddress, see [Values that You Specify in the Policy Statement for a Signed URL That Uses a Custom Policy \(p. 237\)](#).
2. Remove all whitespace (including tabs and newline characters) from the policy statement. You might have to include escape characters in the string in application code.
 3. Base64-encode the policy statement using MIME base64 encoding. For more information, see [Section 6.8, Base64 Content-Transfer-Encoding in RFC 2045, MIME \(Multipurpose Internet Mail Extensions\) Part One: Format of Internet Message Bodies](#).
 4. Replace characters that are invalid in a URL query string with characters that are valid. The following table lists invalid and valid characters.

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| + | - (hyphen) |
| = | _ (underscore) |
| / | ~ (tilde) |

5. Append the resulting value to your signed URL after `Policy=`.
6. Create a signature for the signed URL by hashing, signing, and base64-encoding the policy statement. For more information, see [Creating a Signature for a Signed URL That Uses a Custom Policy \(p. 241\)](#).

Values that You Specify in the Policy Statement for a Signed URL That Uses a Custom Policy

When you create a policy statement for a custom policy, you specify the following values.

Resource

The value that you specifies depends on whether you're creating signed URLs for web or RTMP distributions.

Note

You can specify only one value for Resource.

Web distributions (optional but recommended)

The base URL including your query strings, if any, but excluding the CloudFront Policy, Signature, and Key-Pair-Id parameters, for example:

```
http://d111111abcdef8.cloudfront.net/images/horizon.jpg?  
size=large&license=yes
```

Important

If you omit the Resource parameter for a web distribution, users can access all of the objects associated with any distribution that is associated with the key pair that you use to create the signed URL.

Note the following:

- **Protocol** – The value must begin with `http://`, `https://`, or `*`.
- **Query string parameters** – If you have no query string parameters, omit the question mark.
- **Wildcard characters** – You can use the wildcard character that matches zero or more characters (*) or the wild-card character that matches exactly one character (?) anywhere in the string. For example, the value:

```
http://d111111abcdef8.cloudfront.net/*game_download.zip*
```

would include (for example) the following objects:

- `http://d111111abcdef8.cloudfront.net/game_download.zip`
- `http://d111111abcdef8.cloudfront.net/example_game_download.zip?`
`license=yes`
- `http://d111111abcdef8.cloudfront.net/test_game_download.zip?`
`license=temp`
- **Alternate domain names** – If you specify an alternate domain name (CNAME) in the URL, you must specify the alternate domain name when referencing the object in your web page or application. Do not specify the Amazon S3 URL for the object.

RTMP distributions

Include only the stream name. For example, if the full URL for a streaming video is:

```
rtmp://s5c39gqb8ow64r.cloudfront.net/videos/cfx/st/mp3_name.mp3
```

then use the following value for `Resource`:

```
videos/mp3_name
```

Do not include a prefix such as `mp3:` or `mp4:`. Also, depending on the player you're using, you might have to omit the file extension from the value of `Resource`. For example, you might need to use `sydney-vacation` instead of `sydney-vacation.flv`.

DateLessThan

The expiration date and time for the URL in Unix time format (in seconds) and Coordinated Universal Time (UTC). Do not enclose the value in quotation marks. For information about UTC, see [RFC 3339](#), *Date and Time on the Internet: Timestamps*, <http://tools.ietf.org/html/rfc3339>.

For example, January 1, 2013 10:00 am UTC converts to 1357034400 in Unix time format.

This is the only required parameter in the Condition section. CloudFront requires this value to prevent users from having permanent access to your private content.

For more information, see [When Does CloudFront Check the Expiration Date and Time in a Signed URL? \(p. 226\)](#)

DateGreaterThan (Optional)

An optional start date and time for the URL in Unix time format (in seconds) and Coordinated Universal Time (UTC). Users are not allowed to access the object before the specified date and time. Do not enclose the value in quotation marks.

IpAddress (Optional)

The IP address of the client making the GET request. Note the following:

- To allow any IP address to access the object, omit the `IpAddress` parameter.
- You can specify either one IP address or one IP address range. For example, you can't set the policy to allow access if the client's IP address is in one of two separate ranges.
- To allow access from a single IP address, you specify:

`"IPv4 IP address/32"`

- You must specify IP address ranges in standard IPv4 CIDR format (for example, `192.0.2.0/24`). For more information, see *RFC 4632, Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*, <http://tools.ietf.org/html/rfc4632>.

Important

IP addresses in IPv6 format, such as `2001:0db8:85a3:0000:0000:8a2e:0370:7334`, are not supported.

If you're using a custom policy that includes `IpAddress`, do not enable IPv6 for the distribution. If you want to restrict access to some content by IP address and support IPv6 requests for other content, you can create two distributions. For more information, see [Enable IPv6 \(p. 81\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

Example Policy Statements for a Signed URL That Uses a Custom Policy

The following example policy statements show how to control access to a specific object, all of the objects in a directory, or all of the objects associated with a key pair ID. The examples also show how to control access from an individual IP address or a range of IP addresses, and how to prevent users from using the signed URL after a specified date and time.

If you copy and paste any of these examples, remove any whitespace (including tabs and newline characters), replace the applicable values with your own values, and include a newline character after the closing brace `{}`.

For more information, see [Values that You Specify in the Policy Statement for a Signed URL That Uses a Custom Policy \(p. 237\)](#).

Topics

- [Example Policy Statement: Accessing One Object from a Range of IP Addresses \(p. 239\)](#)
- [Example Policy Statement: Accessing All Objects in a Directory from a Range of IP Addresses \(p. 240\)](#)
- [Example Policy Statement: Accessing All Objects Associated with a Key Pair ID from One IP Address \(p. 240\)](#)

Example Policy Statement: Accessing One Object from a Range of IP Addresses

The following example custom policy in a signed URL specifies that a user can access the object `http://d111111abcdef8.cloudfront.net/game_download.zip` from IP addresses in the range `192.0.2.0/24` until January 1, 2013 10:00 am UTC:

```
{
  "Statement": [
    {
      "Resource": "http://d11111abcdef8.cloudfront.net/game_download.zip",
      "Condition": {
        "IpAddress": {"AWS:SourceIp": "192.0.2.0/24"},
        "DateLessThan": {"AWS:EpochTime": 1357034400}
      }
    }
  ]
}
```

Example Policy Statement: Accessing All Objects in a Directory from a Range of IP Addresses

The following example custom policy allows you to create signed URLs for any object in the `training` directory, as indicated by the `*` wildcard character in the `Resource` parameter. Users can access the object from an IP address in the range `192.0.2.0/24` until January 1, 2013 10:00 am UTC:

```
{
  "Statement": [
    {
      "Resource": "http://d11111abcdef8.cloudfront.net/training/*",
      "Condition": {
        "IpAddress": {"AWS:SourceIp": "192.0.2.0/24"},
        "DateLessThan": {"AWS:EpochTime": 1357034400}
      }
    }
  ]
}
```

Each signed URL in which you use this policy includes a base URL that identifies a specific object, for example:

<http://d11111abcdef8.cloudfront.net/training/orientation.pdf>

Example Policy Statement: Accessing All Objects Associated with a Key Pair ID from One IP Address

The following sample custom policy allows you to create signed URLs for any object associated with any distribution, as indicated by the `*` wildcard character in the `Resource` parameter. The user must use the IP address `192.0.2.10/32`. (The value `192.0.2.10/32` in CIDR notation refers to a single IP address, `192.0.2.10`.) The objects are available only from January 1, 2013 10:00 am UTC until January 2, 2013 10:00 am UTC:

```
{
  "Statement": [
    {
      "Resource": "http://*",
      "Condition": {
        "IpAddress": {"AWS:SourceIp": "192.0.2.10/32"},
        "DateGreaterThanOrEqual": {"AWS:EpochTime": 1357034400},
        "DateLessThan": {"AWS:EpochTime": 1357120800}
      }
    }
  ]
}
```

Each signed URL in which you use this policy includes a base URL that identifies a specific object in a specific CloudFront distribution, for example:

<http://d111111abcdef8.cloudfront.net/training/orientation.pdf>

The signed URL also includes a key pair ID, which must be associated with a trusted signer in the distribution (d111111abcdef8.cloudfront.net) that you specify in the base URL.

Creating a Signature for a Signed URL That Uses a Custom Policy

The signature for a signed URL that uses a custom policy is a hashed, signed, and base64-encoded version of the policy statement. To create a signature for a custom policy, perform the applicable procedure. The version that you choose depends on your distribution type (web or RTMP) and, for RTMP distributions, the media player that you're using (Adobe Flash Player or another media player):

- [Option 1: To create a signature for a web distribution or for an RTMP distribution \(without Adobe Flash Player\) by using a custom policy \(p. 241\)](#)
- [Option 2: To create a signature for an RTMP distribution by using a custom policy \(Adobe Flash Player\) \(p. 242\)](#)

For additional information and examples of how to hash, sign, and encode the policy statement, see:

- [Using a Linux Command and OpenSSL for Base64-Encoding and Encryption \(p. 259\)](#)
- [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)

Option 1: To create a signature for a web distribution or for an RTMP distribution (without Adobe Flash Player) by using a custom policy

1. Use the SHA-1 hash function and RSA to hash and sign the JSON policy statement that you created in the procedure [To create the policy statement for a signed URL that uses a custom policy \(p. 236\)](#). Use the version of the policy statement that no longer includes whitespace but that has not yet been base64-encoded.

For the private key that is required by the hash function, use the private key that is associated with the applicable active trusted signer.

Note

The method that you use to hash and sign the policy statement depends on your programming language and platform. For sample code, see [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#).

2. Remove whitespace (including tabs and newline characters) from the hashed and signed string.
3. Base64-encode the string using MIME base64 encoding. For more information, see [Section 6.8, Base64 Content-Transfer-Encoding in RFC 2045, MIME \(Multipurpose Internet Mail Extensions\) Part One: Format of Internet Message Bodies](#).
4. Replace characters that are invalid in a URL query string with characters that are valid. The following table lists invalid and valid characters.

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| + | - (hyphen) |
| = | _ (underscore) |
| / | ~ (tilde) |

5. Append the resulting value to your signed URL after `&Signature=`, and return to [To create a signed URL using a custom policy \(p. 234\)](#) to finish concatenating the parts of your signed URL.

Option 2: To create a signature for an RTMP distribution by using a custom policy (Adobe Flash Player)

1. Use the SHA-1 hash function and RSA to hash and sign the JSON policy statement that you created in the [To create the policy statement for a signed URL that uses a custom policy \(p. 236\)](#) procedure. Use the version of the policy statement that no longer includes whitespace but that has not yet been base64-encoded.

For the private key that is required by the hash function, use the private key that is associated with the applicable active trusted signer.

Note

The method that you use to hash and sign the policy statement depends on your programming language and platform. For sample code, see [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#).

2. Remove whitespace (including tabs and newline characters) from the hashed and signed string.

Continue on to Step 3 if the stream name is passed in from a web page.

If the stream name is not passed in from a web page, skip the rest of this procedure. For example, if you wrote your own player that fetches stream names from within the Adobe Flash .swf file, skip the rest of this procedure.

3. Base64-encode the string using MIME base64 encoding. For more information, see [Section 6.8, Base64 Content-Transfer-Encoding in RFC 2045, MIME \(Multipurpose Internet Mail Extensions\) Part One: Format of Internet Message Bodies](#).
4. Replace characters that are invalid in a URL query string with characters that are valid. The following table lists invalid and valid characters.

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| + | - (hyphen) |
| = | _ (underscore) |
| / | ~ (tilde) |

5. Some versions of Adobe Flash Player require that you URL-encode the characters ?, =, and &. For information about whether your version of Adobe Flash Player requires this character substitution, refer to the Adobe website.

If your version of Adobe Flash Player does not require that you URL-encode the characters ?, =, and &, skip to Step 6.

If your version of Adobe Flash Player requires URL-encoding those characters, replace them as indicated in the following table. (You already replaced = in the previous step.)

| Replace these invalid characters | With this URL encoding |
|----------------------------------|------------------------|
| ? | %3F |
| & | %26 |

6. Append the resulting value to your signed URL after `&Signature=`, and return to [To create a signed URL using a custom policy \(p. 234\)](#) to finish concatenating the parts of your signed URL.

Using Signed Cookies

CloudFront signed cookies allow you to control who can access your content when you don't want to change your current URLs or when you want to provide access to multiple restricted files, for example, all of the files in the subscribers' area of a website. This topic explains the considerations when using signed cookies and describes how to set signed cookies using canned and custom policies.

Topics

- [Choosing Between Canned and Custom Policies for Signed Cookies \(p. 244\)](#)
- [How Signed Cookies Work \(p. 244\)](#)
- [Preventing Misuse of Signed Cookies \(p. 245\)](#)
- [When Does CloudFront Check the Expiration Date and Time in a Signed Cookie? \(p. 246\)](#)
- [Sample Code and Third-Party Tools \(p. 246\)](#)
- [Setting Signed Cookies Using a Canned Policy \(p. 246\)](#)
- [Setting Signed Cookies Using a Custom Policy \(p. 251\)](#)

Choosing Between Canned and Custom Policies for Signed Cookies

When you create a signed cookie, you write a policy statement in JSON format that specifies the restrictions on the signed cookie, for example, how long the cookie is valid. You can use canned policies or custom policies. The following table compares canned and custom policies:

| Description | Canned Policy | Custom Policy |
|---|---------------|----------------|
| You can reuse the policy statement for multiple objects. To reuse the policy statement, you must use wildcard characters in the <code>Resource</code> object. For more information, see Values That You Specify in the Policy Statement for a Custom Policy for Signed Cookies (p. 254) . | No | Yes |
| You can specify the date and time that users can begin to access your content | No | Yes (optional) |
| You can specify the date and time that users can no longer access your content | Yes | Yes |
| You can specify the IP address or range of IP addresses of the users who can access your content | No | Yes (optional) |

For information about creating signed cookies using a canned policy, see [Setting Signed Cookies Using a Canned Policy \(p. 246\)](#).

For information about creating signed cookies using a custom policy, see [Setting Signed Cookies Using a Custom Policy \(p. 251\)](#).

How Signed Cookies Work

Here's an overview of how you configure CloudFront for signed cookies and how CloudFront responds when a user submits a request that contains a signed cookie.

1. In your CloudFront distribution, you specify one or more trusted signers, which are the AWS accounts that you want to have permission to create signed URLs and signed cookies.

For more information, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#).

2. You develop your application to determine whether a user should have access to your content and, if so, to send three Set-Cookie headers to the viewer. (Each Set-Cookie header can contain only one name-value pair, and a CloudFront signed cookie requires three name-value pairs.) You must send the Set-Cookie headers to the viewer before the viewer requests your private content. If you set a short expiration time on the cookie, you might also want to send three more Set-Cookie headers in response to subsequent requests, so that the user continues to have access.

Typically, your CloudFront distribution will have at least two cache behaviors, one that doesn't require authentication and one that does. The error page for the secure portion of the site includes a redirector or a link to a login page.

If you configure your distribution to cache objects based on cookies, CloudFront doesn't cache separate objects based on the attributes in signed cookies.

3. A user signs in to your website and either pays for content or meets some other requirement for access.
4. Your application returns the Set-Cookie headers in the response, and the viewer stores the name-value pairs.
5. The user requests an object.

The user's browser or other viewer gets the name-value pairs from step 4 and adds them to the request in a Cookie header. This is the signed cookie.

6. CloudFront uses the public key to validate the signature in the signed cookie and to confirm that the cookie hasn't been tampered with. If the signature is invalid, the request is rejected.

If the signature in the cookie is valid, CloudFront looks at the policy statement in the cookie (or constructs one if you're using a canned policy) to confirm that the request is still valid. For example, if you specified a beginning and ending date and time for the cookie, CloudFront confirms that the user is trying to access your content during the time period that you want to allow access.

If the request meets the requirements in the policy statement, CloudFront serves your content as it does for content that isn't restricted: it determines whether the object is already in the edge cache, forwards the request to the origin if necessary, and returns the object to the user.

Preventing Misuse of Signed Cookies

If you specify the Domain parameter in a Set-Cookie header, specify the most precise value possible to limit potential access by someone with the same root domain name. For example, apex.example.com is preferable to example.com, especially when you don't control example.com. This helps prevent someone from accessing your content from nadir.example.com.

To prevent this type of attack, do the following:

- Exclude the Expires and Max-Age cookie attributes, so that the Set-Cookie header creates a session cookie. Session cookies are automatically deleted when the user closes the browser, which reduces the possibility of someone getting unauthorized access to your content.
- Include the Secure attribute, so that the cookie is encrypted when a viewer includes it in a request.
- When possible, use a custom policy and include the IP address of the viewer.
- In the CloudFront-Expires attribute, specify the shortest reasonable expiration time based on how long you want users to have access to your content.

When Does CloudFront Check the Expiration Date and Time in a Signed Cookie?

To determine whether a signed cookie is still valid, CloudFront checks the expiration date and time in the cookie at the time of the HTTP request. If a client begins to download a large object immediately before the expiration time, the download should complete even if the expiration time passes during the download. If the TCP connection drops and the client tries to restart the download after the expiration time passes, the download will fail.

If a client uses Range GETs to get an object in smaller pieces, any GET request that occurs after the expiration time passes will fail. For more information about Range GETs, see [How CloudFront Processes Partial Requests for an Object \(Range GETs\) \(p. 176\)](#).

Sample Code and Third-Party Tools

The sample code for private content shows only how to create the signature for signed URLs. However, the process for creating a signature for a signed cookie is very similar, so much of the sample code is still relevant. For more information, see the following topics:

- [Create a URL Signature Using Perl \(p. 260\)](#)
- [Create a URL Signature Using PHP \(p. 268\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 270\)](#)
- [Create a URL Signature Using Java \(p. 276\)](#)

Additional sample code for creating signed URLs is available on the [Amazon CloudFront Sample Code & Libraries](#) page.

For information about third-party tools that support private content, including creating signed URLs, see [Tools and Code Examples for Configuring Private Content \(p. 366\)](#).

Setting Signed Cookies Using a Canned Policy

Topics

- [Creating a Signature for a Signed Cookie That Uses a Canned Policy \(p. 248\)](#)

To set a signed cookie using a canned policy, perform the following procedure.

To set a signed cookie using a canned policy

1. If you're using .NET or Java to create signed URLs, and if you haven't reformatted the private key for your key pair from the default .pem format to a format compatible with .NET or with Java, do so now. For more information, see [Reformatting the CloudFront Private Key \(.NET and Java Only\) \(p. 218\)](#).
2. Program your application to send three Set-Cookie headers to approved viewers. You need three Set-Cookie headers because each Set-Cookie header can contain only one name-value pair, and a CloudFront signed cookie requires three name-value pairs. The name-value pairs are: `CloudFront-Expires`, `CloudFront-Signature`, and `CloudFront-Key-Pair-Id`. The values must be present on the viewer before a user makes the first request for an object that you want to control access to.

Note

In general, we recommend that you exclude `Expires` and `Max-Age` attributes. Excluding the attributes causes the browser to delete the cookie when the user closes the browser,

which reduces the possibility of someone getting unauthorized access to your content. For more information, see [Preventing Misuse of Signed Cookies \(p. 245\)](#).

The names of cookie attributes are case sensitive.

Line breaks are included only to make the attributes more readable.

```
Set-Cookie:  
Domain=optional domain name;  
Path=/optional directory path;  
Secure;  
HttpOnly;  
CloudFront-Expires=date and time in Unix time format (in seconds) and Coordinated Universal Time (UTC)  
  
Set-Cookie:  
Domain=optional domain name;  
Path=/optional directory path;  
Secure;  
HttpOnly;  
CloudFront-Signature=hashed and signed version of the policy statement  
  
Set-Cookie:  
Domain=optional domain name;  
Path=/optional directory path;  
Secure;  
HttpOnly;  
CloudFront-Key-Pair-Id=active CloudFront key pair Id for the key pair that you are using to generate the signature
```

(Optional) Domain

The domain name for the requested object. If you don't specify a `Domain` attribute, the default value is the domain name in the URL, and it applies only to the specified domain name, not to subdomains. If you specify a `Domain` attribute, it also applies to subdomains. A leading dot in the domain name (for example, `Domain=.example.com`) is optional. In addition, if you specify a `Domain` attribute, the domain name in the URL and the value of the `Domain` attribute must match.

You can specify the domain name that CloudFront assigned to your distribution, for example, `d111111abcdef8.cloudfront.net`, but you can't specify `*.cloudfront.net` for the domain name.

If you want to use an alternate domain name such as `example.com` in URLs, you must add the alternate domain name to your distribution regardless of whether you specify the `Domain` attribute. For more information, see [Alternate Domain Names \(CNAMEs\) \(p. 78\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

(Optional) Path

The path for the requested object. If you don't specify a `Path` attribute, the default value is the path in the URL.

Secure

Requires that the viewer encrypt cookies before sending a request. We recommend that you send the `Set-Cookie` header over an HTTPS connection to ensure that the cookie attributes are protected from man-in-the-middle attacks.

HttpOnly

Requires that the viewer send the cookie only in HTTP or HTTPS requests.

CloudFront-Expires

Specify the expiration date and time in Unix time format and Coordinated Universal Time (UTC). For example, March 16, 2015 10:00 am UTC converts to 1426500000 in Unix time format. For information about UTC, see *RFC 3339, Date and Time on the Internet: Timestamps*, <http://tools.ietf.org/html/rfc3339>.

CloudFront-Signature

A hashed, signed, and base64-encoded version of a JSON policy statement. For more information, see [Creating a Signature for a Signed Cookie That Uses a Canned Policy \(p. 248\)](#).

CloudFront-Key-Pair-Id

The ID for an active CloudFront key pair, for example, APKA9ONS7QCOWEXAMPLE. The CloudFront key pair ID tells CloudFront which public key to use to validate the signed cookie. CloudFront compares the information in the signature with the information in the policy statement to verify that the URL has not been tampered with.

The key pair ID that you include in CloudFront signed cookies must be associated with an AWS account that is one of the trusted signers for the applicable cache behavior.

For more information, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#).

If you make a key pair inactive while rotating CloudFront key pairs, you must update your application to use a new active key pair for one of your trusted signers. For more information about rotating key pairs, see [Rotating CloudFront Key Pairs \(p. 221\)](#).

Example Set-Cookie headers for one signed cookie when you're using the domain name that is associated with your distribution in the URLs for your objects:

```
Set-Cookie: Domain=d11111abcdef8.cloudfront.net; Path=/images/*; Secure; HttpOnly;
CloudFront-Expires=1426500000
Set-Cookie: Domain=d11111abcdef8.cloudfront.net; Path=/images/*; Secure; HttpOnly;
CloudFront-Signature=yXrSIgyQoeE4FBI4eMKF6ho~CA8_
Set-Cookie: Domain=d11111abcdef8.cloudfront.net; Path=/images/*; Secure; HttpOnly;
CloudFront-Key-Pair-Id=APKA9ONS7QCOWEXAMPLE
```

Example Set-Cookie headers for one signed cookie when you're using the alternate domain name example.org in the URLs for your objects:

```
Set-Cookie: Domain=example.org; Path=/images/*; Secure; HttpOnly; CloudFront-
Expires=1426500000
Set-Cookie: Domain=example.org; Path=/images/*; Secure; HttpOnly; CloudFront-
Signature=yXrSIgyQoeE4FBI4eMKF6ho~CA8_
Set-Cookie: Domain=example.org; Path=/images/*; Secure; HttpOnly; CloudFront-Key-Pair-
Id=APKA9ONS7QCOWEXAMPLE
```

If you want to use an alternate domain name such as example.com in URLs, you must add the alternate domain name to your distribution regardless of whether you specify the Domain attribute. For more information, see [Alternate Domain Names \(CNAMEs\) \(p. 78\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

[Creating a Signature for a Signed Cookie That Uses a Canned Policy](#)

To create the signature for a signed cookie that uses a canned policy, you perform the following tasks:

1. Create a policy statement. See [Creating a Policy Statement for a Signed Cookie That Uses a Canned Policy \(p. 249\)](#).
2. Sign the policy statement to create a signature. See [Signing the Policy Statement to Create a Signature for a Signed Cookie That Uses a Canned Policy \(p. 250\)](#).

Creating a Policy Statement for a Signed Cookie That Uses a Canned Policy

When you set a signed cookie that uses a canned policy, the `CloudFront-Signature` attribute is a hashed and signed version of a policy statement. For signed cookies that use a canned policy, you don't include the policy statement in the `Set-Cookie` header, as you do for signed cookies that use a custom policy. To create the policy statement, perform the following procedure.

To create a policy statement for a signed cookie that uses a canned policy

1. Construct the policy statement using the following JSON format and using UTF-8 character encoding. Include all punctuation and other literal values exactly as specified. For information about the `Resource` and `DateLessThan` parameters, see [Values That You Specify in the Policy Statement for a Canned Policy for Signed Cookies \(p. 249\)](#).

```
{  
    "Statement": [  
        {  
            "Resource": "base URL or stream name",  
            "Condition": {  
                "DateLessThan": {  
                    "AWS:EpochTime": ending date and time in Unix time format and UTC  
                }  
            }  
        }  
    ]  
}
```

2. Remove all whitespace (including tabs and newline characters) from the policy statement. You might have to include escape characters in the string in application code.

Values That You Specify in the Policy Statement for a Canned Policy for Signed Cookies

When you create a policy statement for a canned policy, you specify the following values:

Resource

The base URL including your query strings, if any, for example:

```
http://d11111abcdef8.cloudfront.net/images/horizon.jpg?  
size=large&license=yes
```

You can specify only one value for `Resource`.

Note the following:

- **Protocol** – The value must begin with `http://` or `https://`.
- **Query string parameters** – If you have no query string parameters, omit the question mark.
- **Alternate domain names** – If you specify an alternate domain name (CNAME) in the URL, you must specify the alternate domain name when referencing the object in your web page or application. Do not specify the Amazon S3 URL for the object.

DateLessThan

The expiration date and time for the URL in Unix time format (in seconds) and Coordinated Universal Time (UTC). Do not enclose the value in quotation marks.

For example, March 16, 2015 10:00 am UTC converts to 1426500000 in Unix time format.

This value must match the value of the `CloudFront-Expires` attribute in the `Set-Cookie` header. Do not enclose the value in quotation marks.

For more information, see [When Does CloudFront Check the Expiration Date and Time in a Signed Cookie? \(p. 246\)](#).

Example Policy Statement for a Canned Policy

When you use the following example policy statement in a signed cookie, a user can access the object `http://d111111abcdef8.cloudfront.net/horizon.jpg` until March 16, 2015 10:00 am UTC:

```
{  
    "Statement": [  
        {  
            "Resource": "http://d111111abcdef8.cloudfront.net/horizon.jpg?  
size=large&license=yes",  
            "Condition": {  
                "DateLessThan": {  
                    "AWS:EpochTime": 1426500000  
                }  
            }  
        }  
    ]  
}
```

Signing the Policy Statement to Create a Signature for a Signed Cookie That Uses a Canned Policy

To create the value for the `CloudFront-Signature` attribute in a `Set-Cookie` header, you hash and sign the policy statement that you created in [To create a policy statement for a signed cookie that uses a canned policy \(p. 249\)](#).

For additional information and examples of how to hash, sign, and encode the policy statement, see the following topics:

- [Using a Linux Command and OpenSSL for Base64-Encoding and Encryption \(p. 259\)](#)
- [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)

To create a signature for a signed cookie using a canned policy

1. Use the SHA-1 hash function and RSA to hash and sign the policy statement that you created in the procedure [To create a policy statement for a signed cookie that uses a canned policy \(p. 249\)](#). Use the version of the policy statement that no longer includes whitespace.

For the private key that is required by the hash function, use the private key that is associated with the applicable active trusted signer.

Note

The method that you use to hash and sign the policy statement depends on your programming language and platform. For sample code, see [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#).

2. Remove whitespace (including tabs and newline characters) from the hashed and signed string.
3. Base64-encode the string using MIME base64 encoding. For more information, see [Section 6.8, Base64 Content-Transfer-Encoding in RFC 2045, MIME \(Multipurpose Internet Mail Extensions\) Part One: Format of Internet Message Bodies](#).

4. Replace characters that are invalid in a URL query string with characters that are valid. The following table lists invalid and valid characters.

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| + | - (hyphen) |
| = | _ (underscore) |
| / | ~ (tilde) |

5. Include the resulting value in the `Set-Cookie` header for the `CloudFront-Signature` name-value pair. Then return to [To set a signed cookie using a canned policy \(p. 246\)](#) add the `Set-Cookie` header for `CloudFront-Key-Pair-Id`.

Setting Signed Cookies Using a Custom Policy

Topics

- [Creating a Policy Statement for a Signed Cookie That Uses a Custom Policy \(p. 253\)](#)
- [Example Policy Statements for a Signed Cookie That Uses a Custom Policy \(p. 256\)](#)
- [Creating a Signature for a Signed Cookie That Uses a Custom Policy \(p. 257\)](#)

To set a signed cookie that uses a custom policy, perform the following procedure.

To set a signed cookie using a custom policy

1. If you're using .NET or Java to create signed URLs, and if you haven't reformatted the private key for your key pair from the default .pem format to a format compatible with .NET or with Java, do so now. For more information, see [Reformatting the CloudFront Private Key \(.NET and Java Only\) \(p. 218\)](#).
2. Program your application to send three `Set-Cookie` headers to approved viewers. You need three `Set-Cookie` headers because each `Set-Cookie` header can contain only one name-value pair, and a CloudFront signed cookie requires three name-value pairs. The name-value pairs are: `CloudFront-Policy`, `CloudFront-Signature`, and `CloudFront-Key-Pair-Id`. The values must be present on the viewer before a user makes the first request for an object that you want to control access to.

Note

In general, we recommend that you exclude `Expires` and `Max-Age` attributes. This causes the browser to delete the cookie when the user closes the browser, which reduces the possibility of someone getting unauthorized access to your content. For more information, see [Preventing Misuse of Signed Cookies \(p. 245\)](#).

The names of cookie attributes are case sensitive.

Line breaks are included only to make the attributes more readable.

```

Set-Cookie:
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly;
CloudFront-Policy=base64 encoded version of the policy statement

Set-Cookie:
Domain=optional domain name;

```

```
Path=/optional directory path;
Secure;
HttpOnly;
CloudFront-Signature=hashed and signed version of the policy statement

Set-Cookie:
Domain=optional domain name;
Path=/optional directory path;
Secure;
HttpOnly;
CloudFront-Key-Pair-Id=active CloudFront key pair Id for the key pair that you are
using to generate the signature
```

(Optional) Domain

The domain name for the requested object. If you don't specify a `Domain` attribute, the default value is the domain name in the URL, and it applies only to the specified domain name, not to subdomains. If you specify a `Domain` attribute, it also applies to subdomains. A leading dot in the domain name (for example, `Domain=.example.com`) is optional. In addition, if you specify a `Domain` attribute, the domain name in the URL and the value of the `Domain` attribute must match.

You can specify the domain name that CloudFront assigned to your distribution, for example, `d111111abcdef8.cloudfront.net`, but you can't specify `*.cloudfront.net` for the domain name.

If you want to use an alternate domain name such as `example.com` in URLs, you must add the alternate domain name to your distribution regardless of whether you specify the `Domain` attribute. For more information, see [Alternate Domain Names \(CNAMEs\) \(p. 78\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

(Optional) Path

The path for the requested object. If you don't specify a `Path` attribute, the default value is the path in the URL.

Secure

Requires that the viewer encrypt cookies before sending a request. We recommend that you send the `Set-Cookie` header over an HTTPS connection to ensure that the cookie attributes are protected from man-in-the-middle attacks.

HttpOnly

Requires that the viewer send the cookie only in HTTP or HTTPS requests.

CloudFront-Policy

Your policy statement in JSON format, with white space removed, then base64 encoded. For more information, see [Creating a Signature for a Signed Cookie That Uses a Custom Policy \(p. 257\)](#).

The policy statement controls the access that a signed cookie grants to a user: the objects that the user can access, an expiration date and time, an optional date and time that the URL becomes valid, and an optional IP address or range of IP addresses that are allowed to access the object.

CloudFront-Signature

A hashed, signed, and base64-encoded version of the JSON policy statement. For more information, see [Creating a Signature for a Signed Cookie That Uses a Custom Policy \(p. 257\)](#).

CloudFront-Key-Pair-Id

The ID for an active CloudFront key pair, for example, APKA9ONS7QCOWEXAMPLE. The CloudFront key pair ID tells CloudFront which public key to use to validate the signed cookie. CloudFront compares the information in the signature with the information in the policy statement to verify that the URL has not been tampered with.

The key pair ID that you include in CloudFront signed cookies must be associated with an AWS account that is one of the trusted signers for the applicable cache behavior.

For more information, see [Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies \(Trusted Signers\) \(p. 216\)](#).

If you make a key pair inactive while rotating CloudFront key pairs, you must update your application to use a new active key pair for one of your trusted signers. For more information about rotating key pairs, see [Rotating CloudFront Key Pairs \(p. 221\)](#).

Example Set-Cookie headers for one signed cookie when you're using the domain name that is associated with your distribution in the URLs for your objects:

```
Set-Cookie: Domain=d11111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly; CloudFront-Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dhbWVfZG93Set-Cookie: Domain=d11111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly; CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_Set-Cookie: Domain=d11111abcdef8.cloudfront.net; Path=/; Secure; HttpOnly; CloudFront-Key-Pair-Id=APKA9ONS7QCOWEXAMPLE
```

Example Set-Cookie headers for one signed cookie when you're using the alternate domain name example.org in the URLs for your objects:

```
Set-Cookie: Domain=example.org; Path=/; Secure; HttpOnly; CloudFront-Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc291cmNlIjoiaHR0cDovL2QxMTExMTFhYmNkZWY4LmNsb3VkZnJvbnQubmV0L2dhbWVfZG93Set-Cookie: Domain=example.org; Path=/; Secure; HttpOnly; CloudFront-Signature=dtKhpJ3aUYxqDIwepczPiDb9NXQ_Set-Cookie: Domain=example.org; Path=/; Secure; HttpOnly; CloudFront-Key-Pair-Id=APKA9ONS7QCOWEXAMPLE
```

If you want to use an alternate domain name such as example.com in URLs, you must add the alternate domain name to your distribution regardless of whether you specify the Domain attribute. For more information, see [Alternate Domain Names \(CNAMEs\) \(p. 78\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

Creating a Policy Statement for a Signed Cookie That Uses a Custom Policy

To create a policy statement for a custom policy, perform the following procedure. For several example policy statements that control access to objects in a variety of ways, see [Example Policy Statements for a Signed Cookie That Uses a Custom Policy \(p. 256\)](#).

To create the policy statement for a signed cookie that uses a custom policy

1. Construct the policy statement using the following JSON format.

```
{  
  "Statement": [  
    {  
      "Resource": "URL of the object",  
      "Condition": {  
        "StringLike": {  
          "Fn::Equals": {  
            "Fn::GetHeader": "Cookie",  
            "Value": "CloudFront-Key-Pair-Id=APKA9ONS7QCOWEXAMPLE" // replace with your CloudFront key pair ID  
          }  
        }  
      }  
    }  
  ]  
}
```

```

        "Condition": {
            "DateLessThan": {"AWS:EpochTime": required ending date and time in Unix time
                format and UTC},
            "DateGreaterThanOrEqual": {"AWS:EpochTime": optional beginning date and time in Unix
                time format and UTC},
            "IpAddress": {"AWS:SourceIp": optional IP address}
        }
    }
}

```

Note the following:

- You can include only one statement.
 - Use UTF-8 character encoding.
 - Include all punctuation and parameter names exactly as specified. Abbreviations for parameter names are not accepted.
 - The order of the parameters in the Condition section doesn't matter.
 - For information about the values for Resource, DateLessThan, DateGreaterThanOrEqual, and IpAddress, see [Values That You Specify in the Policy Statement for a Custom Policy for Signed Cookies \(p. 254\)](#).
2. Remove all whitespace (including tabs and newline characters) from the policy statement. You might have to include escape characters in the string in application code.
 3. Base64-encode the policy statement using MIME base64 encoding. For more information, see [Section 6.8, Base64 Content-Transfer-Encoding in RFC 2045, MIME \(Multipurpose Internet Mail Extensions\) Part One: Format of Internet Message Bodies](#).
 4. Replace characters that are invalid in a URL query string with characters that are valid. The following table lists invalid and valid characters.

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| + | - (hyphen) |
| = | _ (underscore) |
| / | ~ (tilde) |

5. Include the resulting value in your Set-Cookie header after CloudFront-Policy=.
6. Create a signature for the Set-Cookie header for CloudFront-Signature by hashing, signing, and base64-encoding the policy statement. For more information, see [Creating a Signature for a Signed Cookie That Uses a Custom Policy \(p. 257\)](#).

Values That You Specify in the Policy Statement for a Custom Policy for Signed Cookies

When you create a policy statement for a custom policy, you specify the following values.

Resource

The base URL including your query strings, if any:

```
http://d111111abcdef8.cloudfront.net/images/horizon.jpg?
size=large&license=yes
```

Important

If you omit the Resource parameter, users can access all of the objects associated with any distribution that is associated with the key pair that you use to create the signed URL.

You can specify only one value for `Resource`.

Note the following:

- **Protocol** – The value must begin with `http://` or `https://`.
- **Query string parameters** – If you have no query string parameters, omit the question mark.
- **Wildcards** – You can use the wildcard character that matches zero or more characters (*) or the wild-card character that matches exactly one character (?) anywhere in the string. For example, the value:

`http://d111111abcdef8.cloudfront.net/*game_download.zip*`

would include (for example) the following objects:

- `http://d111111abcdef8.cloudfront.net/game_download.zip`
- `http://d111111abcdef8.cloudfront.net/example_game_download.zip?license=yes`
- `http://d111111abcdef8.cloudfront.net/test_game_download.zip?license=temp`
- **Alternate domain names** – If you specify an alternate domain name (CNAME) in the URL, you must specify the alternate domain name when referencing the object in your web page or application. Do not specify the Amazon S3 URL for the object.

DateLessThan

The expiration date and time for the URL in Unix time format (in seconds) and Coordinated Universal Time (UTC). Do not enclose the value in quotation marks.

For example, March 16, 2015 10:00 am UTC converts to 1426500000 in Unix time format.

For more information, see [When Does CloudFront Check the Expiration Date and Time in a Signed Cookie? \(p. 246\)](#).

DateGreaterThanOrEqual (Optional)

An optional start date and time for the URL in Unix time format (in seconds) and Coordinated Universal Time (UTC). Users are not allowed to access the object before the specified date and time. Do not enclose the value in quotation marks.

IpAddress (Optional)

The IP address of the client making the GET request. Note the following:

- To allow any IP address to access the object, omit the `IpAddress` parameter.
- You can specify either one IP address or one IP address range. For example, you can't set the policy to allow access if the client's IP address is in one of two separate ranges.
- To allow access from a single IP address, you specify:

`"IPv4 IP address/32"`

- You must specify IP address ranges in standard IPv4 CIDR format (for example, `192.0.2.0/24`). For more information, go to *RFC 4632, Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*, <http://tools.ietf.org/html/rfc4632>.

Important

IP addresses in IPv6 format, such as `2001:0db8:85a3:0000:0000:8a2e:0370:7334`, are not supported.

If you're using a custom policy that includes `IpAddress`, do not enable IPv6 for the distribution. If you want to restrict access to some content by IP address and support IPv6 requests for other content, you can create two distributions. For more information, see [Enable IPv6 \(p. 81\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

Example Policy Statements for a Signed Cookie That Uses a Custom Policy

The following example policy statements show how to control access to a specific object, all of the objects in a directory, or all of the objects associated with a key pair ID. The examples also show how to control access from an individual IP address or a range of IP addresses, and how to prevent users from using the signed cookie after a specified date and time.

If you copy and paste any of these examples, remove any whitespace (including tabs and newline characters), replace the applicable values with your own values, and include a newline character after the closing brace (}).

For more information, see [Values That You Specify in the Policy Statement for a Custom Policy for Signed Cookies \(p. 254\)](#).

Topics

- [Example Policy Statement: Accessing One Object from a Range of IP Addresses \(p. 256\)](#)
- [Example Policy Statement: Accessing All Objects in a Directory from a Range of IP Addresses \(p. 256\)](#)
- [Example Policy Statement: Accessing All Objects Associated with a Key Pair ID from One IP Address \(p. 257\)](#)

Example Policy Statement: Accessing One Object from a Range of IP Addresses

The following example custom policy in a signed cookie specifies that a user can access the object `http://d111111abcdef8.cloudfront.net/game_download.zip` from IP addresses in the range `192.0.2.0/24` until January 1, 2013 10:00 am UTC:

```
{  
    "Statement": [  
        {  
            "Resource": "http://d111111abcdef8.cloudfront.net/game_download.zip",  
            "Condition": {  
                "IpAddress": {"AWS:SourceIp": "192.0.2.0/24"},  
                "DateLessThan": {"AWS:EpochTime": 1357034400}  
            }  
        }  
    ]  
}
```

Example Policy Statement: Accessing All Objects in a Directory from a Range of IP Addresses

The following example custom policy allows you to create signed cookies for any object in the `training` directory, as indicated by the `*` wildcard character in the `Resource` parameter. Users can access the object from an IP address in the range `192.0.2.0/24` until January 1, 2013 10:00 am UTC:

```
{  
    "Statement": [  
        {  
            "Resource": "http://d111111abcdef8.cloudfront.net/training/*",  
            "Condition": {  
                "IpAddress": {"AWS:SourceIp": "192.0.2.0/24"},  
                "DateLessThan": {"AWS:EpochTime": 1357034400}  
            }  
        }  
    ]  
}
```

```
    ]  
}
```

Each signed cookie in which you use this policy includes a base URL that identifies a specific object, for example:

<http://d111111abcdef8.cloudfront.net/training/orientation.pdf>

Example Policy Statement: Accessing All Objects Associated with a Key Pair ID from One IP Address

The following sample custom policy allows you to set signed cookies for any object associated with any distribution, as indicated by the * wildcard character in the Resource parameter. The user must use the IP address 192.0.2.10/32. (The value 192.0.2.10/32 in CIDR notation refers to a single IP address, 192.0.2.10.) The objects are available only from January 1, 2013 10:00 am UTC until January 2, 2013 10:00 am UTC:

```
{  
    "Statement": [  
        {  
            "Resource": "http:///*",  
            "Condition": {  
                "IpAddress": {"AWS:SourceIp": "192.0.2.10/32"},  
                "DateGreaterThanOrEqual": {"AWS:EpochTime": 1357034400},  
                "DateLessThan": {"AWS:EpochTime": 1357120800}  
            }  
        }  
    ]  
}
```

Each signed cookie in which you use this policy includes a base URL that identifies a specific object in a specific CloudFront distribution, for example:

<http://d111111abcdef8.cloudfront.net/training/orientation.pdf>

The signed cookie also includes a key pair ID, which must be associated with a trusted signer in the distribution (d111111abcdef8.cloudfront.net) that you specify in the base URL.

Creating a Signature for a Signed Cookie That Uses a Custom Policy

The signature for a signed cookie that uses a custom policy is a hashed, signed, and base64-encoded version of the policy statement.

For additional information and examples of how to hash, sign, and encode the policy statement, see:

- [Using a Linux Command and OpenSSL for Base64-Encoding and Encryption \(p. 259\)](#)
- [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)

To create a signature for a signed cookie by using a custom policy

1. Use the SHA-1 hash function and RSA to hash and sign the JSON policy statement that you created in the procedure [To create the policy statement for a signed URL that uses a custom policy \(p. 236\)](#). Use the version of the policy statement that no longer includes whitespace but that has not yet been base64-encoded.

For the private key that is required by the hash function, use the private key that is associated with the applicable active trusted signer.

Note

The method that you use to hash and sign the policy statement depends on your programming language and platform. For sample code, see [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#).

2. Remove whitespace (including tabs and newline characters) from the hashed and signed string.
3. Base64-encode the string using MIME base64 encoding. For more information, see [Section 6.8, Base64 Content-Transfer-Encoding in RFC 2045, MIME \(Multipurpose Internet Mail Extensions\) Part One: Format of Internet Message Bodies](#).
4. Replace characters that are invalid in a URL query string with characters that are valid. The following table lists invalid and valid characters.

| Replace these invalid characters | With these valid characters |
|----------------------------------|-----------------------------|
| + | - (hyphen) |
| = | _ (underscore) |
| / | ~ (tilde) |

5. Include the resulting value in the `Set-Cookie` header for the `CloudFront-Signature=` name-value pair, and return to [To set a signed cookie using a custom policy \(p. 251\)](#) to add the `Set-Cookie` header for `CloudFront-Key-Pair-Id`.

Using a Linux Command and OpenSSL for Base64-Encoding and Encryption

You can use the following Linux command-line command and OpenSSL to hash and sign the policy statement, base64-encode the signature, and replace characters that are not valid in URL query string parameters with characters that are valid.

For information about OpenSSL, go to <http://www.openssl.org>.

```
❶ cat policy | ❷ tr -d "\n" | ❸ openssl sha1 -sign private-key.pem | ❹
openssl base64 | ❺ tr -- '+=/-' '_~'
```

where:

- ❶ cat reads the policy file.
- ❷ tr -d "\n" removes a newline character that was added by cat.
- ❸ OpenSSL hashes the file using SHA-1 and signs it using RSA and the private key file `private-key.pem`.
- ❹ OpenSSL base64-encodes the hashed and signed policy statement.
- ❺ tr replaces characters that are not valid in URL query string parameters with characters that are valid.

For code examples that demonstrate creating a signature in several programming languages see [Code Examples for Creating a Signature for a Signed URL \(p. 260\)](#).

Code Examples for Creating a Signature for a Signed URL

This section includes downloadable application examples that demonstrate how to create signatures for signed URLs. Examples are available in Perl, PHP, C#, and Java. You can use any of the examples to create signed URLs. The Perl script runs on Linux/Mac platforms. The PHP example will work on any server that runs PHP. The C# example uses the .NET Framework.

For an example of how to use cookies with Ruby on Rails, see [Tools and Code Examples for Configuring Private Content \(p. 366\)](#) in the topic [Amazon CloudFront Resources \(p. 364\)](#).

You can find example code for signed URLs and for signed cookies in a variety of programming languages. Perform an internet search on sample app *language* cloudfont signed URLs or on sample app *language* cloudfont signed cookies.

Topics

- [Create a URL Signature Using Perl \(p. 260\)](#)
- [Create a URL Signature Using PHP \(p. 268\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 270\)](#)
- [Create a URL Signature Using Java \(p. 276\)](#)

Create a URL Signature Using Perl

This section includes a Perl script for Linux/Mac platforms that you can use to create the signature for private content. To create the signature, run the script with command line arguments that specify the CloudFront URL, the path to the private key of the signer, the key ID, and an expiration date for the URL. The tool can also decode signed URLs.

Note

Creating a URL signature is just one part of the process of serving private content using a signed URL. For more information about the end-to-end process, see [Using Signed URLs \(p. 224\)](#).

Topics

- [Example of Using a Perl Script to Create a Signed URL \(p. 260\)](#)
- [Source for the Perl Script to Create a Signed URL \(p. 262\)](#)

Example of Using a Perl Script to Create a Signed URL

The following example shows how you can use the Perl script provided in this topic to create an RTMP distribution signature. To start, save the script as a file called cfsign.pl. Then run the script using the following command line arguments:

```
$ cfsign.pl --action encode --stream example/video.mp4 --private-key /path/to/my-private-key.pem --key-pair-id PK12345EXAMPLE --expires 1265838202
```

This script generates the policy statement from the command line arguments. The signature that it generates is an SHA1 hash of the policy statement.

The following is an example base64-encoded stream name:

```
mp4:example/video.mp4%3FPolicy%3DewogICJTdGF0ZW1lbnoiolt7CiAgICAgICJSZXNvdXJjZSI
```

```
6ImRyciIsCiAgICAgICJDb25kaXRpb24iOnsKICAgICAgICAiSXBBZGRyZXNzIjp7IkFXUzpTb3VyY2V
JcC16IjAuMC4wLjAvMCJ9LAogICAgICAgICJEXYR1tGVzc1RoYW4iOnsiQvdTOkVwb2NoVGltZSI6MjE
ONTkxNjgwMH0KICAgICAgfQogICAgEXAMPLE_%26Signature%3DewtHqEXK~68tsZt-eOFnZKGwTf2a
J1bKhXkK5SSiVqcG9pieCRV3xTEPtc29OzeXlsDvRycOM2WK0cXzcyYZhpl9tv2796ihHiCTAwIHQ8yP
17Af4nWtOLIZHoH6wkR3tU1cQHs8R1d-g-SlZGjNBXr~J2MbaJzm8i6EXAMPLE_%26Key-Pair-Id%3
DPK12345EXAMPLE
```

This signature authenticates the request to stream the private content `example/video.mp4`.

If you're using Adobe Flash Player and the stream name is passed in from a web page using JavaScript, you must base64-encode the signature and replace characters that are invalid in a URL request parameter (+, =, /) with characters that are valid (-, _, and ~, respectively).

If the stream name is not passed in from a web page, you don't need to base64-encode the signature. For example, you would not base64-encode the signature if you write your own player and the stream names are fetched from within the Adobe Flash .swf file.

The following example uses jwplayer with CloudFront.

```
<script type='text/javascript'>
var so1 = new SWFObject
    ('http://d841721fxaaqy9.cloudfront.net/player/player.swf',
     'mpl', '640', '360', '9');
so1.addParam('allowfullscreen','true');
so1.addParam('allowscriptaccess','always');
so1.addParam('wmode','opaque');
so1.addVariable('streamer','rtmp://s33r3xe4ayhhis.cloudfront.net/cfx/st');
so1.addVariable("file","mp4:example/video.mp4?Policy%3DewogICJTdGF0ZW1lbNQi
Olt7CiAgICAgICJSZXNvdXJjZSI6ImRyciIsCiAgICAgICJDb25kaXRpb24iOnsKICAgICAgICA
iSXBBZGRyZXNzIjp7IkFXUzpTb3VyY2VjC16IjAuMC4wLjAvMCJ9LAogICAgICAgICJEXYR1TG
Vzc1RoYW4iOnsiQvdTOkVwb2NoVGltZSI6MjEONTkxNjgwMH0KICAgICAgfQogICAgEXAMPLE_%
26Signature%3DewtHqEXK~68tsZt-eOFnZKGwTf2aJ1bKhXkK5SSiVqcG9pieCRV3xTEPtc29O
zeXlsDvRycOM2WK0cXzcyYZhpl9tv2796ihHiCTAwIHQ8yP17Af4nWtOLIZHoH6wkR3tU1cQHs8
R1d-g-SlZGjNBXr~J2MbaJzm8i6EXAMPLE_%26Key-Pair-Id%3DPK12345EXAMPLE
so1.write('flv');
</script>
```

When you retrieve a stream to play from within an Adobe Flash .swf file, do not URL-encode the stream name. For example:

```
mp4:example/video.mp4?Policy=ewogICJTdGF0ZW1lbNQiOlt7CiAgICAgICJSZXNvdXJjZSI6Im
yciIsCiAgICAgICJDb25kaXRpb24iOnsKICAgICAgICAiSXBBZGRyZXNzIjp7IkFXUzpTb3VyY2VjC1
6IjAuMC4wLjAvMCJ9LAogICAgICAgICJEXYR1tGVzc1RoYW4iOnsiQvdTOkVwb2NoVGltZSI6MjEONTk
xNjgwMH0KICAgICAgfQogICAgEXAMPLE_&Signature=ewtHqEXK~68tsZt-eOFnZKGwTf2aJ1bKhXkK
5SSiVqcG9pieCRV3xTEPtc29OzeXlsDvRycOM2WK0cXzcyYZhpl9tv2796ihHiCTAwIHQ8yP17Af4nWt
OLIZHoH6wkR3tU1cQHs8R1d-g-SlZGjNBXr~J2MbaJzm8i6EXAMPLE_&Key-Pair-Id=PK12345
EXAMPLE
```

For more information about the command line switches and features of this tool see the comments in the Perl source code, which is included in the next section.

See also

- [Create a URL Signature Using PHP \(p. 268\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 270\)](#)
- [Create a URL Signature Using Java \(p. 276\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)

Source for the Perl Script to Create a Signed URL

The following Perl source code can be used to create a signed URL for CloudFront. Comments in the code include information about the command line switches and the features of the tool.

```
#!/usr/bin/perl -w

# Copyright 2008 Amazon Technologies, Inc. Licensed under the Apache License, Version 2.0
# (the "License");
# you may not use this file except in compliance with the License. You may obtain a copy of
# the License at:
#
# http://aws.amazon.com/apache2.0
#
# This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
# KIND, either express or implied.
# See the License for the specific language governing permissions and limitations under the
# License.

=head1 cfsign.pl

cfsign.pl - A tool to generate and verify AWS CloudFront signed URLs

=head1 SYNOPSIS

This script uses an existing RSA key pair to sign and verify AWS CloudFront signed URLs

View the script source for details as to which CPAN packages are required beforehand.

For help, try:

cfsign.pl --help

URL signing examples:

cfsign.pl --action encode --url http://images.my-website.com/gallery1.zip --policy
sample_policy.json --private-key privkey.pem --key-pair-id mykey

cfsign.pl --action encode --url http://images.my-website.com/gallery1.zip --expires
1257439868 --private-key privkey.pem --key-pair-id mykey

Stream signing example:

cfsign.pl --action encode --stream videos/myvideo.mp4 --expires 1257439868 --private-key
privkey.pem --key-pair-id mykey

URL decode example:

cfsign.pl --action decode --url "http://mydist.cloudfront.net/?Signature=AGO-
PgxxkYo99MkJFHvjfGXjG1QDEXeaDb4Qtzmy85wqyJjK7eKojoWa4BCRcow__&Policy=eyJTdGF0ZW1bnQiOlt7IlJlc291cmNlIj
Pair-Id=mykey"

To generate an RSA key pair, you can use openssl and the following commands:

# Generate a 1024bit key pair
openssl genrsa -out private-key.pem 1024
openssl rsa -in private-key.pem -pubout -out public-key.pem

=head1 OPTIONS

=over 8
```

```

=item B<--help>

Print a help message and exits.

=item B<--action> [action]

The action to execute. action can be one of:

    encode - Generate a signed URL (using a canned policy or a user policy)
    decode - Decode a signed URL

=item B<--url>

The URL to en/decode

=item B<--stream>

The stream to en/decode

=item B<--private-key>

The path to your private key.

=item B<--key-pair-id>

The AWS Portal assigned key pair identifier.

=item B<--policy>

The CloudFront policy document.

=item B<--expires>

The Unix epoch time when the URL is to expire. If both this option and
the --policy option are specified, --policy will be used. Otherwise, this
option alone will use a canned policy.

=back

=cut

use strict;
use warnings;

# you might need to use CPAN to get these modules.
# run perl -MCPAN -e "install <module>" to get them.
# The openssl command line will also need to be in your $PATH.
use File::Temp qw/tempfile/;
use Getopt::Long;
use IPC::Open2;
use MIME::Base64 qw(encode_base64 decode_base64);
use Pod::Usage;
use URI;

my $CANNED_POLICY
    = '{"Statement": [{"Resource": "<RESOURCE>", "Condition": {"DateLessThan": {"AWS:EpochTime": <EXPIRES>}}}]}'';

my $POLICY_PARAM      = "Policy";
my $EXPIRES_PARAM     = "Expires";
my $SIGNATURE_PARAM   = "Signature";
my $KEY_PAIR_ID_PARAM = "Key-Pair-Id";

my $verbose = 0;
my $policy_filename = "";
my $expires_epoch = 0;

```

```

my $action = "";
my $help = 0;
my $key_pair_id = "";
my $url = "";
my $stream = "";
my $private_key_filename = "";

my $result = GetOptions("action=s"      => \$action,
                       "policy=s"     => \$policy_filename,
                       "expires=i"   => \$expires_epoch,
                       "private-key=s"=> \$private_key_filename,
                       "key-pair-id=s"=> \$key_pair_id,
                       "verbose"      => \$verbose,
                       "help"         => \$help,
                       "url=s"        => \$url,
                       "stream=s"     => \$stream,
                      );
;

if ($help or !$result) {
    pod2usage(1);
    exit;
}

if ($url eq "" and $stream eq "") {
    print STDERR "Must include a stream or a URL to encode or decode with the --stream or
--url option\n";
    exit;
}

if ($url ne "" and $stream ne "") {
    print STDERR "Only one of --url and --stream may be specified\n";
    exit;
}

if ($url ne "" and !is_url_valid($url)) {
    exit;
}

if ($stream ne "") {
    exit unless is_stream_valid($stream);

    # The signing mechanism is identical, so from here on just pretend we're
    # dealing with a URL
    $url = $stream;
}

if ($action eq "encode") {
    # The encode action will generate a private content URL given a base URL,
    # a policy file (or an expires timestamp) and a key pair id parameter
    my $private_key;
    my $public_key;
    my $public_key_file;

    my $policy;
    if ($policy_filename eq "") {
        if ($expires_epoch == 0) {
            print STDERR "Must include policy filename with --policy argument or an
expires" .
                           "time using --expires\n";
        }
    }

    $policy = $CANNED_POLICY;
    $policy =~ s/<EXPIRES>/\$expires_epoch/g;
    $policy =~ s/<RESOURCE>/\$url/g;
} else {
    if (! -e $policy_filename) {

```

```

        print STDERR "Policy file $policy_filename does not exist\n";
        exit;
    }
    $expires_epoch = 0; # ignore if set
    $policy = read_file($policy_filename);
}

if ($private_key_filename eq "") {
    print STDERR "You must specific the path to your private key file with --private-
key\n";
    exit;
}

if (! -e $private_key_filename) {
    print STDERR "Private key file $private_key_filename does not exist\n";
    exit;
}

if ($key_pair_id eq "") {
    print STDERR "You must specify an AWS portal key pair id with --key-pair-id\n";
    exit;
}

my $encoded_policy = url_safe_base64_encode($policy);
my $signature = rsa_sha1_sign($policy, $private_key_filename);
my $encoded_signature = url_safe_base64_encode($signature);

my $generated_url = create_url($url, $encoded_policy, $encoded_signature, $key_pair_id,
$expires_epoch);

if ($stream ne "") {
    print "Encoded stream (for use within a swf):\n" . $generated_url . "\n";
    print "Encoded and escaped stream (for use on a webpage):\n" .
escape_url_for_webpage($generated_url) . "\n";
} else {
    print "Encoded URL:\n" . $generated_url . "\n";
}
} elsif ($action eq "decode") {
    my $decoded = decode_url($url);
    if (!$decoded) {
        print STDERR "Improperly formed URL\n";
        exit;
    }

    print_decoded_url($decoded);
} else {
    # No action specified, print help. But only if this is run as a program (caller will
    be empty)
    pod2usage(1) unless caller();
}

# Decode a private content URL into its component parts
sub decode_url {
    my $url = shift;

    if ($url =~ /(.*)\?(.*)/) {
        my $base_url = $1;
        my $params = $2;

        my @unparsed_params = split(/&, $params);
        my %params = ();
        foreach my $param (@unparsed_params) {
            my ($key, $val) = split(/\=/, $param);
            $params{$key} = $val;
        }
    }
}

```

```

my $encoded_signature = "";
if (exists $params{$SIGNATURE_PARAM}) {
    $encoded_signature = $params{"Signature"};
} else {
    print STDERR "Missing Signature URL parameter\n";
    return 0;
}

my $encoded_policy = "";
if (exists $params{$POLICY_PARAM}) {
    $encoded_policy = $params{$POLICY_PARAM};
} else {
    if (!exists $params{$EXPIRES_PARAM}) {
        print STDERR "Either the Policy or Expires URL parameter needs to be
specified\n";
        return 0;
    }

    my $expires = $params{$EXPIRES_PARAM};

    my $policy = $CANNED_POLICY;
    $policy =~ s/<EXPIRES>/\$expires/g;

    my $url_without_cf_params = $url;
    $url_without_cf_params =~ s/\$SIGNATURE_PARAM=[^&]*\$/g;
    $url_without_cf_params =~ s/\$POLICY_PARAM=[^&]*\$/g;
    $url_without_cf_params =~ s/\$EXPIRES_PARAM=[^&]*\$/g;
    $url_without_cf_params =~ s/\$KEY_PAIR_ID_PARAM=[^&]*\$/g;

    if ($url_without_cf_params =~ /(.*)\?$/ ) {
        $url_without_cf_params = $1;
    }

    $policy =~ s/<RESOURCE>/\$url_without_cf_params/g;

    $encoded_policy = url_safe_base64_encode($policy);
}

my $key = "";
if (exists $params{$KEY_PAIR_ID_PARAM}) {
    $key = $params{$KEY_PAIR_ID_PARAM};
} else {
    print STDERR "Missing \$KEY_PAIR_ID_PARAM parameter\n";
    return 0;
}

my $policy = url_safe_base64_decode($encoded_policy);

my %ret = ();
$ret{"base_url"} = $base_url;
$ret{"policy"} = $policy;
$ret{"key"} = $key;

return \%ret;
} else {
    return 0;
}
}

# Print a decoded URL out
sub print_decoded_url {
    my $decoded = shift;

    print "Base URL: \n" . $decoded->{"base_url"} . "\n";
    print "Policy: \n" . $decoded->{"policy"} . "\n";
}

```

```

        print "Key: \n" . $decoded->{"key"} . "\n";
    }

# Encode a string with base 64 encoding and replace some invalid URL characters
sub url_safe_base64_encode {
    my ($value) = @_;
    my $result = encode_base64($value);
    $result =~ tr|+=/|-_-|;
    return $result;
}

# Decode a string with base 64 encoding. URL-decode the string first
# followed by reversing any special character ("+=/") translation.
sub url_safe_base64_decode {
    my ($value) = @_;
    $value =~ s/%([0-9A-Fa-f]{2})/chr(hex($1))/eg;
    $value =~ tr|-_-|+=/|;

    my $result = decode_base64($value);

    return $result;
}

# Create a private content URL
sub create_url {
    my ($path, $policy, $signature, $key_pair_id, $expires) = @_;

    my $result;
    my $separator = $path =~ /\?/ ? '&' : '?';
    if ($expires) {
        $result = "$path$separator$EXPIRES_PARAM=$expires&$SIGNATURE_PARAM=$signature&$KEY_PAIR_ID_PARAM=$key_pair_id";
    } else {
        $result = "$path$separator$POLICY_PARAM=$policy&$SIGNATURE_PARAM=$signature&$KEY_PAIR_ID_PARAM=$key_pair_id";
    }
    $result =~ s/\n//g;

    return $result;
}

# Sign a document with given private key file.
# The first argument is the document to sign
# The second argument is the name of the private key file
sub rsa_sha1_sign {
    my ($to_sign, $pvkFile) = @_;

    return write_to_program("openssl sha1 -sign $pvkFile", $to_sign);
}

# Helper function to write data to a program
sub write_to_program {
    my ($prog, $data) = @_;

    my $pid = open2(*README, *WRITEME, $prog);
    print WRITEME $data;
    close WRITEME;

    # slurp entire contents of output into scalar
    my $output;
    local $/ = undef;
    $output = <README>;
    close README;
}

```

```

        waitpid($pid, 0);

        return $output;
    }

# Read a file into a string and return the string
sub read_file {
    my ($file) = @_;

    open(INFILE, "<$file") or die("Failed to open $file: $!");
    my $str = join('', <INFILE>);
    close INFILE;

    return $str;
}

sub is_url_valid {
    my ($url) = @_;

    # HTTP distributions start with http[s]:// and are the correct thing to sign
    if ($url =~ /^https?:\/\/\//) {
        return 1;
    } else {
        print STDERR "CloudFront requires absolute URLs for HTTP distributions\n";
        return 0;
    }
}

sub is_stream_valid {
    my ($stream) = @_;

    if ($stream =~ /^rtmp:\// or $stream =~ /^\/?cfx\/st/) {
        print STDERR "Streaming distributions require that only the stream name is signed.\n";
        print STDERR "The stream name is everything after, but not including, cfx/st/\n";
        return 0;
    } else {
        return 1;
    }
}

# flash requires that the query parameters in the stream name are url
# encoded when passed in through javascript, etc. This sub handles the minimal
# required url encoding.
sub escape_url_for_webpage {
    my ($url) = @_;

    $url =~ s/\/?/%3F/g;
    $url =~ s/=/%3D/g;
    $url =~ s/&/%26/g;

    return $url;
}

1;

```

Create a URL Signature Using PHP

Any web server that runs PHP can use the PHP demo code to create policy statements and signatures for private CloudFront RTMP distributions. The example creates a functioning web page with signed URL links that play a video stream using CloudFront streaming. To get the example, download [Signature Code for Video Streaming in PHP](#).

You can also create signed URLs by using the `UrlSigner` class in the AWS SDK for PHP. For more information, see [Class UrlSigner in the AWS SDK for PHP API Reference](#).

Note

Creating a URL signature is just one part of the process of serving private content using a signed URL. For more information about the entire process, see [Using Signed URLs \(p. 224\)](#).

In the following code segment, the function `rsa_sha1_sign` hashes and signs the policy statement. The arguments required are a policy statement, an `out` parameter to contain the signature, and the private key for your AWS account or for a trusted AWS account that you specify. Next, the `url_safe_base64_encode` function creates a URL-safe version of the signature.

Example RSA SHA1 Hashing in PHP

```
function rsa_sha1_sign($policy, $private_key_filename) {
    $signature = "";

    // load the private key
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);

    // compute signature
    openssl_sign($policy, $signature, $pkeyid);

    // free the key from memory
    openssl_free_key($pkeyid);

    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    // replace unsafe characters +, = and / with
    // the safe characters -, _ and ~
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
}
```

The following code constructs a *canned* policy statement needed for creating the signature. For more information about canned policies, see [Creating a Signed URL Using a Canned Policy \(p. 227\)](#).

Example Canned Signing Function in PHP

```
function get_canned_policy_stream_name($video_path, $private_key_filename, $key_pair_id,
$expires) {
    // this policy is well known by CloudFront, but you still need to sign it,
    // since it contains your parameters
    $canned_policy = '{"Statement": [{"Resource": "' . $video_path . '", "Condition": {"DateLessThan": {"AWS:EpochTime": "' . $expires . '"}}}]}'';

    // sign the canned policy
    $signature = rsa_sha1_sign($canned_policy, $private_key_filename);
    // make the signature safe to be included in a url
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, null, $encoded_signature, $key_pair_id,
$expires);
    // url-encode the query string characters to work around a flash player bug
```

```
    return encode_query_params($stream_name);
}
```

The following code constructs a *custom* policy statement needed for creating the signature. For more information about custom policies, see [Creating a Signed URL Using a Custom Policy \(p. 234\)](#).

Note

The \$expires variable is a date/time stamp that must be an integer, not a string.

Example Custom Signing Function in PHP

```
function get_custom_policy_stream_name($video_path, $private_key_filename, $key_pair_id,
$policy) {
    // sign the policy
    $signature = rsa_sha1_sign($policy, $private_key_filename);
    // make the signature safe to be included in a url
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, $encoded_policy, $encoded_signature,
$key_pair_id, null);
    // url-encode the query string characters to work around a flash player bug
    return encode_query_params($stream_name);
}
```

For more information about the OpenSSL implementation of SHA-1, see [The Open Source Toolkit for SSL/TLS](#).

See also

- [Create a URL Signature Using Perl \(p. 260\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 270\)](#)
- [Create a URL Signature Using Java \(p. 276\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)

Create a URL Signature Using C# and the .NET Framework

The C# examples in this section implement an example application that demonstrates how to create the signatures for CloudFront private distributions using canned and custom policy statements. The examples includes utility functions based on the [AWS .NET SDK](#) that can be useful in .NET applications.

You can also create signed URLs and signed cookies by using the AWS SDK for .NET. In the [AWS SDK for .NET API Reference](#), see the following topics:

- **Signed URLs** – Amazon.CloudFront > AmazonCloudFrontUrlSigner
- **Signed cookies** – Amazon.CloudFront > AmazonCloudFrontCookieSigner

Note

Creating a URL signature is just one part of the process of serving private content using a signed URL. For more information about the entire process, see [Using Signed URLs \(p. 224\)](#).

To download the code, go to [Signature Code in C#](#).

To use the RSA keys provided by [AWS Account/Security](#) in the .NET framework, you must convert the AWS-supplied .pem files to the XML format that the .NET framework uses.

After conversion, the RSA private key file is in the following format:

Example RSA Private Key in the XML .NET Framework Format

```
<RSAKeyValue>
  <Modulus>
    wO5IvYCP5UcoCKDo1dcspoMehWBZcyfs9QEzGi60e5y+ewGr1oW+vB2GPB
    ANBiVPCUHTFWhwaiBd3oglmF01GQ1jP/jOfmXHUK2kUUlnJp+oOBL2NiufTqcW6h/L5lIpD8Yq+NRHg
    Ty4zDsyR2880MvXv88yEFURCkqEXAMPLE=
  </Modulus>
  <Exponent>AQAB</Exponent>
  <P>
    5bmKDaTz
    npENGVqz4Cea8XPH+sxt+2VaAwYnsarVUoSBeVt8WLloVuZGG9IZYmH5KtexEu7fZveYd9UEXAMPLE==
  </P>
  <Q>
    1v91/WN1a1N3rOK4VGOCokx7kR2SyTMSbZgF9IWJNougR/WZw7HTnjipO3c9dy1Ms9pUKwUF4
    6d7049EXAMPLE==
  </Q>
  <DP>
    RgrSKuLWXMyBH+/l1Dx/I4tXuAJIrlPyo+Vmi0c7b5NzHptkSHEPfR9s1
    OK0Vqjknc1qCJ3Ig86OMEtEXAMPLE==
  </DP>
  <DQ>
    pjPjvSFw+RoaTu0pgCA/jwW/FGyfN6iim1RFbkT4
    z49DZb2IM885f3vf35eLTaEYRYUHQgZtChNEV0TEXAMPLE==
  </DQ>
  <InverseQ>
    nkvoJTG5QtGNgWb9i
    CVtzrL/1pFEOHbJXwEJdU99N+7sMK+1066DL/HSBUCD63qD4USpnf0myc24in0EXAMPLE==</InverseQ>
  <D>
    Bc7mp7XYHyuNPzxChjWNJZIq+A73gm0ASDv6At7F8Vi9r0xUlQe/v0AQs3ycN8QlyR4XMbzMlyk
    3yjxFDXo4ZKQtOGzLGteCU2srANiLv26/imXA8FVidZftTAtLviWQZBVPtEYIA69ATUYPEq0a5u5wjGy
    UOij90WyuEXAMPLE=
  </D>
</RSAKeyValue>
```

The following C# code creates a signed URL that uses a canned policy by performing the following steps:

- Creates a policy statement.
- Hashes the policy statement using SHA1, and signs the result using RSA and the private key for your AWS account or for a trusted AWS account that you specify.
- Base64-encodes the hashed and signed policy statement and replaces special characters to make the string safe to use as a URL request parameter.
- Concatenates the applicable values.

For the complete implementation, see the example at [Signature Code in C#](#).

Example Canned Policy Signing Method in C#

```
public static string ToUrlSafeBase64String(byte[] bytes)
{
    return System.Convert.ToBase64String(bytes)
        .Replace('+', '-')
        .Replace('=', '_')
        .Replace('/', '-');
}

public static string CreateCannedPrivateURL(string urlString,
    string durationUnits, string durationNumber, string pathToPolicyStmnt,
```

```

        string pathToPrivateKey, string privateKeyId)
{
    // args[] 0-thisMethod, 1-resourceUrl, 2-seconds-minutes-hours-days
    // to expiration, 3-numberOfPreviousUnits, 4-pathToPolicyStmnt,
    // 5-pathToPrivateKey, 6-PrivateKeyId

    TimeSpan timeSpanInterval = GetDuration(durationUnits, durationNumber);

    // Create the policy statement.
    string strPolicy = CreatePolicyStatement(pathToPolicyStmnt,
        urlString,
        DateTime.Now,
        DateTime.Now.Add(timeSpanInterval),
        "0.0.0.0/0");
    if ("Error!" == strPolicy) return "Invalid time frame." +
        "Start time cannot be greater than end time.";

    // Copy the expiration time defined by policy statement.
    string strExpiration = CopyExpirationTimeFromPolicy(strPolicy);

    // Read the policy into a byte buffer.
    byte[] bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);

    // Initialize the SHA1CryptoServiceProvider object and hash the policy data.
    using (SHA1CryptoServiceProvider
        cryptoSHA1 = new SHA1CryptoServiceProvider())
    {
        bufferPolicy = cryptoSHA1.ComputeHash(bufferPolicy);

        // Initialize the RSACryptoServiceProvider object.
        RSACryptoServiceProvider providerRSA = new RSACryptoServiceProvider();
        XmlDocument xmlPrivateKey = new XmlDocument();

        // Load PrivateKey.xml, which you created by converting your
        // .pem file to the XML format that the .NET framework uses.
        // Several tools are available.
        xmlPrivateKey.Load(pathToPrivateKey);

        // Format the RSACryptoServiceProvider providerRSA and
        // create the signature.
        providerRSA.FromXmlString(xmlPrivateKey.InnerXml);
        RSAPKCS1SignatureFormatter rsaFormatter =
            new RSAPKCS1SignatureFormatter(providerRSA);
        rsaFormatter.SetHashAlgorithm("SHA1");
        byte[] signedPolicyHash = rsaFormatter.CreateSignature(bufferPolicy);

        // Convert the signed policy to URL-safe base64 encoding and
        // replace unsafe characters + / with the safe characters - _ ~
        string strSignedPolicy = ToUrlSafeBase64String(signedPolicyHash);

        // Concatenate the URL, the timestamp, the signature,
        // and the key pair ID to form the signed URL.
        return urlString +
            "?Expires=" +
            strExpiration +
            "&Signature=" +
            strSignedPolicy +
            "&Key-Pair-Id=" +
            privateKeyId;
    }
}

```

The following C# code creates a signed URL that uses a custom policy by performing the following steps:

1. Creates a policy statement.

2. Base64-encodes the policy statement and replaces special characters to make the string safe to use as a URL request parameter.
3. Hashes the policy statement using SHA1, and encrypts the result using RSA and the private key for your AWS account or for a trusted AWS account that you specify.
4. Base64-encodes the hashed policy statement and replacing special characters to make the string safe to use as a URL request parameter.
5. Concatenates the applicable values.

For the complete implementation, see the example at [Signature Code in C#](#).

Example Custom Policy Signing Method in C#

```

public static string ToUrlSafeBase64String(byte[] bytes)
{
    return System.Convert.ToBase64String(bytes)
        .Replace('+', '-')
        .Replace('=', '_')
        .Replace('/', '-');
}

public static string CreateCustomPrivateURL(string urlString,
    string durationUnits, string durationNumber, string startIntervalFromNow,
    string ipAddress, string pathToPolicyStmnt, string pathToPrivateKey,
    string privateKeyId)
{
    // args[] 0-thisMethod, 1-resourceUrl, 2-seconds-minutes-hours-days
    // to expiration, 3-numberOfPreviousUnits, 4-starttimeFromNow,
    // 5-ip_address, 6-pathToPolicyStmt, 7-pathToPrivateKey, 8-privateKeyId

    TimeSpan timeSpanInterval = GetDuration(durationUnits, durationNumber);
    TimeSpan timeSpanToStart = GetDurationByUnits(durationUnits,
        startIntervalFromNow);
    if (null == timeSpanToStart)
        return "Invalid duration units." +
            "Valid options: seconds, minutes, hours, or days";

    string strPolicy = CreatePolicyStatement(
        pathToPolicyStmnt, urlString, DateTime.Now.Add(timeSpanToStart),
        DateTime.Now.Add(timeSpanInterval), ipAddress);

    // Read the policy into a byte buffer.
    byte[] bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);

    // Convert the policy statement to URL-safe base64 encoding and
    // replace unsafe characters + = / with the safe characters - _ ~

    string urlSafePolicy = ToUrlSafeBase64String(bufferPolicy);

    // Initialize the SHA1CryptoServiceProvider object and hash the policy data.
    byte[] bufferPolicyHash;
    using (SHA1CryptoServiceProvider cryptoSHA1 =
        new SHA1CryptoServiceProvider())
    {
        bufferPolicyHash = cryptoSHA1.ComputeHash(bufferPolicy);

        // Initialize the RSACryptoServiceProvider object.
        RSACryptoServiceProvider providerRSA = new RSACryptoServiceProvider();
        XmlDocument xmlPrivateKey = new XmlDocument();

        // Load PrivateKey.xml, which you created by converting your
        // .pem file to the XML format that the .NET framework uses.
    }
}

```

```

// Several tools are available.
xmlPrivateKey.Load("PrivateKey.xml");

// Format the RSACryptoServiceProvider providerRSA
// and create the signature.
providerRSA.FromXmlString(xmlPrivateKey.InnerXml);
RSAPKCS1SignatureFormatter RSAFormatter =
    new RSAPKCS1SignatureFormatter(providerRSA);
RSAFormatter.SetHashAlgorithm("SHA1");
byte[] signedHash = RSAFormatter.CreateSignature(bufferPolicyHash);

// Convert the signed policy to URL-safe base64 encoding and
// replace unsafe characters + = / with the safe characters - _ ~
string strSignedPolicy = ToUrlSafeBase64String(signedHash);

return urlString +
    "?Policy=" +
    urlSafePolicy +
    "&Signature=" +
    strSignedPolicy +
    "&Key-Pair-Id=" +
    PrivateKeyId;
}
}

```

Example Utility Methods for Signature Generation

The following methods get the policy statement from a file and parse time intervals for signature generation.

```

public static string CreatePolicyStatement(string policyStmnt,
    string resourceUrl,
    DateTime startTime,
    DateTime endTime,
    string ipAddress)

{
    // Create the policy statement.
    FileStream streamPolicy = new FileStream(policyStmnt, FileMode.Open, FileAccess.Read);
    using (StreamReader reader = new StreamReader(streamPolicy))
    {
        string strPolicy = reader.ReadToEnd();

        TimeSpan startTimeSpanFromNow = (startTime - DateTime.Now);
        TimeSpan endTimeSpanFromNow = (endTime - DateTime.Now);
        TimeSpan intervalStart =
            (DateTime.UtcNow.Add(startTimeSpanFromNow)) -
            new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
        TimeSpan intervalEnd =
            (DateTime.UtcNow.Add(endTimeSpanFromNow)) -
            new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);

        int startTimestamp = (int)intervalStart.TotalSeconds; // START_TIME
        int endTimestamp = (int)intervalEnd.TotalSeconds; // END_TIME

        if (startTimestamp > endTimestamp)
            return "Error!";

        // Replace variables in the policy statement.
        strPolicy = strPolicy.Replace("RESOURCE", resourceUrl);
        strPolicy = strPolicy.Replace("START_TIME", startTimestamp.ToString());
        strPolicy = strPolicy.Replace("END_TIME", endTimestamp.ToString());
        strPolicy = strPolicy.Replace("IP_ADDRESS", ipAddress);
        strPolicy = strPolicy.Replace("EXPIRES", endTimestamp.ToString());
    }
}

```

```

        return strPolicy;
    }

public static TimeSpan GetDuration(string units, string numUnits)
{
    TimeSpan timeSpanInterval = new TimeSpan();
    switch (units)
    {
        case "seconds":
            timeSpanInterval = new TimeSpan(0, 0, 0, int.Parse(numUnits));
            break;
        case "minutes":
            timeSpanInterval = new TimeSpan(0, 0, int.Parse(numUnits), 0);
            break;
        case "hours":
            timeSpanInterval = new TimeSpan(0, int.Parse(numUnits), 0, 0);
            break;
        case "days":
            timeSpanInterval = new TimeSpan(int.Parse(numUnits), 0, 0, 0);
            break;
        default:
            Console.WriteLine("Invalid time units;" +
                "use seconds, minutes, hours, or days");
            break;
    }
    return timeSpanInterval;
}

private static TimeSpan GetDurationByUnits(string durationUnits,
    string startIntervalFromNow)
{
    switch (durationUnits)
    {
        case "seconds":
            return new TimeSpan(0, 0, int.Parse(startIntervalFromNow));
        case "minutes":
            return new TimeSpan(0, int.Parse(startIntervalFromNow), 0);
        case "hours":
            return new TimeSpan(int.Parse(startIntervalFromNow), 0, 0);
        case "days":
            return new TimeSpan(int.Parse(startIntervalFromNow), 0, 0, 0);
        default:
            return new TimeSpan(0, 0, 0, 0);
    }
}

public static string CopyExpirationTimeFromPolicy(string policyStatement)
{
    int startExpiration = policyStatement.IndexOf("EpochTime");
    string strExpirationRough = policyStatement.Substring(startExpiration +
        "EpochTime".Length);
    char[] digits = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' };

    List<char> listDigits = new List<char>(digits);
    StringBuilder buildExpiration = new StringBuilder(20);

    foreach (char c in strExpirationRough)
    {
        if (listDigits.Contains(c))
            buildExpiration.Append(c);
    }
    return buildExpiration.ToString();
}

```

See also

- [Create a URL Signature Using Perl \(p. 260\)](#)
- [Create a URL Signature Using PHP \(p. 268\)](#)
- [Create a URL Signature Using Java \(p. 276\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)

Create a URL Signature Using Java

The [Open source Java toolkit for Amazon S3 and CloudFront](#) provides example code and information about CloudFront development in Java. For information about private distributions, go to [Private Distributions](#) at [Programmer Guide: Code Samples](#).

You can also create signed URLs by using the `CloudFrontUrlSigner` class in the AWS SDK for Java. For more information, see [Class UrlSigner](#) in the [AWS SDK for Java API Reference](#).

Note

Creating a URL signature is just one part of the process of serving private content using a signed URL. For more information about the entire process, see [Using Signed URLs \(p. 224\)](#).

The following methods are from the Java open source toolkit for Amazon S3 and CloudFront. You must convert the private key from PEM to DER format for Java implementations to use it.

Example Java Policy and Signature Encryption Methods

```
// Signed URLs for a private distribution
// Note that Java only supports SSL certificates in DER format,
// so you will need to convert your PEM-formatted file to DER format.
// To do this, you can use openssl:
// openssl pkcs8 -topk8 -nocrypt -in origin.pem -inform PEM -out new.der
//   -outform DER
// So the encoder works correctly, you should also add the bouncy castle jar
// to your project and then add the provider.

Security.addProvider(new org.bouncycastle.jce.provider.BouncyCastleProvider());

String distributionDomain = "a1b2c3d4e5f6g7.cloudfront.net";
String privateKeyFilePath = "/path/to/rsa-private-key.der";
String s3ObjectKey = "s3/object/key.txt";
String policyResourcePath = "http://" + distributionDomain + "/" + s3ObjectKey;

// Convert your DER file into a byte array.

byte[] derPrivateKey = ServiceUtils.readInputStreamToBytes(new
    FileInputStream(privateKeyFilePath));

// Generate a "canned" signed URL to allow access to a
// specific distribution and object

String signedUrlCanned = CloudFrontService.signUrlCanned(
    "http://" + distributionDomain + "/" + s3ObjectKey, // Resource URL or Path
    keyPairId,      // Certificate identifier,
                    // an active trusted signer for the distribution
    derPrivateKey, // DER Private key data
    ServiceUtils.parseIso8601Date("2011-11-14T22:20:00.000Z") // DateLessThan
);
System.out.println(signedUrlCanned);

// Build a policy document to define custom restrictions for a signed URL.
```

```
String policy = CloudFrontService.buildPolicyForSignedUrl(  
    // Resource path (optional, can include '*' and '?' wildcards)  
    policyResourcePath,  
    // DateLessThan  
    ServiceUtils.parseIso8601Date("2011-11-14T22:20:00.000Z"),  
    // CIDR IP address restriction (optional, 0.0.0.0/0 means everyone)  
    "0.0.0.0/0",  
    // DateGreaterThanOr (optional)  
    ServiceUtils.parseIso8601Date("2011-10-16T06:31:56.000Z")  
);  
  
// Generate a signed URL using a custom policy document.  
  
String signedUrl = CloudFrontService.signUrl(  
    // Resource URL or Path  
    "http://" + distributionDomain + "/" + s3ObjectKey,  
    // Certificate identifier, an active trusted signer for the distribution  
    keyPairId,  
    // DER Private key data  
    derPrivateKey,  
    // Access control policy  
    policy  
);  
System.out.println(signedUrl);
```

See also

- [Create a URL Signature Using Perl \(p. 260\)](#)
- [Create a URL Signature Using PHP \(p. 268\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 270\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)

Using HTTPS with CloudFront

For web distributions, you can configure CloudFront to require that viewers use HTTPS to request your objects, so connections are encrypted when CloudFront communicates with viewers. You also can configure CloudFront to use HTTPS to get objects from your origin, so connections are encrypted when CloudFront communicates with your origin.

If you configure CloudFront to require HTTPS both to communicate with viewers and to communicate with your origin, here's what happens when CloudFront receives a request for an object. The process works basically the same way whether your origin is an Amazon S3 bucket or a custom origin such as an HTTP/S server:

1. A viewer submits an HTTPS request to CloudFront. There's some SSL/TLS negotiation here between the viewer and CloudFront. In the end, the viewer submits the request in an encrypted format.
2. If the object is in the CloudFront edge cache, CloudFront encrypts the response and returns it to the viewer, and the viewer decrypts it.
3. If the object is not in the CloudFront cache, CloudFront performs SSL/TLS negotiation with your origin and, when the negotiation is complete, forwards the request to your origin in an encrypted format.
4. Your origin decrypts the request, encrypts the requested object, and returns the object to CloudFront.
5. CloudFront decrypts the response, re-encrypts it, and forwards the object to the viewer. CloudFront also saves the object in the edge cache so that the object is available the next time it's requested.
6. The viewer decrypts the response.

For information about how to require HTTPS between viewers and CloudFront, and between CloudFront and your origin, see the following topics.

Topics

- [Requiring HTTPS for Communication Between Viewers and CloudFront \(p. 278\)](#)
- [Requiring HTTPS for Communication Between CloudFront and Your Custom Origin \(p. 280\)](#)
- [Requiring HTTPS for Communication Between CloudFront and Your Amazon S3 Origin \(p. 282\)](#)
- [Using Alternate Domain Names and HTTPS \(p. 283\)](#)
- [Supported Protocols and Ciphers \(p. 296\)](#)
- [Charges for HTTPS Connections \(p. 299\)](#)

Requiring HTTPS for Communication Between Viewers and CloudFront

You can configure one or more cache behaviors in your CloudFront distribution to require HTTPS for communication between viewers and CloudFront. You also can configure one or more cache behaviors to allow both HTTP and HTTPS, so that CloudFront requires HTTPS for some objects but not for others. The configuration steps depend on which domain name you're using in object URLs:

- If you're using the domain name that CloudFront assigned to your distribution, such as d111111abcdef8.cloudfront.net, you change the **Viewer Protocol Policy** setting for one or more cache

behaviors to require HTTPS communication. In that configuration, CloudFront provides the SSL/TLS certificate.

To change the value of **Viewer Protocol Policy** by using the CloudFront console, see the procedure later in this section.

For information about how to use the CloudFront API to change the value of the `ViewerProtocolPolicy` element, see [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

- If you're using your own domain name, such as example.com, you need to change several CloudFront settings. You also need to use an SSL/TLS certificate provided by AWS Certificate Manager (ACM), import a certificate from a third-party certificate authority into ACM or the IAM certificate store, or create and import a self-signed certificate. For more information, see [Using Alternate Domain Names and HTTPS \(p. 283\)](#).

Note

If you want to ensure that the objects that viewers get from CloudFront were encrypted when CloudFront got them from your origin, always use HTTPS between CloudFront and your origin. If you recently changed from HTTP to HTTPS between CloudFront and your origin, we recommend that you invalidate objects in CloudFront edge locations. CloudFront will return an object to a viewer regardless of whether the protocol used by the viewer (HTTP or HTTPS) matches the protocol that CloudFront used to get the object. For more information about removing or replacing objects in a distribution, see [Adding, Removing, or Replacing Objects in a Distribution \(p. 158\)](#).

To require HTTPS between viewers and CloudFront for one or more cache behaviors, perform the following procedure.

To configure CloudFront to require HTTPS between viewers and CloudFront

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the top pane of the CloudFront console, choose the ID for the distribution that you want to update.
3. On the **Behaviors** tab, choose the cache behavior that you want to update, and then choose **Edit**.
4. Specify one of the following values for **Viewer Protocol Policy**:

Redirect HTTP to HTTPS

Viewers can use both protocols. HTTP GET and HEAD requests are automatically redirected to HTTPS requests. CloudFront returns HTTP status code 301 (Moved Permanently) along with the new HTTPS URL. The viewer then resubmits the request to CloudFront using the HTTPS URL.

Important

If you send POST, PUT, DELETE, OPTIONS, or PATCH over HTTP with an HTTP to HTTPS cache behavior and a request protocol version of HTTP 1.1 or above, CloudFront redirects the request to a HTTPS location with a HTTP status code 307 (Temporary Redirect). This guarantees that the request is sent again to the new location using the same method and body payload.

If you send POST, PUT, DELETE, OPTIONS, or PATCH requests over HTTP to HTTPS cache behavior with a request protocol version below HTTP 1.1, CloudFront returns a HTTP status code 403 (Forbidden).

When a viewer makes an HTTP request that is redirected to an HTTPS request, CloudFront charges for both requests. For the HTTP request, the charge is only for the request and for the headers that CloudFront returns to the viewer. For the HTTPS request, the charge is for the request, and for the headers and the object that are returned by your origin.

HTTPS Only

Viewers can access your content only if they're using HTTPS. If a viewer sends an HTTP request instead of an HTTPS request, CloudFront returns HTTP status code 403 (Forbidden) and does not return the object.

5. Choose **Yes, Edit**.
6. Repeat steps 3 through 5 for each additional cache behavior that you want to require HTTPS for between viewers and CloudFront.
7. Confirm the following before you use the updated configuration in a production environment:
 - The path pattern in each cache behavior applies only to the requests that you want viewers to use HTTPS for.
 - The cache behaviors are listed in the order that you want CloudFront to evaluate them in. For more information, see [Path Pattern \(p. 71\)](#).
 - The cache behaviors are routing requests to the correct origins.

Requiring HTTPS for Communication Between CloudFront and Your Custom Origin

If you're using an Amazon S3 bucket as your origin, see [Requiring HTTPS for Communication Between CloudFront and Your Amazon S3 Origin \(p. 282\)](#).

If you want to require HTTPS for communication between CloudFront and your custom origin, and you're using the domain name that CloudFront assigned to your distribution in the URLs for your objects (for example, <https://d111111abcdef8.cloudfront.net/logo.jpg>), perform the following procedures to change the **Origin Protocol Policy** setting for the applicable origins in your distribution, and install an SSL/TLS certificate on your custom origin server.

Changing CloudFront Settings

The following procedure explains how to configure CloudFront to use HTTPS to communicate with an Elastic Load Balancing load balancer, an Amazon EC2 instance, or another custom origin. For information about using the CloudFront API to update a web distribution, see [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

To configure CloudFront to require HTTPS between CloudFront and your custom origin

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the top pane of the CloudFront console, choose the ID for the distribution that you want to update.
3. On the **Origins** tab, choose the origin that you want to update, and then choose **Edit**.
4. Update the following settings:

Origin Protocol Policy

Change the **Origin Protocol Policy** for the applicable origins in your distribution:

- **HTTPS Only** – CloudFront uses only HTTPS to communicate with your custom origin.
- **Match Viewer** – CloudFront communicates with your custom origin using HTTP or HTTPS, depending on the protocol of the viewer request. For example, if you choose **Match Viewer** for **Origin Protocol Policy** and the viewer uses HTTPS to request an object from CloudFront, CloudFront also uses HTTPS to forward the request to your origin.

Choose **Match Viewer** only if you specify **Redirect HTTP to HTTPS** or **HTTPS Only** for **Viewer Protocol Policy**.

CloudFront caches the object only once even if viewers make requests using both HTTP and HTTPS protocols.

Origin SSL Protocols

Choose the **Origin SSL Protocols** for the applicable origins in your distribution. The SSLv3 protocol is less secure, so we recommend that you choose SSLv3 only if your origin doesn't support TLSv1 or later.

Note

The TLSv1 handshake is both backwards and forwards compatible with SSLv3, but TLSv1.1 and TLSv1.2 are not. In this case, the openssl only sends a SSLv3 handshake.

5. Choose **Yes, Edit**.
6. Repeat steps 3 through 5 for each additional origin that you want to require HTTPS for between CloudFront and your custom origin.
7. Confirm the following before you use the updated configuration in a production environment:
 - The path pattern in each cache behavior applies only to the requests that you want viewers to use HTTPS for.
 - The cache behaviors are listed in the order that you want CloudFront to evaluate them in. For more information, see [Path Pattern \(p. 71\)](#).
 - The cache behaviors are routing requests to the origins that you changed the **Origin Protocol Policy** for.

Installing an SSL/TLS Certificate on Your Custom Origin Server

You can use an SSL/TLS certificate from the following sources on your custom origin:

- If your origin is an Elastic Load Balancing load balancer, you can use a certificate provided by AWS Certificate Manager (ACM). You also can use a certificate that is signed by a trusted third-party certificate authority and imported into ACM.
- For origins other than ELB load balancers, you must use a certificate that is signed by a trusted third-party certificate authority, for example, Comodo, DigiCert, or Symantec.

When CloudFront uses HTTPS to communicate with your origin, CloudFront verifies that the certificate was issued by a trusted certificate authority. CloudFront supports the same certificate authorities as Mozilla; for the current list, see [Mozilla Included CA Certificate List](#). You can't use a self-signed certificate for HTTPS communication between CloudFront and your origin.

Important

If the origin server returns an expired certificate, an invalid certificate, or a self-signed certificate, or if the origin server returns the certificate chain in the wrong order, CloudFront drops the TCP connection, returns HTTP status code 502 (Bad Gateway), and sets the x-Cache header to `Error` from `cloudfront`. Also, if the full chain of certificates, including the intermediate certificate, is not present, CloudFront drops the TCP connection.

One of the domain names in the certificate must match one or both of the following values:

- The value that you specified for **Origin Domain Name** for the applicable origin in your distribution.

- If you configured CloudFront to forward the Host header to your origin, the value of the Host header. For more information about forwarding headers to your origin, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

Requiring HTTPS for Communication Between CloudFront and Your Amazon S3 Origin

When your origin is an Amazon S3 bucket, CloudFront always forwards requests to S3 by using the protocol that viewers used to submit the requests. The default setting for the [Origin Protocol Policy \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 68\)](#) setting is **Match Viewer** and can't be changed.

If you want to require HTTPS for communication between CloudFront and Amazon S3, you must change the value of **Viewer Protocol Policy** to **Redirect HTTP to HTTPS** or **HTTPS Only**. The procedure later in this section explains how to use the CloudFront console to change **Viewer Protocol Policy**. For information about using the CloudFront API to update the `ViewerProtocolPolicy` element for a web distribution, see [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

Note the following about using HTTPS when the origin is an Amazon S3 bucket:

- If your Amazon S3 bucket is configured as a website endpoint, you can't configure CloudFront to use HTTPS to communicate with your origin because Amazon S3 doesn't support HTTPS connections in that configuration.
- Amazon S3 provides the SSL/TLS certificate, so you don't have to.

To configure CloudFront to require HTTPS between CloudFront and your Amazon S3 origin

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the top pane of the CloudFront console, choose the ID for the distribution that you want to update.
3. On the **Behaviors** tab, choose the cache behavior that you want to update, and then choose **Edit**.
4. Specify one of the following values for **Viewer Protocol Policy**:

Redirect HTTP to HTTPS

Viewers can use both protocols, but HTTP requests are automatically redirected to HTTPS requests. CloudFront returns HTTP status code 301 (Moved Permanently) along with the new HTTPS URL. The viewer then resubmits the request to CloudFront using the HTTPS URL.

Important

CloudFront doesn't redirect `DELETE`, `OPTIONS`, `PATCH`, `POST`, or `PUT` requests from HTTP to HTTPS. If you configure a cache behavior to redirect to HTTPS, CloudFront responds to HTTP `DELETE`, `OPTIONS`, `PATCH`, `POST`, or `PUT` requests for that cache behavior with HTTP status code 403 (Forbidden).

When a viewer makes an HTTP request that is redirected to an HTTPS request, CloudFront charges for both requests. For the HTTP request, the charge is only for the request and for the headers that CloudFront returns to the viewer. For the HTTPS request, the charge is for the request, and for the headers and the object returned by your origin.

HTTPS Only

Viewers can access your content only if they're using HTTPS. If a viewer sends an HTTP request instead of an HTTPS request, CloudFront returns HTTP status code 403 (Forbidden) and does not return the object.

5. Choose **Yes, Edit**.
6. Repeat steps 3 through 5 for each additional cache behavior that you want to require HTTPS for between viewers and CloudFront, and between CloudFront and S3.
7. Confirm the following before you use the updated configuration in a production environment:
 - The path pattern in each cache behavior applies only to the requests that you want viewers to use HTTPS for.
 - The cache behaviors are listed in the order that you want CloudFront to evaluate them in. For more information, see [Path Pattern \(p. 71\)](#).
 - The cache behaviors are routing requests to the correct origins.

Using Alternate Domain Names and HTTPS

By default, you can deliver your content to viewers over HTTPS by using your CloudFront distribution domain name in your URLs, for example, `https://d111111abcdef8.cloudfront.net/image.jpg`. For more information, see [Requiring HTTPS for Communication Between Viewers and CloudFront \(p. 278\)](#).

If you want your viewers to use HTTPS and you want to use your own domain name in the URLs for your objects (for example, `https://www.example.com/image.jpg`), you need to perform several additional steps, as explained in this topic.

Important

When you add a certificate to your distribution, CloudFront immediately propagates the certificate to all of its edge locations. As new edge locations become available, CloudFront will propagate the certificate to those locations, too. You can't restrict the edge locations that CloudFront propagates the certificates to.

Topics

- [Choosing How CloudFront Serves HTTPS Requests \(p. 283\)](#)
- [Requirements for Using SSL/TLS Certificates with CloudFront \(p. 285\)](#)
- [Limits on Using SSL/TLS Certificates with CloudFront \(HTTPS Between Viewers and CloudFront Only\) \(p. 288\)](#)
- [Configuring Alternate Domain Names and HTTPS \(p. 289\)](#)
- [Determining the Size of the Public Key in an SSL/TLS Certificate \(p. 292\)](#)
- [Increasing the Limit for SSL/TLS Certificates \(p. 293\)](#)
- [Rotating SSL/TLS Certificates \(p. 294\)](#)
- [Reverting from a Custom SSL/TLS Certificate to the Default CloudFront Certificate \(p. 295\)](#)
- [Switching from a Custom SSL/TLS Certificate with Dedicated IP Addresses to SNI \(p. 295\)](#)

Choosing How CloudFront Serves HTTPS Requests

If you want your viewers to use HTTPS and to use alternate domain names for your objects, you need to choose how CloudFront serves HTTPS requests. When you configure CloudFront to use alternate domain names, CloudFront can serve HTTPS requests either by using a dedicated IP address in each edge location or by using Server Name Indication (SNI). This section explains how each option works.

Serving HTTPS Requests Using Dedicated IP Addresses (Works for All Clients)

If you configure CloudFront to serve HTTPS requests using dedicated IP addresses, CloudFront associates your alternate domain name with a dedicated IP address in each CloudFront edge location. When a viewer submits an HTTPS request for your content, here's what happens:

1. DNS routes the request to the IP address for your distribution in the applicable edge location.
2. CloudFront uses the IP address to identify your distribution and to determine which SSL/TLS certificate to return to the viewer.
3. The viewer and CloudFront perform SSL/TLS negotiation using your SSL/TLS certificate.
4. CloudFront returns the requested content to the viewer.

This method works for every HTTPS request, regardless of the browser or other viewer that the user is using.

Important

If you configure CloudFront to serve HTTPS requests using dedicated IP addresses, you incur an additional monthly charge. The charge begins when you associate your SSL/TLS certificate with a distribution and you enable the distribution. For more information about CloudFront pricing, see [Amazon CloudFront Pricing](#). In addition, see [Using the Same Certificate for Multiple CloudFront Distributions](#) (p. 289).

Serving HTTPS Requests Using SNI (Works for Most Clients)

If you configure CloudFront to serve HTTPS requests using Server Name Indication (SNI), CloudFront associates your alternate domain name with an IP address for each edge location, but the IP address is not dedicated to your distribution. When a viewer submits an HTTPS request for your content, DNS routes the request to the IP address for the applicable edge location. However, because the IP address isn't dedicated to your distribution, CloudFront can't determine, based on the IP address, which domain the request is for.

SSL/TLS negotiation occurs very early in the process of establishing an HTTPS connection. If CloudFront can't immediately determine which domain the request is for, it drops the connection. Using a dedicated IP address is one way to associate a request with a domain. The other is SNI, which is an extension to the TLS protocol that is supported by most modern browsers. When a viewer that supports SNI submits an HTTPS request for your content, here's what happens:

1. The viewer automatically gets the domain name from the request URL and adds it to a field in the request header.
2. When CloudFront receives the request, it finds the domain name in the request header and responds to the request with the applicable SSL/TLS certificate.
3. The viewer and CloudFront perform SSL/TLS negotiation.
4. CloudFront returns the requested content to the viewer.

For a current list of the browsers that support SNI, see the Wikipedia entry [Server Name Indication](#).

If you want to use SNI but some of your users' browsers don't support SNI, you have several options:

- Configure CloudFront to serve HTTPS requests by using dedicated IP addresses instead of SNI.
- Use the CloudFront SSL/TLS certificate instead of a custom certificate. This requires that you use the CloudFront domain name for your distribution in the URLs for your objects, for example, `https://d11111abcdef8.cloudfront.net/logo.png`.

If you use the default CloudFront certificate, viewers must support the SSL protocol TLSv1 or later. CloudFront doesn't support SSLv3 with the default CloudFront certificate.

You also need to change the SSL/TLS certificate that CloudFront is using from a custom certificate to the default CloudFront certificate:

- If you haven't used your distribution to distribute your content, you can just change the configuration. For more information, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#).
- If you have used your distribution to distribute your content, you need to create a new CloudFront distribution and change the URLs for your objects to reduce or eliminate the amount of time that your content is unavailable. For more information, see [Reverting from a Custom SSL/TLS Certificate to the Default CloudFront Certificate \(p. 295\)](#).
- If you can control which browser your users use, have them upgrade their browser to one that supports SNI.
- Use HTTP instead of HTTPS.

Requirements for Using SSL/TLS Certificates with CloudFront

The requirements for SSL/TLS certificates are described in this topic. They apply, except as noted, to both of the following:

- Certificates for using HTTPS between viewers and CloudFront
- Certificates for using HTTPS between CloudFront and your origin

Topics

- [Certificate Issuer \(p. 285\)](#)
- [AWS Region that You Request a Certificate In \(for AWS Certificate Manager\) \(p. 286\)](#)
- [Certificate Format \(p. 286\)](#)
- [Intermediate Certificates \(p. 286\)](#)
- [Key Type \(p. 286\)](#)
- [Private Key \(p. 286\)](#)
- [Permissions \(p. 287\)](#)
- [Size of the Public Key \(p. 287\)](#)
- [Supported Types of Certificates \(p. 287\)](#)
- [Certificate Expiration Date and Renewal \(p. 287\)](#)
- [Domain Names in the CloudFront Distribution and in the Certificate \(p. 288\)](#)
- [Minimum SSL Protocol Version \(p. 288\)](#)
- [Supported HTTP Versions \(p. 288\)](#)

Certificate Issuer

The certificate issuer you must use depends on whether you want to require HTTPS between viewers and CloudFront or between CloudFront and your origin:

- **HTTPS between viewers and CloudFront** – You can use a certificate that was issued by a trusted certificate authority (CA) such as Comodo, DigiCert, or Symantec; you can use a certificate provided by AWS Certificate Manager (ACM); or you can use a self-signed certificate.

- **HTTPS between CloudFront and a custom origin** – If the origin is *not* an ELB load balancer, such as Amazon EC2, the certificate must be issued by a trusted CA such as Comodo, DigiCert, or Symantec. If your origin is an ELB load balancer, you can also use a certificate provided by ACM.

Important

When CloudFront uses HTTPS to communicate with your origin, CloudFront verifies that the certificate was issued by a trusted CA. CloudFront supports the same certificate authorities as Mozilla; for the current list, see [Mozilla Included CA Certificate List](#). You cannot use a self-signed certificate for HTTPS communication between CloudFront and your origin.

For more information about getting and installing an SSL/TLS certificate, refer to the documentation for your HTTP server software and to the documentation for the certificate authority. For information about ACM, see the [AWS Certificate Manager User Guide](#).

AWS Region that You Request a Certificate In (for AWS Certificate Manager)

If you want to require HTTPS between viewers and CloudFront, you must change the AWS region to US East (N. Virginia) in the AWS Certificate Manager console before you request or import a certificate.

If you want to require HTTPS between CloudFront and your origin, and you're using an ELB load balancer as your origin, you can request or import a certificate in any region.

Certificate Format

The certificate must be in X.509 PEM format. This is the default format if you're using AWS Certificate Manager.

Intermediate Certificates

If you're using a third-party certificate authority (CA), in the .pem file, list all of the intermediate certificates in the certificate chain, beginning with one for the CA that signed the certificate for your domain. Typically, you'll find a file on the CA website that lists intermediate and root certificates in the proper chained order.

Important

Do not include the following: the root certificate, intermediate certificates that are not in the trust path, or your CA's public key certificate.

Here's an example:

```
-----BEGIN CERTIFICATE-----  
Intermediate certificate 2  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----  
Intermediate certificate 1  
-----END CERTIFICATE-----
```

Key Type

CloudFront supports only RSA public/private key pairs.

Private Key

If you're using a certificate from a third-party certificate authority (CA), note the following:

- The private key must match the public key that is in the certificate.
- The private key also must be an RSA private key in PEM format, where the PEM header is BEGIN RSA PRIVATE KEY and the footer is END RSA PRIVATE KEY.
- The private key cannot be encrypted with a password.

If AWS Certificate Manager (ACM) provided the certificate, ACM doesn't release the private key. The private key is stored in ACM for use by AWS services that are integrated with ACM.

Permissions

You must have permission to use and import the SSL/TLS certificate, including permission from the certificate authority (CA) that issued the certificate to import it to a content delivery network (CDN).

If you're using AWS Certificate Manager (ACM), we recommend that you use AWS Identity and Access Management permissions to restrict access to the certificates. For more information, see [Permissions and Policies](#) in the *AWS Certificate Manager User Guide*.

Size of the Public Key

The length of the public key for a certificate depends on where you're storing it.

- Importing a certificate into AWS Certificate Manager (ACM): public key length must be 1024 or 2048 bits.
- Uploading a certificate to the AWS Identity and Access Management (IAM) certificate store: maximum size of the public key is 2048 bits.

We recommend using 2048 bits.

For information about the public keys for certificates provided by ACM, see [ACM Certificate Characteristics](#) in the *AWS Certificate Manager User Guide*.

For information about how to determine the size of the public key, see [Determining the Size of the Public Key in an SSL/TLS Certificate \(p. 292\)](#).

Supported Types of Certificates

CloudFront supports all types of certificates, including the following:

- Domain-validated certificates
- Extended validation (EV) certificates
- High-assurance certificates
- Wildcard certificates (*.example.com)
- Subject alternative name (SAN) certificates (example.com and example.net)

Certificate Expiration Date and Renewal

If you're using certificates that you get from a third-party certificate authority (CA), you are responsible for monitoring certificate expiration dates and for renewing SSL/TLS certificates that you import into AWS Certificate Manager (ACM) or upload to the AWS Identity and Access Management certificate store.

If you're using ACM-provided certificates, ACM manages certificate renewals for you. For more information, see [Managed Renewal](#) in the *AWS Certificate Manager User Guide*.

Domain Names in the CloudFront Distribution and in the Certificate

When you're using a custom origin, the SSL/TLS certificate on your origin includes a domain name in the Common Name field, and possibly several more in the Subject Alternative Names field. (CloudFront supports wildcard characters in certificate domain names.)

One of the domain names in the certificate must match the domain name that you specify for Origin Domain Name. If no domain name matches, CloudFront returns HTTP status code 502 (Bad Gateway) to the viewer.

Minimum SSL Protocol Version

If you're using dedicated IP addresses, you can choose the minimum SSL protocol version for the connection between viewers and CloudFront by choosing a security policy.

For more information, see [Security Policy \(p. 79\)](#) in the topic [Values That You Specify When You Create or Update a Web Distribution \(p. 63\)](#).

Supported HTTP Versions

If you associate one certificate with more than one CloudFront distribution, all the distributions associated with the certificate must use the same option for [Supported HTTP Versions \(p. 80\)](#). You specify this option when you create or update a CloudFront distribution.

Limits on Using SSL/TLS Certificates with CloudFront (HTTPS Between Viewers and CloudFront Only)

Note the following limits on using SSL/TLS certificates with CloudFront. These limits apply only to the SSL/TLS certificates that you provision by using AWS Certificate Manager (ACM), or that you import into ACM or upload to the IAM certificate store for HTTPS communication between viewers and CloudFront.

Maximum Number of Certificates per CloudFront Distribution

You can associate a maximum of one SSL/TLS certificate with each CloudFront distribution.

Maximum Number of Certificates that You Can Import into ACM or Upload to the IAM Certificate Store

If you obtained your SSL/TLS certificates from a third-party CA, you need to store the certificates in one of the following locations:

- **AWS Certificate Manager** – For the current limit on the number of ACM certificates, see [Limits](#) in the *AWS Certificate Manager User Guide*. The listed limit is a total that includes certificates that you provision by using ACM and certificates that you import into ACM.
- **IAM certificate store** – For the current limit on the number of certificates that you can upload to the IAM certificate store for an AWS account, see [Limitations on IAM Entities and Objects](#) in the *IAM User Guide*. To request a higher limit, see [Request IAM limit increase](#).

Maximum Number of Certificates per AWS Account (Dedicated IP Addresses Only)

If you want to serve HTTPS requests by using dedicated IP addresses, note the following:

- By default, CloudFront gives you permission to use two certificates with your AWS account, one for everyday use and one for when you need to rotate certificates for multiple distributions.
- If you're already using this feature but you need to increase the number of custom SSL/TLS certificates that you can use with your AWS account, go to the [Support Center](#) and create a case.

Indicate how many certificates that you need permission to use, and describe the circumstances in your request. We'll update your account as soon as possible.

Using the Same Certificate for CloudFront Distributions that Were Created by Using Different AWS Accounts

If you're using a third-party CA and if you want to use the same certificate with multiple CloudFront distributions that were created by using different AWS accounts, you must import the certificate into ACM or upload it to the IAM certificate store once for each AWS account.

If you're using certificates provided by ACM, you can't configure CloudFront to use certificates that were created by a different AWS account.

Using the Same Certificate for CloudFront and for Other AWS Services (IAM Certificate Store Only)

If you bought a certificate from a trusted certificate authority such as Comodo, DigiCert, or Symantec, you can use the same certificate for CloudFront and for other AWS services. If you're importing the certificate into ACM, you need to import it only once to use it for multiple AWS services.

If you're using certificates provided by ACM, the certificates are stored in ACM.

Using the Same Certificate for Multiple CloudFront Distributions

You can use the same certificate for any or all of the CloudFront distributions that you're using to serve HTTPS requests. Note the following:

- You can use the same certificate both for serving requests using dedicated IP addresses and for serving requests using SNI.
- You can associate only one certificate with each distribution.
- Each distribution must include one or more alternate domain names that also appear in the Common Name field or the Subject Alternative Names field in the certificate.
- If you're serving HTTPS requests using dedicated IP addresses and you created all of your distributions by using the same AWS account, you can significantly reduce your cost by using the same certificate for all distributions. CloudFront charges for each certificate, not for each distribution.

For example, suppose you create three distributions by using the same AWS account, and you use the same certificate for all three distributions. You would be charged only one fee for using dedicated IP addresses.

However, if you're serving HTTPS requests using dedicated IP addresses and using the same certificate to create CloudFront distributions in different AWS accounts, each account will be charged the fee for using dedicated IP addresses. For example, if you create three distributions by using three different AWS accounts and you use the same certificate for all three distributions, each account will be charged the full fee for using dedicated IP addresses.

Configuring Alternate Domain Names and HTTPS

To use alternate domain names in the URLs for your objects and to use HTTPS between viewers and CloudFront, perform the applicable procedures.

Topics

- [Requesting Permission to Use Three or More SSL/TLS Certificates \(p. 290\)](#)
- [Getting an SSL/TLS Certificate \(p. 290\)](#)
- [Importing an SSL/TLS Certificate \(p. 290\)](#)
- [Updating Your CloudFront Distribution \(p. 291\)](#)

Requesting Permission to Use Three or More SSL/TLS Certificates

If you need permission to permanently associate three or more SSL/TLS Dedicated IP certificates with CloudFront, perform the following procedure. For more details about HTTPS requests, see [Choosing How CloudFront Serves HTTPS Requests \(p. 283\)](#)

Note

This procedure is for using 3 or more DedicatedIP certificates across your CloudFront distributions. The default value is 2. Keep in mind you cannot bind more than 1 SSL certificate to a distribution.

You can only associate a single SSL/TLS certificate to a CloudFront distribution at a time. This number is for the total number of Dedicated IP SSL certificates you can use across all of your CloudFront distributions.

To request permission to use three or more certificates with a CloudFront distribution

1. Go to the [Support Center](#) and create a case.
2. Indicate how many certificates you need permission to use, and describe the circumstances in your request. We'll update your account as soon as possible.
3. Continue with the next procedure.

Getting an SSL/TLS Certificate

Get an SSL/TLS certificate if you don't already have one. For more information, see the applicable documentation:

- To use a certificate provided by AWS Certificate Manager (ACM), see the [AWS Certificate Manager User Guide](#). Then skip to [Updating Your CloudFront Distribution \(p. 291\)](#).

Note

We recommend that you use ACM to provision, manage, and deploy SSL/TLS certificates on AWS managed resources.

- To get a certificate from a third-party certificate authority (CA), see the documentation provided by the certificate authority. When you have the certificate, continue with the next procedure.
- To create a self-signed certificate, see the documentation for the application that you're using to create and sign the certificate. Then continue with the next procedure.

Importing an SSL/TLS Certificate

If you got your certificate from a third-party CA, import the certificate into ACM or upload it to the IAM certificate store:

ACM (Recommended)

ACM lets you import third-party certificates from the ACM console, as well as programmatically. For information about importing a certificate to ACM, see [Importing Certificates into AWS Certificate Manager](#) in the [AWS Certificate Manager User Guide](#).

IAM certificate store

If ACM is not available in your region, use the following AWS CLI command to upload your third-party certificate to the IAM certificate store. (For a list of the regions where ACM is available, see [AWS Certificate Manager](#) in the "AWS Regions and Endpoints" chapter of the [Amazon Web Services General Reference](#).)

```
aws iam upload-server-certificate --server-certificate-name CertificateName --  
certificate-body file://public_key_certificate_file --private-key file://privatekey.pem  
--certificate-chain file://certificate_chain_file --path /cloudfront/path/
```

Note the following:

- **AWS Account** – You must upload the certificate to the IAM certificate store using the same AWS account that you used to create your CloudFront distribution.
- **--path Parameter** – When you upload the certificate to IAM, the value of the `-path` parameter (certificate path) must start with `/cloudfront/`, for example, `/cloudfront/production/` or `/cloudfront/test/`. The path must end with a `/`.
- **Existing certificates** – You must specify values for the `--server-certificate-name` and `--path` parameters that are different from the values that are associated with existing certificates.
- **Using the CloudFront Console** – The value that you specify for the `--server-certificate-name` parameter in the AWS CLI, for example, `myServerCertificate`, appears in the **SSL Certificate** list in the CloudFront console.
- **Using the CloudFront API** – Make note of the alphanumeric string that the AWS CLI returns, for example, `AS1A2M3P4L5E67SIIXR3J`. This is the value that you will specify in the `IAMCertificateId` element. You don't need the IAM ARN, which is also returned by the CLI.

For more information about the AWS CLI, see the [AWS Command Line Interface User Guide](#) and the [AWS Command Line Interface Reference](#).

Updating Your CloudFront Distribution

To update settings for your distribution, perform the following procedure:

To configure your CloudFront distribution for alternate domain names

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the ID for the distribution that you want to update.
3. On the **General** tab, choose **Edit**.
4. Update the following values:

Alternate Domain Names (CNAMEs)

Add the applicable alternate domain names. Separate domain names with commas, or type each domain name on a new line.

SSL Certificate (Web Distributions Only)

Choose **Custom SSL Certificate**, and choose a certificate from the list.

If you uploaded a certificate to the IAM certificate store but it doesn't appear in the list, review the procedure [Importing an SSL/TLS Certificate \(p. 290\)](#) to confirm that you correctly uploaded the certificate.

Important

After you associate your SSL/TLS certificate with your CloudFront distribution, do not delete the certificate from ACM or the IAM certificate store until you remove the certificate from all distributions and until the status of the distributions has changed to **Deployed**.

Clients Supported (Web Distributions Only)

Choose the applicable option:

- **All Clients:** CloudFront serves your HTTPS content using dedicated IP addresses. If you select this option, you incur additional charges when you associate your SSL/TLS certificate with a distribution that is enabled. For more information, see [Amazon CloudFront Pricing](#).
- **Only Clients that Support Server Name Indication (SNI):** Older browsers or other clients that don't support SNI must use another method to access your content.

For more information, see [Choosing How CloudFront Serves HTTPS Requests \(p. 283\)](#).

5. Choose **Yes, Edit**.
6. Configure CloudFront to require HTTPS between viewers and CloudFront:
 - a. On the **Behaviors** tab, choose the cache behavior that you want to update, and choose **Edit**.
 - b. Specify one of the following values for **Viewer Protocol Policy**:

Redirect HTTP to HTTPS

Viewers can use both protocols, but HTTP requests are automatically redirected to HTTPS requests. CloudFront returns HTTP status code 301 (Moved Permanently) along with the new HTTPS URL. The viewer then resubmits the request to CloudFront using the HTTPS URL.

Important

CloudFront doesn't redirect `DELETE`, `OPTIONS`, `PATCH`, `POST`, or `PUT` requests from HTTP to HTTPS. If you configure a cache behavior to redirect to HTTPS, CloudFront responds to HTTP `DELETE`, `OPTIONS`, `PATCH`, `POST`, or `PUT` requests for that cache behavior with HTTP status code 403 (Forbidden).

When a viewer makes an HTTP request that is redirected to an HTTPS request, CloudFront charges for both requests. For the HTTP request, the charge is only for the request and for the headers that CloudFront returns to the viewer. For the HTTPS request, the charge is for the request, and for the headers and the object returned by your origin.

HTTPS Only

Viewers can access your content only if they're using HTTPS. If a viewer sends an HTTP request instead of an HTTPS request, CloudFront returns HTTP status code 403 (Forbidden) and does not return the object.

- c. Choose **Yes, Edit**.
 - d. Repeat steps a through c for each additional cache behavior that you want to require HTTPS for between viewers and CloudFront.
7. Confirm the following before you use the updated configuration in a production environment:
 - The path pattern in each cache behavior applies only to the requests that you want viewers to use HTTPS for.
 - The cache behaviors are listed in the order that you want CloudFront to evaluate them in. For more information, see [Path Pattern \(p. 71\)](#).
 - The cache behaviors are routing requests to the correct origins.

Determining the Size of the Public Key in an SSL/TLS Certificate

When you're using CloudFront alternate domain names and HTTPS, the size of the public key in an SSL/TLS certificate cannot exceed 2048 bits. (This is not the number of characters in the public key.) You can determine the size of the public key by running the following OpenSSL command:

```
openssl x509 -in path and filename of SSL/TLS certificate -text -noout
```

where:

- `-in` specifies the path and filename of your SSL/TLS certificate.
- `-text` causes OpenSSL to display the length of the public key in bits.
- `-noout` prevents OpenSSL from displaying the public key.

Example output:

```
Public-Key: (2048 bit)
```

Increasing the Limit for SSL/TLS Certificates

There are limits on the number of SSL/TLS certificates that you can import into [AWS Certificate Manager](#) or upload to [AWS Identity and Access Management](#). There also is a limit on the number of SSL/TLS certificates that you can use with an AWS account when you configure CloudFront to serve HTTPS requests by using dedicated IP addresses. However, you can request higher limits.

Topics

- [Certificates That You Can Import into ACM \(p. 293\)](#)
- [Certificates That You Can Upload to IAM \(p. 293\)](#)
- [Certificates That You Can Use with Dedicated IP Addresses \(p. 294\)](#)

Certificates That You Can Import into ACM

For the limit on the number of certificates that you can import into ACM, see [ACM Limits](#) in the *Amazon Web Services General Reference*.

To request a higher limit, [create a case](#) with the AWS Support Center. Specify the following values:

- **Regarding** – Accept the default value of **Service Limit Increase**
- **Limit Type** – Choose **Certificate Manager**
- **Region** – Specify the AWS Region where you want to import certificates
- **Limit** – Choose **(ACM) Number of ACM Certificates**

Then fill out the rest of the form.

Certificates That You Can Upload to IAM

For the limit on the number of certificates that you can upload to IAM, see [IAM Limits](#) in the *Amazon Web Services General Reference*.

To request a higher limit, [create a case](#) with the AWS Support Center. Specify the following values:

- **Regarding** – Accept the default value of **Service Limit Increase**
- **Limit Type** – Choose **Certificate Manager**
- **Region** – Specify the AWS Region where you want to import certificates
- **Limit** – Choose **(IAM) Server Certificate Limit**

Then fill out the rest of the form.

Certificates That You Can Use with Dedicated IP Addresses

For the limit on the number of SSL certificates that you can use for each AWS account when serving HTTPS requests using dedicated IP addresses, see [CloudFront Limits](#) in the *Amazon Web Services General Reference*.

To request a higher limit, [create a case](#) with the AWS Support Center. Specify the following values:

- **Regarding** – Accept the default value of **Service Limit Increase**
- **Limit Type** – Accept the default value of **CloudFront Distributions**
- **Limit** – Choose **Dedicated IP SSL Certificate Limit per Account**

Then fill out the rest of the form.

Rotating SSL/TLS Certificates

If you're using certificates provided by AWS Certificate Manager (ACM), you don't need to rotate SSL/TLS certificates. ACM manages certificate renewals for you. For more information, see [Managed Renewal](#) in the *AWS Certificate Manager User Guide*.

Note

ACM does not manage certificate renewals for certificates that you acquire from third-party certificate authorities and import into ACM.

If you're using a third-party certificate authority and you imported certificates into ACM (recommended) or uploaded them to the IAM certificate store, you'll occasionally need to replace one certificate with another. For example, you need to replace a certificate when the expiration date on the certificate approaches.

Important

If you configured CloudFront to serve HTTPS requests by using dedicated IP addresses, you might incur an additional, pro-rated charge for using one or more additional certificates while you're rotating certificates. We recommend that you update your distributions promptly to minimize the additional charge.

To rotate certificates, perform the following procedure. Viewers can continue to access your content while you rotate certificates as well as after the process is complete.

To rotate SSL/TLS certificates

1. [Increasing the Limit for SSL/TLS Certificates \(p. 293\)](#) to determine whether you need permission to use more SSL certificates. If so, request permission and wait until permission is granted before you continue with step 2.
2. Import the new certificate into ACM or upload it to IAM. For more information, see [Importing an SSL/TLS Certificate](#) in the *Amazon CloudFront Developer Guide*
3. Update your distributions one at a time to use the new certificate. For more information, see [Listing, Viewing, and Updating CloudFront Distributions](#) in the *Amazon CloudFront Developer Guide*.
4. (Optional) After you have updated all of your CloudFront distributions, you can delete the old certificate from ACM or from IAM.

Important

Do not delete an SSL/TLS certificate until you remove it from all distributions and until the status of the distributions that you have updated has changed to **Deployed**.

Reverting from a Custom SSL/TLS Certificate to the Default CloudFront Certificate

If you configured CloudFront to use HTTPS between viewers and CloudFront, and you configured CloudFront to use a custom SSL/TLS certificate, you can change your configuration to use the default CloudFront SSL/TLS certificate. The process depends on whether you've used your distribution to distribute your content:

- If you have not used your distribution to distribute your content, you can just change the configuration. For more information, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#).
- If you have used your distribution to distribute your content, you need to create a new CloudFront distribution and change the URLs for your objects to reduce or eliminate the amount of time that your content is unavailable. To do that, perform the following procedure.

To revert to the default CloudFront certificate

1. Create a new CloudFront distribution with the desired configuration. For **SSL Certificate**, choose **Default CloudFront Certificate (*.cloudfront.net)**.

For more information, see [Task List for Creating a Web Distribution \(p. 59\)](#).

2. For objects that you're distributing using CloudFront, update the URLs in your application to use the domain name that CloudFront assigned to the new distribution. For example, change `https://www.example.com/images/logo.png` to `https://d111111abcdef8.cloudfront.net/images/logo.png`.
3. Either delete the distribution that is associated with a custom SSL/TLS certificate, or update the distribution to change the value of **SSL Certificate to Default CloudFront Certificate (*.cloudfront.net)**. For more information, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#).

Important

Until you complete this step, AWS continues to charge you for using a custom SSL/TLS certificate.

4. (Optional) Delete your custom SSL/TLS certificate.
 - a. Run the AWS CLI command `list-server-certificates` to get the certificate ID of the certificate that you want to delete. For more information, see [list-server-certificates](#) in the *AWS Command Line Interface Reference*.
 - b. Run the AWS CLI command `delete-signing-certificate` to delete the certificate. For more information, see [delete-signing-certificate](#) in the *AWS Command Line Interface Reference*.

Switching from a Custom SSL/TLS Certificate with Dedicated IP Addresses to SNI

If you configured CloudFront to use a custom SSL/TLS certificate with dedicated IP addresses, you can switch to using a custom SSL/TLS certificate with SNI instead and eliminate the charge that is associated with dedicated IP addresses. The following procedure shows you how.

Important

This update to your CloudFront configuration has no effect on viewers that support SNI; they can access your content before and after the change, as well as while the change is propagating to CloudFront edge locations. Viewers that don't support SNI cannot access your

content after the change. For more information, see [Choosing How CloudFront Serves HTTPS Requests \(p. 283\)](#).

To switch from a custom SSL/TLS certificate with dedicated IP addresses to SNI

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the ID of the distribution that you want to view or update.
3. Choose **Distribution Settings**.
4. On the **General** tab, choose **Edit**.
5. Change the setting of **Custom SSL Client Support** to **Only Clients that Support Server Name Indication (SNI)**.
6. Choose **Yes, Edit**.

Supported Protocols and Ciphers

You can choose HTTPS settings both for communication between viewers and CloudFront, and between CloudFront and your origin:

- **Between viewers and CloudFront** – If you require HTTPS between viewers and CloudFront, you also choose a security policy, which determines the protocols that viewers and CloudFront can use to communicate. In addition, a security policy determines which ciphers CloudFront can use to encrypt the content that it returns to viewers.
- **Between CloudFront and your origin** – If you require HTTPS between CloudFront and your origin, you also choose the protocols that CloudFront and your origin use to communicate. The protocols that you choose determine which ciphers your origin can use to encrypt content that it returns to CloudFront

See the applicable topic.

Topics

- [Supported SSL/TLS Protocols and Ciphers for Communication Between Viewers and CloudFront \(p. 296\)](#)
- [Supported SSL/TLS Protocols and Ciphers for Communication Between CloudFront and Your Origin \(p. 298\)](#)

Supported SSL/TLS Protocols and Ciphers for Communication Between Viewers and CloudFront

To choose whether to require HTTPS between viewers and CloudFront, specify the applicable value for [Viewer Protocol Policy \(p. 72\)](#).

If you choose to require HTTPS, you also choose the security policy that you want CloudFront to use for HTTPS connections. A security policy determines two settings:

- The SSL/TLS protocol that CloudFront uses to communicate with viewers
- The cipher that CloudFront uses to encrypt the content that it returns to viewers

We recommend that you specify **TLSv1.1_2016** unless your users are using browsers or devices that don't support TLSv1.1 or later. When you use a custom SSL certificate and SNI, you must use TLSv1 or later.

To choose a security policy, specify the applicable value for [Security Policy \(p. 79\)](#). The following table lists the protocols and ciphers that CloudFront can use for each security policy.

A viewer must support at least one of the supported ciphers to establish an HTTPS connection with CloudFront. If you're using an SSL/TLS certificate in AWS Certificate Manager, a viewer must support one of the *-RSA-* ciphers. CloudFront chooses a cipher in the listed order from among the ciphers that the viewer supports. See also [OpenSSL and RFC Cipher Names \(p. 297\)](#).

| | Security Policy | | | | |
|-----------------------------|-----------------|---|---|---|---|
| TLSv1.2 | ◆ | ◆ | ◆ | ◆ | ◆ |
| TLSv1.1 | ◆ | ◆ | ◆ | ◆ | |
| TLSv1 | ◆ | ◆ | ◆ | | |
| SSLv3 | ◆ | | | | |
| Ciphers Supported | | | | | |
| ECDHE-RSA-AES128-GCM-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ |
| ECDHE-RSA-AES128-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ |
| ECDHE-RSA-AES128-SHA | ◆ | ◆ | ◆ | ◆ | |
| ECDHE-RSA-AES256-GCM-SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ |
| ECDHE-RSA-AES256-SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ |
| ECDHE-RSA-AES256-SHA | ◆ | ◆ | ◆ | ◆ | |
| AES128-GCM-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ |
| AES256-GCM-SHA384 | ◆ | ◆ | ◆ | ◆ | ◆ |
| AES128-SHA256 | ◆ | ◆ | ◆ | ◆ | ◆ |
| AES256-SHA | ◆ | ◆ | ◆ | ◆ | |
| AES128-SHA | ◆ | ◆ | ◆ | ◆ | |
| DES-CBC3-SHA | ◆ | ◆ | | | |
| RC4-MD5 | ◆ | | | | |

OpenSSL and RFC Cipher Names

OpenSSL and IETF RFC 5246, [The Transport Layer Security \(TLS\) Protocol Version 1.2](#), use different names for the same ciphers. The following table maps the OpenSSL name to the RFC name for each cipher that CloudFront supports.

| OpenSSL Cipher Name | RFC Cipher Name |
|-----------------------------|---------------------------------------|
| ECDHE-RSA-AES128-GCM-SHA256 | TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 |

| OpenSSL Cipher Name | RFC Cipher Name |
|-----------------------------|---------------------------------------|
| ECDHE-RSA-AES128-SHA256 | TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 |
| ECDHE-RSA-AES128-SHA | TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA |
| ECDHE-RSA-AES256-GCM-SHA384 | TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 |
| ECDHE-RSA-AES256-SHA384 | TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 |
| ECDHE-RSA-AES256-SHA | TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA |
| AES128-GCM-SHA256 | TLS_RSA_WITH_AES_128_GCM_SHA256 |
| AES256-GCM-SHA384 | TLS_RSA_WITH_AES_256_GCM_SHA384 |
| AES128-SHA256 | TLS_RSA_WITH_AES_128_CBC_SHA256 |
| AES256-SHA | TLS_RSA_WITH_AES_256_CBC_SHA |
| AES128-SHA | TLS_RSA_WITH_AES_128_CBC_SHA |
| DES-CBC3-SHA | TLS_RSA_WITH_3DES_EDE_CBC_SHA |
| RC4-MD5 | TLS_RSA_WITH_RC4_128_MD5 |

Supported SSL/TLS Protocols and Ciphers for Communication Between CloudFront and Your Origin

You can choose whether to require HTTPS between CloudFront and your origin and, if so, which protocols to allow:

- To choose whether to require HTTPS between CloudFront and your origin, specify the applicable value for [Origin Protocol Policy \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 68\)](#).
- To choose the SSL/TLS protocols that you want CloudFront and your origin to use to communicate, specify the applicable values for [Origin SSL Protocols \(Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only\) \(p. 68\)](#).

CloudFront forwards HTTPS requests to the origin server by using the following ciphers. Your origin server must support at least one of these ciphers for CloudFront to establish an HTTPS connection to your origin.

| OpenSSL Cipher Name | RFC Cipher Name |
|-------------------------|---------------------------------------|
| ECDHE-RSA-AES128-SHA256 | TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 |
| ECDHE-RSA-AES256-SHA384 | TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 |
| AES256-SHA | TLS_RSA_WITH_AES_256_CBC_SHA |
| AES128-SHA | TLS_RSA_WITH_AES_128_CBC_SHA |
| DES-CBC3-SHA | TLS_RSA_WITH_3DES_EDE_CBC_SHA |
| RC4-MD5 | TLS_RSA_WITH_RC4_128_MD5 |

Charges for HTTPS Connections

You always incur a surcharge for HTTPS requests. For more information, see [Amazon CloudFront Pricing](#).

Authentication and Access Control for CloudFront

To perform any operation on CloudFront resources, such as creating a web distribution or an invalidation, [AWS Identity and Access Management \(IAM\)](#) requires you to authenticate that you're an approved AWS user. If you're using the CloudFront console, you authenticate your identity by providing your AWS user name and a password. If you're accessing CloudFront programmatically, your application authenticates your identity for you by using access keys or by signing requests.

After you authenticate your identity, IAM controls your access to AWS by verifying that you have permissions to perform operations and access resources. If you are an account administrator, you can use IAM to control the access of other users to the resources that are associated with your account.

This chapter explains how to use [AWS Identity and Access Management \(IAM\)](#) and CloudFront to help secure your resources.

Topics

- [Authentication \(p. 300\)](#)
- [Access Control \(p. 301\)](#)

Authentication

You can access AWS as any of the following types of identities:

- **AWS account root user** – When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.
- **IAM user** – An [IAM user](#) is an identity within your AWS account that has specific custom permissions (for example, permissions to create a web distribution in CloudFront). You can use an IAM user name and password to sign in to secure AWS webpages like the [AWS Management Console](#), [AWS Discussion Forums](#), or the [AWS Support Center](#).

In addition to a user name and password, you can also generate [access keys](#) for each user. You can use these keys when you access AWS services programmatically, either through [one of the several SDKs](#) or by using the [AWS Command Line Interface \(CLI\)](#). The SDK and CLI tools use the access keys to cryptographically sign your request. If you don't use AWS tools, you must sign the request yourself. CloudFront supports *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see [Signature Version 4 Signing Process](#) in the [AWS General Reference](#).

- **IAM role** – An [IAM role](#) is an IAM identity that you can create in your account that has specific permissions. It is similar to an *IAM user*, but it is not associated with a specific person. An IAM role

enables you to obtain temporary access keys that can be used to access AWS services and resources. IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – Instead of creating an IAM user, you can use existing user identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an [identity provider](#). For more information about federated users, see [Federated Users and Roles](#) in the *IAM User Guide*.
- **AWS service access** – You can use an IAM role in your account to grant an AWS service permissions to access your account's resources. For example, you can create a role that allows Amazon Redshift to access an Amazon S3 bucket on your behalf and then load data from that bucket into an Amazon Redshift cluster. For more information, see [Creating a Role to Delegate Permissions to an AWS Service](#) in the *IAM User Guide*.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances](#) in the *IAM User Guide*.

Access Control

To create, update, delete, or list CloudFront resources, you need permissions to perform the operation, and you need permissions to access the corresponding resources. In addition, to perform the operation programmatically, you need valid access keys.

The following sections describe how to manage permissions for CloudFront:

- [Overview of Managing Access Permissions to Your CloudFront Resources \(p. 301\)](#)
- [Using Identity-Based Policies \(IAM Policies\) for CloudFront \(p. 304\)](#)
- [CloudFront API Permissions: Actions, Resources, and Conditions Reference \(p. 309\)](#)

Overview of Managing Access Permissions to Your CloudFront Resources

Every AWS resource is owned by an AWS account, and permissions to create or access a resource are governed by permissions policies.

Note

An *account administrator* (or administrator user) is a user that has administrator privileges. For more information about administrators, see [IAM Best Practices](#) in the *IAM User Guide*.

When you grant permissions, you decide who gets the permissions, the resources they get permissions for, and the actions that they get permission to perform.

Topics

- [ARNs for CloudFront Resources \(p. 302\)](#)
- [Understanding Resource Ownership \(p. 302\)](#)
- [Managing Access to Resources \(p. 302\)](#)
- [Specifying Policy Elements: Resources, Actions, Effects, and Principals \(p. 303\)](#)
- [Specifying Conditions in a Policy \(p. 304\)](#)

ARNs for CloudFront Resources

All CloudFront resources—web and RTMP distributions, invalidations, and origin access identities—use the same format for Amazon Resource Names (ARNs):

`arn:aws:cloudfront::optional-account-id:*`

CloudFront provides API actions to work with each of these types of resources. For more information, see the [Amazon CloudFront API Reference](#). For a list of actions and the ARN that you specify to grant or deny permission to use each action, see [CloudFront API Permissions: Actions, Resources, and Conditions Reference \(p. 309\)](#).

Understanding Resource Ownership

An AWS account owns the resources that are created in the account, regardless of who created the resources. Specifically, the resource owner is the AWS account of the principal entity (that is, the root account, an IAM user, or an IAM role) that authenticates the resource creation request.

The following examples illustrate how this works:

- If you use the root account credentials of your AWS account to create a web distribution, your AWS account is the owner of the distribution.
- If you create an IAM user in your AWS account and grant permissions to create a web distribution to that user, the user can create a web distribution. The AWS account that created the user owns the distribution.
- If you create an IAM role in your AWS account with permissions to create a web distribution, anyone who can assume the role can create a web distribution. Your AWS account, to which the role belongs, owns the distribution.

Managing Access to Resources

A *permissions policy* specifies who has access to what. This section explains the options for creating permissions policies for CloudFront. For general information about IAM policy syntax and descriptions, see the [AWS IAM Policy Reference](#) in the *IAM User Guide*.

Policies attached to an IAM identity are referred to as identity-based policies (IAM policies), and policies attached to a resource are referred to as resource-based policies. CloudFront supports only identity-based policies (IAM policies).

Topics

- [Identity-Based Policies \(IAM Policies\) \(p. 302\)](#)
- [Resource-Based Policies \(p. 303\)](#)

Identity-Based Policies (IAM Policies)

You can attach policies to IAM identities. For example, you can do the following:

- **Attach a permissions policy to a user or a group in your account** – An account administrator can use a permissions policy that is associated with a particular user to grant permissions for that user to create a web distribution.
- **Attach a permissions policy to a role (grant cross-account permissions)** – You can grant permissions to perform CloudFront actions to a user that was created in another AWS account. To do so, you attach a permissions policy to an IAM role, and then you allow the user in the other account to assume the role. The following example explains how this works for two AWS accounts, account A and account B:
 1. Account A administrator creates an IAM role and attaches to the role a permissions policy that grants permissions to create or access resources that are owned by account A.
 2. Account A administrator attaches a trust policy to the role. The trust policy identifies account B as the principal that can assume the role.
 3. Account B administrator can then delegate permissions to assume the role to users or groups in account B. This allows users in account B to create or access resources in account A.

For more information about how to delegate permissions to users in another AWS account, see [Access Management](#) in the *IAM User Guide*.

The following example policy allows a user to perform the `CreateDistribution` action to programmatically create a web distribution for your AWS account:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "cloudfront:CreateDistribution"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

For information about the permissions required to perform operations by using the CloudFront console, see [Permissions Required to Use the CloudFront Console \(p. 305\)](#). For more information about attaching policies to identities for CloudFront, see [Using Identity-Based Policies \(IAM Policies\) for CloudFront \(p. 304\)](#). For more information about users, groups, roles, and permissions, see [Identities \(Users, Groups, and Roles\)](#) in the *IAM User Guide*.

Resource-Based Policies

Other services, such as Amazon S3, support attaching permissions policies to resources. For example, you can attach a policy to an S3 bucket to manage access permissions to that bucket. CloudFront doesn't support attaching policies to resources.

Specifying Policy Elements: Resources, Actions, Effects, and Principals

CloudFront includes API actions (see [Amazon CloudFront API Reference](#)) that you can use on each CloudFront resource (see [ARNs for CloudFront Resources \(p. 302\)](#)). You can grant a user or a federated user permission to perform any or all of these actions.

The following are the basic policy elements:

- **Resource** – You use an Amazon Resource Name (ARN) to identify the resource that the policy applies to. For more information, see [ARNs for CloudFront Resources \(p. 302\)](#).

- **Action** – You use action keywords to identify resource operations that you want to allow or deny. For example, depending on the specified `Effect`, the `cloudfront:CreateDistribution` permission allows or denies the user permissions to perform the CloudFront `CreateDistribution` action.
- **Effect** – You specify the effect, either allow or deny, when a user tries to perform the action on the specified resource. If you don't explicitly grant access to an action, access is implicitly denied. You can also explicitly deny access to a resource, which you might do to make sure that a user cannot access it, even if a different policy grants access.
- **Principal** – In identity-based policies (IAM policies), the user that the policy is attached to is the implicit principal. For resource-based policies, you specify the user, account, service, or other entity that you want to receive permissions (applies to resource-based policies only). CloudFront doesn't support resource-based policies.

For more information about IAM policy syntax and descriptions, see the [AWS IAM Policy Reference](#) in the *IAM User Guide*.

For a list showing all of the CloudFront API operations and the resources that they apply to, see [CloudFront API Permissions: Actions, Resources, and Conditions Reference \(p. 309\)](#).

Specifying Conditions in a Policy

When you grant permissions, you can use the IAM policy language to specify when a policy should take effect. For example, you might want a policy to be applied only after a specific date. For more information about specifying conditions in a policy language, see [Condition](#) in the *IAM User Guide*.

To express conditions, you use predefined condition keys. There are no condition keys specific to CloudFront. However, there are AWS-wide condition keys that you can use as appropriate. For a complete list of AWS-wide keys, see [Available Keys for Conditions](#) in the *IAM User Guide*.

Using Identity-Based Policies (IAM Policies) for CloudFront

This topic provides examples of identity-based policies that demonstrate how an account administrator can attach permissions policies to IAM identities (that is, users, groups, and roles) and thereby grant permissions to perform operations on CloudFront resources.

Important

We recommend that you first review the introductory topics that explain the basic concepts and options to manage access to your CloudFront resources. For more information, see [Overview of Managing Access Permissions to Your CloudFront Resources \(p. 301\)](#).

Topics

- [Permissions Required to Use the CloudFront Console \(p. 305\)](#)
- [AWS Managed \(Predefined\) Policies for CloudFront \(p. 306\)](#)
- [Customer Managed Policy Examples \(p. 307\)](#)

The following shows a permissions policy. The `Sid`, or statement ID, is optional:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowAllCloudFrontPermissions",  
            "Effect": "Allow",  
            "Action": "cloudfront:CreateDistribution",  
            "Resource": "*"  
        }  
    ]  
}
```

```
        "Action": ["cloudfront:*"],
        "Resource": "*"
    }
}
```

The policy grants permissions to perform all CloudFront operations, which is sufficient to access CloudFront programmatically. If you're using the console to access CloudFront, see [Permissions Required to Use the CloudFront Console \(p. 305\)](#).

For a list of actions and the ARN that you specify to grant or deny permission to use each action, see [CloudFront API Permissions: Actions, Resources, and Conditions Reference \(p. 309\)](#).

Permissions Required to Use the CloudFront Console

To grant full access to the CloudFront console, you grant the permissions in the following permissions policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "acm>ListCertificates",
                "cloudfront:*",
                "cloudwatch>DescribeAlarms",
                "cloudwatch>PutMetricAlarm",
                "cloudwatch>GetMetricStatistics",
                "elasticloadbalancing>DescribeLoadBalancers",
                "iam>ListServerCertificates",
                "sns>ListSubscriptionsByTopic",
                "sns>ListTopics",
                "waf>GetWebACL",
                "waf>ListWebACLs"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3>ListAllMyBuckets",
                "s3>PutBucketPolicy"
            ],
            "Resource": "arn:aws:s3:::/*"
        }
    ]
}
```

Here's why the permissions are required:

acm>ListCertificates

When you're creating and updating web distributions by using the CloudFront console and you want to configure CloudFront to require HTTPS between the viewer and CloudFront or between CloudFront and the origin, lets you view a list of ACM certificates.

This permission isn't required if you aren't using the CloudFront console.

cloudfront:*

Lets you perform all CloudFront actions.

cloudwatch:DescribeAlarms and cloudwatch:PutMetricAlarm

Let you create and view CloudWatch alarms in the CloudFront console. See also `sns>ListSubscriptionsByTopic` and `sns>ListTopics`.

These permissions aren't required if you aren't using the CloudFront console.

cloudwatch:GetMetricStatistics

Lets CloudFront render CloudWatch metrics in the CloudFront console.

This permission isn't required if you aren't using the CloudFront console.

elasticloadbalancing:DescribeLoadBalancers

When creating and updating web distributions, lets you view a list of Elastic Load Balancing load balancers in the list of available origins.

This permission isn't required if you aren't using the CloudFront console.

iam>ListServerCertificates

When you're creating and updating web distributions by using the CloudFront console and you want to configure CloudFront to require HTTPS between the viewer and CloudFront or between CloudFront and the origin, lets you view a list of certificates in the IAM certificate store.

This permission isn't required if you aren't using the CloudFront console.

s3>ListAllMyBuckets

When you're creating and updating web and RTMP distributions, lets you perform the following operations:

- View a list of S3 buckets in the list of available origins
- View a list of S3 buckets that you can save access logs in

This permission isn't required if you aren't using the CloudFront console.

s3:PutBucketPolicy

When you're creating or updating distributions that restrict access to S3 buckets, lets a user update the bucket policy to grant access to the CloudFront origin access identity. For more information, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

This permission isn't required if you aren't using the CloudFront console.

sns>ListSubscriptionsByTopic and sns>ListTopics

When you create CloudWatch alarms in the CloudFront console, lets you choose an SNS topic for notifications.

These permissions aren't required if you aren't using the CloudFront console.

waf:GetWebACL and waf>ListWebACLS

Lets you view a list of AWS WAF web ACLs in the CloudFront console.

These permissions aren't required if you aren't using the CloudFront console.

AWS Managed (Predefined) Policies for CloudFront

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. These AWS managed policies grant necessary permissions for common use cases so that you can avoid having to investigate what permissions are needed. For more information, see [AWS Managed Policies](#) in the *IAM User Guide*. For CloudFront, IAM provides two managed policies:

- **CloudFrontFullAccess** – Grants full access to CloudFront resources.

Important

If you want CloudFront to create and save access logs, you need to grant additional permissions. For more information, see [Permissions Required to Configure Logging and to Access Your Log Files \(p. 316\)](#).

- **CloudFrontReadOnlyAccess** – Grants read-only access to CloudFront resources.

Note

You can review these permissions policies by signing in to the IAM console and searching for specific policies there. You can also create your own custom IAM policies to allow permissions for CloudFront API operations. You can attach these custom policies to the IAM users or groups that require those permissions.

Customer Managed Policy Examples

You can create your own custom IAM policies to allow permissions for CloudFront API actions. You can attach these custom policies to the IAM users or groups that require the specified permissions. These policies work when you are using the CloudFront API, the AWS SDKs, or the AWS CLI. The following examples show permissions for a few common use cases. For the policy that grants a user full access to CloudFront, see [Permissions Required to Use the CloudFront Console \(p. 305\)](#).

Examples

- [Example 1: Allow Read Access to All Web Distributions \(p. 307\)](#)
- [Example 2: Allow Creation, Updating, and Deletion of Web Distributions \(p. 308\)](#)
- [Example 3: Allow Creation and Listing of Invalidations \(p. 308\)](#)

Example 1: Allow Read Access to All Web Distributions

The following permissions policy grants the user permissions to view all web distributions in the CloudFront console:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "acm>ListCertificates",  
                "cloudfront:GetDistribution",  
                "cloudfront:GetDistributionConfig",  
                "cloudfront>ListDistributions",  
                "cloudfront>ListCloudFrontOriginAccessIdentities",  
                "elasticloadbalancing:DescribeLoadBalancers",  
                "iam>ListServerCertificates",  
                "sns>ListSubscriptionsByTopic",  
                "sns>ListTopics",  
                "waf:GetWebACL",  
                "waf>ListWebACLs"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3>ListAllMyBuckets"  
            ],  
            "Resource": "arn:aws:s3:::/*"  
        }  
    ]  
}
```

```
        ]
    }
```

Example 2: Allow Creation, Updating, and Deletion of Web Distributions

The following permissions policy allows users to create, update, and delete web distributions by using the CloudFront console:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "acm>ListCertificates",
                "cloudfront>CreateDistribution",
                "cloudfront>DeleteDistribution",
                "cloudfront>GetDistribution",
                "cloudfront>GetDistributionConfig",
                "cloudfront>ListDistributions",
                "cloudfront>UpdateDistribution",
                "cloudfront>ListCloudFrontOriginAccessIdentities",
                "elasticloadbalancing>DescribeLoadBalancers",
                "iam>ListServerCertificates",
                "sns>ListSubscriptionsByTopic",
                "sns>ListTopics",
                "waf>GetWebACL",
                "waf>ListWebACLs"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3>ListAllMyBuckets",
                "s3>PutBucketPolicy"
            ],
            "Resource": "arn:aws:s3:::/*"
        }
    ]
}
```

The `cloudfront>ListCloudFrontOriginAccessIdentities` permission allows users to automatically grant to an existing origin access identity the permission to access objects in an Amazon S3 bucket. If you also want users to be able to create origin access identities, you also need to allow the `cloudfront>CreateCloudFrontOriginAccessIdentity` permission.

Example 3: Allow Creation and Listing of Invalidations

The following permissions policy allows users to create and list invalidations. It includes read access to CloudFront distributions because you create and view invalidations by first displaying settings for a distribution:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudfront>GetDistribution",
                "cloudfront>GetDistributionConfig",
                "cloudfront>ListDistributions"
            ],
            "Resource": "arn:aws:cloudfront::/*"
        }
    ]
}
```

```

    "Action": [
        "acm>ListCertificates",
        "cloudfront>GetDistribution",
        "cloudfront>GetDistributionConfig",
        "cloudfront>ListDistributions",
        "cloudfront>ListCloudFrontOriginAccessIdentities",
        "cloudfront>CreateInvalidation",
        "cloudfront>GetInvalidation",
        "cloudfront>ListInvalidations",
        "elasticloadbalancing>DescribeLoadBalancers",
        "iam>ListServerCertificates",
        "sns>ListSubscriptionsByTopic",
        "sns>ListTopics",
        "waf>GetWebACL",
        "waf>ListWebACLS"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "s3>ListAllMyBuckets"
    ],
    "Resource": "arn:aws:s3:::/*"
}
]
}

```

CloudFront API Permissions: Actions, Resources, and Conditions Reference

When you are setting up [Access Control \(p. 301\)](#) and writing a permissions policy that you can attach to an IAM identity (identity-based policies), you can use the following lists as a reference. The lists include each CloudFront API operation, the corresponding actions for which you can grant permissions to perform the action, and the AWS resource for which you can grant the permissions. You specify the actions in the policy's Action field, and you specify the resource value in the policy's Resource field.

You can use AWS-wide condition keys in your CloudFront policies to express conditions. For a complete list of AWS-wide keys, see [Available Keys](#) in the *IAM User Guide*.

Topics

- [Required Permissions for Actions on Web Distributions \(p. 309\)](#)
- [Required Permissions for Actions on RTMP Distributions \(p. 311\)](#)
- [Required Permissions for Actions on Invalidations \(p. 312\)](#)
- [Required Permissions for Actions on Origin Access Identities \(p. 312\)](#)
- [Required Permissions for CloudFront Actions Related to Lambda@Edge \(p. 313\)](#)
- [Required Permissions for Actions on Tags \(p. 314\)](#)

Required Permissions for Actions on Web Distributions

[CreateDistribution](#)

Required Permissions (API Action):

- `cloudfront:CreateDistribution`
- `acm>ListCertificates` (CloudFront console only)
- Only if you configure CloudFront to save access logs:
 - `s3:GetBucketAcl`
 - `s3:PutBucketAcl`
 - The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources:

- CloudFront: *
- ACM: *
- Amazon S3: If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

[CreateDistributionWithTags](#)

Required Permissions (API Action):

- `cloudfront:CreateDistribution`, `cloudfront:TagResource`
- `acm>ListCertificates` (CloudFront console only)
- Only if you configure CloudFront to save access logs:
 - `s3:GetBucketAcl`
 - `s3:PutBucketAcl`
 - The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources:

- CloudFront: *
- ACM: *
- Amazon S3: If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

[GetDistribution](#)

Required Permissions (API Action): `cloudfront:GetDistribution`, `acm>ListCertificates` (CloudFront console only)

Resources: *

[GetDistributionConfig](#)

Required Permissions (API Action): `cloudfront:GetDistributionConfig`,
`acm>ListCertificates` (CloudFront console only)

Resources: *

[ListDistributions](#)

Required Permissions (API Action): `cloudfront>ListDistributions`

Resources: *

[UpdateDistribution](#)

Required Permissions (API Action):

- `cloudfront:UpdateDistribution`
- `acm>ListCertificates` (CloudFront console only)
- Only if you configure CloudFront to save access logs:
 - `s3:GetBucketAcl`
 - `s3:PutBucketAcl`

- The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources:

- CloudFront: *
- ACM: *
- Amazon S3: If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

[DeleteDistribution](#)

Required Permissions (API Action): `cloudfront:DeleteDistribution`

Resources: *

Required Permissions for Actions on RTMP Distributions

[CreateStreamingDistribution](#)

Required Permissions (API Action): `cloudfront>CreateStreamingDistribution`

Only if you configure CloudFront to save access logs:

- `s3:GetBucketAcl`
- `s3:PutBucketAcl`
- The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources: *

If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

[CreateStreamingDistributionWithTags](#)

Required Permissions (API Action): `cloudfront>CreateStreamingDistribution`,
`cloudfront:TagResource`

Only if you configure CloudFront to save access logs:

- `s3:GetBucketAcl`
- `s3:PutBucketAcl`
- The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources: *

If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

[GetStreamingDistribution](#)

Required Permissions (API Action): `cloudfront:GetStreamingDistribution`

Resources: *

[GetStreamingDistributionConfig](#)

Required Permissions (API Action): `cloudfront:GetStreamingDistributionConfig`

Resources: *

[ListStreamingDistributions](#)

Required Permissions (API Action): `cloudfront>ListStreamingDistributions`

Resources: *

[UpdateStreamingDistribution](#)

Required Permissions (API Action): `cloudfront>UpdateStreamingDistribution`

Only if you configure CloudFront to save access logs:

- `s3:GetBucketAcl`
- `s3:PutBucketAcl`
- The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources: *

If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

[DeleteStreamingDistribution](#)

Required Permissions (API Action): `cloudfront>DeleteDistribution`

Resources: *

Required Permissions for Actions on Invalidations

[CreateInvalidation](#)

Required Permissions (API Action): `cloudfront>CreateInvalidation`

Resources: *

[GetInvalidation](#)

Required Permissions (API Action): `cloudfront>GetInvalidation`

Resources: *

[ListInvalidations](#)

Required Permissions (API Action): `cloudfront>ListInvalidations`

Resources: *

Required Permissions for Actions on Origin Access Identities

[CreateCloudFrontOriginAccessIdentity](#)

Required Permissions (API Action): `cloudfront>CreateCloudFrontOriginAccessIdentity`

Resources: *

[GetCloudFrontOriginAccessIdentity](#)

Required Permissions (API Action): `cloudfront>GetCloudFrontOriginAccessIdentity`

Resources: *

[GetCloudFrontOriginAccessIdentityConfig](#)

Required Permissions (API Action):

`cloudfront:GetCloudFrontOriginAccessIdentityConfig`

Resources: *

[ListCloudFrontOriginAccessIdentities](#)

Required Permissions (API Action): `cloudfront>ListDistributions`

Resources: *

[UpdateCloudFrontOriginAccessIdentity](#)

Required Permissions (API Action): `cloudfront:UpdateCloudFrontOriginAccessIdentity`

Resources: *

[DeleteCloudFrontOriginAccessIdentity](#)

Required Permissions (API Action): `cloudfront>DeleteCloudFrontOriginAccessIdentity`

Resources: *

Required Permissions for CloudFront Actions Related to Lambda@Edge

To use Lambda@Edge, you need the following CloudFront permissions so you can create or update a distribution that includes triggers for Lambda functions. For information about the Lambda permissions that you need, see [Setting IAM Permissions](#) in the "AWS Lambda@Edge" chapter in the *AWS Lambda Developer Guide*.

[CreateDistribution](#)

Required Permissions (API Action):

- `cloudfront>CreateDistribution`
- `acm>ListCertificates` (CloudFront console only)
- Only if you configure CloudFront to save access logs:
 - `s3:GetBucketAcl`
 - `s3:PutBucketAcl`
 - The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources:

- CloudFront: *
- ACM: *
- Amazon S3: If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

[CreateDistributionWithTags](#)

Required Permissions (API Action):

- `cloudfront>CreateDistribution`, `cloudfront:TagResource`
- `acm>ListCertificates` (CloudFront console only)
- Only if you configure CloudFront to save access logs:
 - `s3:GetBucketAcl`

- `s3:PutBucketAcl`
- The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources:

- CloudFront: *
- ACM: *
- Amazon S3: If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

[UpdateDistribution](#)

Required Permissions (API Action):

- `cloudfront:UpdateDistribution`
- `acm>ListCertificates` (CloudFront console only)
- Only if you configure CloudFront to save access logs:
 - `s3:GetBucketAcl`
 - `s3:PutBucketAcl`
 - The S3 ACL for the bucket must grant you `FULL_CONTROL`

Resources:

- CloudFront: *
- ACM: *
- Amazon S3: If you configure CloudFront to save access logs, you can optionally restrict access to a specified bucket.

Required Permissions for Actions on Tags

[TagResource](#)

Required Permissions (API Action): `cloudfront:TagResource`

Resources: *

[UntagResource](#)

Required Permissions (API Action): `cloudfront:UntagResource`

Resources: *

[ListTagsForResource](#)

Required Permissions (API Action): `cloudfront>ListTagsForResource`

Resources: *

Access Logs

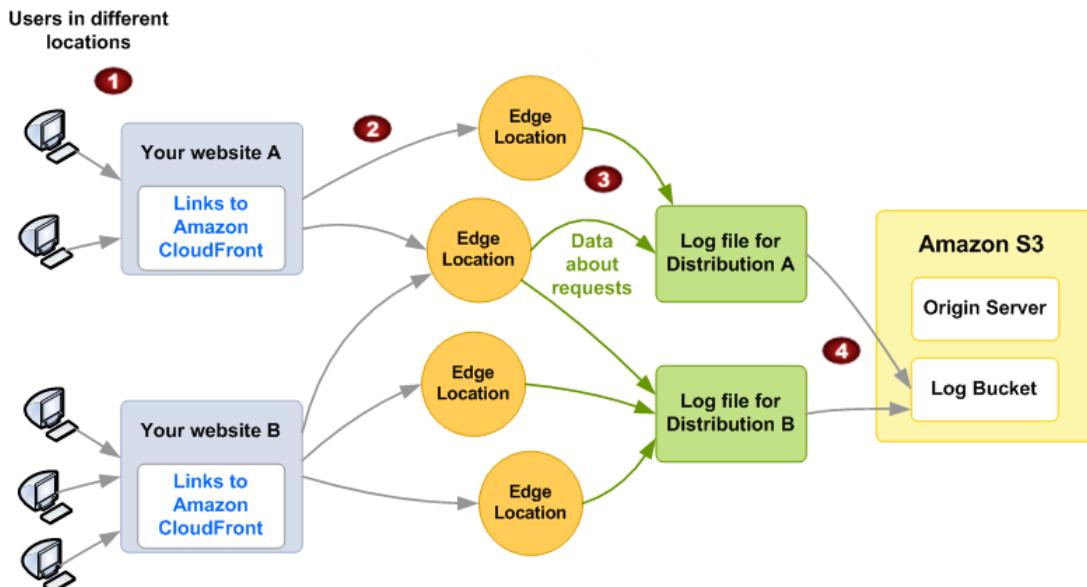
You can configure CloudFront to create log files that contain detailed information about every user request that CloudFront receives. These access logs are available for both web and RTMP distributions. If you enable logging, you can also specify the Amazon S3 bucket that you want CloudFront to save files in.

Topics

- [How Logging Works \(p. 315\)](#)
- [Choosing an Amazon S3 Bucket for Your Access Logs \(p. 316\)](#)
- [Permissions Required to Configure Logging and to Access Your Log Files \(p. 316\)](#)
- [File Name Format \(p. 317\)](#)
- [Timing of Log File Delivery \(p. 317\)](#)
- [Analyzing Access Logs \(p. 318\)](#)
- [Editing Your Logging Settings \(p. 318\)](#)
- [Deleting Log Files from an Amazon S3 Bucket \(p. 318\)](#)
- [Log File Format \(p. 318\)](#)
- [Charges for Access Logs \(p. 329\)](#)

How Logging Works

The following diagram shows how CloudFront logs information about requests for your objects.



The following explains how CloudFront logs information about requests for your objects, as illustrated in the previous graphic.

1. In this diagram, you have two websites, A and B, and two corresponding CloudFront distributions. Users request your objects using URLs that are associated with your distributions.

2. CloudFront routes each request to the appropriate edge location.
3. CloudFront writes data about each request to a log file specific to that distribution. In this example, information about requests related to Distribution A goes into a log file just for Distribution A, and information about requests related to Distribution B goes into a log file just for Distribution B.
4. CloudFront periodically saves the log file for a distribution in the Amazon S3 bucket that you specified when you enabled logging. CloudFront then starts saving information about subsequent requests in a new log file for the distribution.

Each entry in a log file gives details about a single request. For more information about log file format, see [Log File Format \(p. 318\)](#).

Choosing an Amazon S3 Bucket for Your Access Logs

When you enable logging for a distribution, you specify the Amazon S3 bucket that you want CloudFront to store log files in. If you're using Amazon S3 as your origin, we recommend that you do not use the same bucket for your log files; using a separate bucket simplifies maintenance.

You can store the log files for multiple distributions in the same bucket. When you enable logging, you can specify an optional prefix for the file names, so you can keep track of which log files are associated with which distributions.

If no users access your content during a given hour, you don't receive any log files for that hour.

Permissions Required to Configure Logging and to Access Your Log Files

Your AWS account must have the following permissions for the bucket that you specify for log files:

- The S3 access control list (ACL) for the bucket must grant you `FULL_CONTROL`. If you're the bucket owner, your account has this permission by default. If you're not, the bucket owner must update the ACL for the bucket.
- `s3:GetBucketAcl`
- `s3:PutBucketAcl`

Note the following:

ACL for the bucket

When you create or update a distribution and enable logging, CloudFront uses these permissions to update the ACL for the bucket to give the `awsdatafeeds` account `FULL_CONTROL` permission. The `awsdatafeeds` account writes log files to the bucket. If your account doesn't have the required permissions, creating or updating the distribution will fail.

Important

If you update the ACL for the bucket to remove permissions for the `awsdatafeeds` account, CloudFront won't be able to save logs to the S3 bucket any longer.

In some circumstances, if you programmatically submit a request to create a bucket but a bucket with the specified name already exists, S3 resets permissions on the bucket to the default value.

If you configured CloudFront to save access logs in an S3 bucket and you stop getting logs in that bucket, check permissions on the bucket to ensure that CloudFront has the necessary permissions.

ACL for each log file

In addition to the ACL on the bucket, there's an ACL on each log file. The bucket owner has `FULL_CONTROL` permission on each log file, the distribution owner (if different from the bucket owner) has no permission, and the `awsdatafeeds` account has read and write permissions.

Disabling logging

If you disable logging, CloudFront doesn't delete the ACLs for either the bucket or the log files. If you want, you can do that yourself.

File Name Format

The name of each log file that CloudFront saves in your Amazon S3 bucket uses the following file name format:

`bucket-name.s3.amazonaws.com/optional-prefix/distribution-ID.YYYY-MM-DD-HH.unique-ID.gz`

The date and time are in Coordinated Universal time (UTC).

For example, if your bucket name is `mylogs`, your prefix is `myprefix/`, and your distribution ID is `EMLARXS9EXAMPLE`, your file names look similar to this:

`mylogs.s3.amazonaws.com/myprefix/EMLARXS9EXAMPLE.2014-11-14-20.RT4KCN4SGK9.gz`

When you enable logging for a distribution, you can specify an optional prefix for the file names, so you can keep track of which log files are associated with which distributions. If you include a value for the log file prefix and your prefix doesn't include a `/`, CloudFront adds one automatically. If your value does include a `/`, CloudFront doesn't add another one.

The `.gz` at the end of the file name indicates that CloudFront has compressed the log file using gzip.

Timing of Log File Delivery

CloudFront delivers access logs for a distribution up to several times an hour. In general, a log file contains information about the requests that CloudFront received during a given time period. CloudFront usually delivers the log file for that time period to your Amazon S3 bucket within an hour of the events that appear in the log. Note, however, that some or all log file entries for a time period can sometimes be delayed by up to 24 hours. When log entries are delayed, CloudFront saves them in a log file for which the file name includes the date and time of the period in which the requests occurred, not the date and time when the file was delivered.

When creating a log file, CloudFront consolidates information for your distribution from all of the edge locations that received requests for your objects during the time period that the log file covers.

CloudFront can save more than one file for a time period depending on how many requests CloudFront receives for the objects associated with a distribution.

CloudFront begins to reliably deliver access logs about four hours after you enable logging. You might get a few access logs before that time.

Note

If no users request your objects during the time period, you don't receive any log files for that period.

Analyzing Access Logs

Because you can receive multiple access logs an hour, we recommend that you combine all the log files you receive for a given period into one file. You can then analyze the data for that period more quickly and accurately.

Important

We recommend that you use the logs to understand the nature of the requests for your content, not as a complete accounting of all requests. CloudFront delivers access logs on a best-effort basis. The log entry for a particular request might be delivered long after the request was actually processed and, in rare cases, a log entry might not be delivered at all. When a log entry is omitted from access logs, the number of entries in the access logs won't match the usage that appears in the AWS usage and billing reports.

For more information about CloudFront access logs, including recommendations for tools that you can use to analyze access logs, see [Using CloudFront Logging \(p. 365\)](#).

Editing Your Logging Settings

You can enable or disable logging, change the Amazon S3 bucket where your logs are stored, and change the prefix for log files by using the CloudFront console or the CloudFront API. Your changes to logging settings take effect within 12 hours.

For more information, see the following topics:

- Updating a web or an RTMP distribution using the CloudFront console: [Listing, Viewing, and Updating CloudFront Distributions \(p. 49\)](#).
- Updating a web distribution using the CloudFront API: [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.
- Updating an RTMP distribution using the CloudFront API: [PUT Streaming Distribution Config](#) in the *Amazon CloudFront API Reference*.

To use the CloudFront API to change access log settings for web distributions, you must use the 2009-04-02 or later version of the API. To use the CloudFront API to change access log settings for RTMP distributions, you must use the 2010-05-01 or later version of the API.

Deleting Log Files from an Amazon S3 Bucket

CloudFront does not automatically delete log files from your Amazon S3 bucket. For information about deleting log files from an Amazon S3 bucket, see the following topics:

- Using the Amazon S3 console: [Deleting an Object](#) in the *Amazon Simple Storage Service Console User Guide*.
- Using the REST API: [DELETE Object](#) in the *Amazon Simple Storage Service API Reference*.
- Using the SOAP API: [DeleteObject](#) in the *Amazon Simple Storage Service API Reference*.

Log File Format

Topics

- [Web Distribution Log File Format \(p. 319\)](#)
- [RTMP Distribution Log File Format \(p. 327\)](#)

Each entry in a log file gives details about a single user request. The log files for web and for RTMP distributions are not identical, but they share the following characteristics:

- Use the W3C extended log file format. (For more information, go to <http://www.w3.org/TR/WD-logfile.html>.)
- Contain tab-separated values.
- Contain records that are not necessarily in chronological order.
- Contain two header lines: one with the file-format version, and another that lists the W3C fields included in each record.
- Substitute URL-encoded equivalents for spaces and non-standard characters in field values.

These non-standard characters consist of all ASCII codes below 32 and above 127, plus the characters in the following table. The URL encoding standard is RFC 1738. For more information, go to <http://www.ietf.org/rfc/rfc1738.txt>.

| URL-Encoded Value | Character |
|-------------------|-----------|
| %3C | < |
| %3E | > |
| %2522 | " |
| %23 | # |
| %25 | % |
| %7B | { |
| %7D | } |
| %7C | |
| %255C | \ |
| %5E | ^ |
| %7E | ~ |
| %5B | [|
| %5D | |
| %60 | ` |
| %27 | ' |
| %2520 | space |

Web Distribution Log File Format

The log file for a web distribution includes the following fields in the listed order.

| Field Number | Field Name | Description |
|--------------|-----------------|---|
| 1 | date | The date on which the event occurred in the format yyyy-mm-dd, for example, 2015-06-30. The date and time are in Coordinated Universal Time (UTC). |
| 2 | time | The time when the CloudFront server finished responding to the request (in UTC), for example, 01:42:39. |
| 3 | x-edge-location | The edge location that served the request. Each edge location is identified by a three-letter code and an arbitrarily assigned number, for example, DFW3. The three-letter code typically corresponds with the International Air Transport Association airport code for an airport near the edge location. (These abbreviations might change in the future.) For a list of edge locations, see the Amazon CloudFront detail page, http://aws.amazon.com/cloudfront . |
| 4 | sc-bytes | The total number of bytes that CloudFront served to the viewer in response to the request, including headers, for example, 1045619. |
| 5 | c-ip | The IP address of the viewer that made the request, for example, 192.0.2.183 or 2001:0db8:85a3:0000:8a2e:0370:7334. If the viewer used an HTTP proxy or a load balancer to send the request, the value of c-ip is the IP address of the proxy or load balancer. See also x-Forwarded-For in field 20. |
| 6 | cs-method | The HTTP access method: DELETE, GET, HEAD, OPTIONS, PATCH, POST, or PUT. |
| 7 | cs(Host) | The domain name of the CloudFront distribution, for example, d111111abcdef8.cloudfront.net. |
| 8 | cs-uri-stem | The portion of the URI that identifies the path and object, for example, /images/daily-ad.jpg. |
| 9 | sc-status | <p>One of the following values:</p> <ul style="list-style-type: none"> An HTTP status code (for example, 200). For a list of HTTP status codes, see RFC 2616, Hypertext Transfer Protocol—HTTP 1.1, section 10, Status Code Definitions. For more information, see How CloudFront Processes and Caches HTTP 4xx and 5xx Status Codes from Your Origin (p. 203). 000, which indicates that the viewer closed the connection (for example, closed the browser tab) before CloudFront could respond to a request. <p>If the viewer closes the connection after CloudFront starts to send the object, the log contains the applicable HTTP status code.</p> |
| 10 | cs(Referer) | The name of the domain that originated the request. Common referrers include search engines, other websites that link directly to your objects, and your own website. |
| 11 | cs(User-Agent) | The value of the User-Agent header in the request. The User-Agent header identifies the source of the request, such as the type of device and browser that submitted the request and, if the |

| Field Number | Field Name | Description |
|--------------|--------------|---|
| | | request came from a search engine, which search engine. For more information, see User-Agent Header (p. 199) . |
| 12 | cs-uri-query | <p>The query string portion of the URI, if any. When a URI doesn't contain a query string, the value of cs-uri-query is a hyphen (-).</p> <p>For more information, see Configuring CloudFront to Cache Based on Query String Parameters (p. 147).</p> |
| 13 | cs(Cookie) | <p>The cookie header in the request, including name-value pairs and the associated attributes. If you enable cookie logging, CloudFront logs the cookies in all requests regardless of which cookies you choose to forward to the origin: none, all, or a whitelist of cookie names. When a request doesn't include a cookie header, the value of cs(Cookie) is a hyphen (-).</p> <p>For more information about cookies, see Configuring CloudFront to Cache Objects Based on Cookies (p. 150).</p> |

| Field Number | Field Name | Description |
|--------------|---------------------------------|--|
| 14 | <code>x-edge-result-type</code> | <p>How CloudFront classifies the response after the last byte left the edge location. In some cases, the result type can change between the time that CloudFront is ready to send the response and the time that CloudFront has finished sending the response. For example, in HTTP streaming, suppose CloudFront finds a segment in the edge cache. The value of <code>x-edge-response-result-type</code>, the result type immediately before CloudFront begins to respond to the request, is <code>Hit</code>. However, if the user closes the viewer before CloudFront has delivered the entire segment, the final result type—the value of <code>x-edge-result-type</code>—changes to <code>Error</code>.</p> <p>Possible values include:</p> <ul style="list-style-type: none"> • <code>Hit</code> – CloudFront served the object to the viewer from the edge cache. <p>For information about a situation in which CloudFront classifies the result type as <code>Hit</code> even though the response from the origin contains a <code>Cache-Control: no-cache</code> header, see Simultaneous Requests for the Same Object (Traffic Spikes) (p. 199).</p> <ul style="list-style-type: none"> • <code>RefreshHit</code> – CloudFront found the object in the edge cache but it had expired, so CloudFront contacted the origin to determine whether the cache has the latest version of the object and, if not, to get the latest version. • <code>Miss</code> – The request could not be satisfied by an object in the edge cache, so CloudFront forwarded the request to the origin server and returned the result to the viewer. • <code>LimitExceeded</code> – The request was denied because a CloudFront limit was exceeded. • <code>CapacityExceeded</code> – CloudFront returned an HTTP 503 status code (Service Unavailable) because the CloudFront edge server was temporarily unable to respond to requests. • <code>Error</code> – Typically, this means the request resulted in a client error (<code>sc-status</code> is <code>4xx</code>) or a server error (<code>sc-status</code> is <code>5xx</code>). • <code>Redirect</code> – CloudFront redirects from HTTP to HTTPS. <p>If <code>sc-status</code> is <code>403</code> and you configured CloudFront to restrict the geographic distribution of your content, the request might have come from a restricted location. For more information about geo restriction, see Restricting the Geographic Distribution of Your Content (p. 94).</p> <p>If the value of <code>x-edge-result-type</code> is <code>Error</code> and the value of <code>x-edge-response-result-type</code> is not <code>Error</code>, the client disconnected before finishing the download.</p> |
| 15 | <code>x-edge-request-id</code> | An encrypted string that uniquely identifies a request. |

| Field Number | Field Name | Description |
|--------------|-----------------|---|
| 16 | x-host-header | <p>The value that the viewer included in the <code>Host</code> header for this request. This is the domain name in the request:</p> <ul style="list-style-type: none"> If you're using the CloudFront domain name in your object URLs, such as <code>http://d11111abcdef8.cloudfront.net/logo.png</code>, the <code>x-host-header</code> field contains that domain name. If you're using alternate domain names in your object URLs, such as <code>http://example.com/logo.png</code>, the <code>x-host-header</code> field contains the alternate domain name, such as <code>example.com</code>. To use alternate domain names, you must add them to your distribution. For more information, see Using Alternate Domain Names (CNAMEs) (p. 50). <p>If you're using alternate domain names, see <code>cs(Host)</code> in field 7 for the domain name that is associated with your distribution.</p> |
| 17 | cs-protocol | The protocol that the viewer specified in the request, either <code>http</code> or <code>https</code> . |
| 18 | cs-bytes | The number of bytes of data that the viewer included in the request (client to server bytes), including headers. |
| 19 | time-taken | The number of seconds (to the thousandth of a second, for example, 0.002) between the time that a CloudFront edge server receives a viewer's request and the time that CloudFront writes the last byte of the response to the edge server's output queue as measured on the server. From the perspective of the viewer, the total time to get the full object will be longer than this value due to network latency and TCP buffering. |
| 20 | x-forwarded-for | <p>If the viewer used an HTTP proxy or a load balancer to send the request, the value of <code>c-ip</code> in field 5 is the IP address of the proxy or load balancer. In that case, <code>x-forwarded-for</code> is the IP address of the viewer that originated the request.</p> <p>If the viewer did not use an HTTP proxy or a load balancer, the value of <code>x-forwarded-for</code> is a hyphen (-).</p> <p>Note The <code>X-Forwarded-For</code> header contains IPv4 addresses (such as 192.0.2.44) and IPv6 addresses (such as 2001:0db8:85a3:0000:0000:8a2e:0370:7334), as applicable.</p> |

| Field Number | Field Name | Description |
|--------------|---------------------------|--|
| 21 | <code>ssl-protocol</code> | <p>When <code>cs-protocol</code> in field 17 is <code>https</code>, the SSL protocol that the client and CloudFront negotiated for transmitting the request and response. When <code>cs-protocol</code> is <code>http</code>, the value for <code>ssl-protocol</code> is a hyphen (-).</p> <p>Possible values include the following:</p> <ul style="list-style-type: none"> • <code>SSLv3</code> • <code>TLSv1</code> • <code>TLSv1.1</code> • <code>TLSv1.2</code> |
| 22 | <code>ssl-cipher</code> | <p>When <code>cs-protocol</code> in field 17 is <code>https</code>, the SSL cipher that the client and CloudFront negotiated for encrypting the request and response. When <code>cs-protocol</code> is <code>http</code>, the value for <code>ssl-cipher</code> is a hyphen (-).</p> <p>Possible values include the following:</p> <ul style="list-style-type: none"> • <code>ECDHE-RSA-AES128-GCM-SHA256</code> • <code>ECDHE-RSA-AES128-SHA256</code> • <code>ECDHE-RSA-AES128-SHA</code> • <code>ECDHE-RSA-AES256-GCM-SHA384</code> • <code>ECDHE-RSA-AES256-SHA384</code> • <code>ECDHE-RSA-AES256-SHA</code> • <code>AES128-GCM-SHA256</code> • <code>AES256-GCM-SHA384</code> • <code>AES128-SHA256</code> • <code>AES256-SHA</code> • <code>AES128-SHA</code> • <code>DES-CBC3-SHA</code> • <code>RC4-MD5</code> |

| Field Number | Field Name | Description |
|--------------|--|--|
| 23 | <code>x-edge-response-result-type</code> | <p>How CloudFront classified the response just before returning the response to the viewer. See also <code>x-edge-result-type</code> in field 14.</p> <p>Possible values include:</p> <ul style="list-style-type: none"> • <code>Hit</code> – CloudFront served the object to the viewer from the edge cache. • <code>RefreshHit</code> – CloudFront found the object in the edge cache but it had expired, so CloudFront contacted the origin to verify that the cache has the latest version of the object. • <code>Miss</code> – The request could not be satisfied by an object in the edge cache, so CloudFront forwarded the request to the origin server and returned the result to the viewer. • <code>LimitExceeded</code> – The request was denied because a CloudFront limit was exceeded. • <code>CapacityExceeded</code> – CloudFront returned a 503 error because the edge location didn't have enough capacity at the time of the request to serve the object. • <code>Error</code> – Typically, this means the request resulted in a client error (<code>sc-status</code> is <code>4xx</code>) or a server error (<code>sc-status</code> is <code>5xx</code>). • <code>Redirect</code> – CloudFront redirects from HTTP to HTTPS. <p>If <code>sc-status</code> is <code>403</code> and you configured CloudFront to restrict the geographic distribution of your content, the request might have come from a restricted location. For more information about geo restriction, see Restricting the Geographic Distribution of Your Content (p. 94).</p> <p>If the value of <code>x-edge-result-type</code> is <code>Error</code> and the value of <code>x-edge-response-result-type</code> is not <code>Error</code>, the client disconnected before finishing the download.</p> |
| 24 | <code>cs-protocol-version</code> | The HTTP version that the viewer specified in the request. Possible values include <code>HTTP/0.9</code> , <code>HTTP/1.0</code> , <code>HTTP/1.1</code> , and <code>HTTP/2.0</code> . |

| Field Number | Field Name | Description |
|--------------|------------|---|
| 25 | fle-status | <p>When field-level encryption is configured for a distribution, a code that indicates whether the request body was successfully processed. If field-level encryption is not configured for the distribution, the value of <code>fle-status</code> is a hyphen (-).</p> <p>When CloudFront successfully processes the request body, encrypts values in the specified fields, and forwards the request to the origin, the value of the <code>fle-status</code> column is <code>Processed</code>. The value of <code>x-edge-result-type</code>, column 14, can still indicate a client-side or server-side error.</p> <p>If the request exceeds a field-level encryption limit, <code>fle-status</code> contains one of the following error codes, and CloudFront returns HTTP status code 400 to the viewer. For a list of the current limits on field-level encryption, see Limits on Field-Level Encryption (p. 362).</p> <ul style="list-style-type: none"> • <code>FieldLengthLimitClientError</code> – A field that is configured to be encrypted exceeded the length limit • <code>FieldNumberLimitClientError</code> – A request that CloudFront is configured to encrypt contains more than the number of fields allowed • <code>RequestLengthLimitClientError</code> – The length of the request body exceeded the limit when field-level encryption is configured <p>Other possible values for <code>fle-status</code> include the following:</p> <ul style="list-style-type: none"> • <code>ForwardedByContentType</code> – CloudFront forwarded the request to the origin without parsing or encryption because no content type was configured. • <code>ForwardedByQueryArgs</code> – CloudFront forwarded the request to the origin without parsing or encryption because the request contains a query argument that wasn't in the configuration for field-level encryption. • <code>ForwardedDueToNoProfile</code> – CloudFront forwarded the request to the origin without parsing or encryption because no profile was specified in the configuration for field-level encryption. • <code>MalformedContentTypeClientError</code> – CloudFront rejected the request and returned an HTTP 400 status code to the viewer because the value of the Content-Type header was in an invalid format. • <code>MalformedInputClientError</code> – CloudFront rejected the request and returned an HTTP 400 status code to the viewer because the request body was in an invalid format. • <code>MalformedQueryArgsClientError</code> – CloudFront rejected the request and returned an HTTP 400 status code to the viewer because a query argument was empty or in an invalid format. • <code>RejectedByContentType</code> – CloudFront rejected the request and returned an HTTP 400 status code to the viewer because no content type was specified in the configuration for field-level encryption. |

| Field Number | Field Name | Description |
|--------------|----------------------|---|
| | | <ul style="list-style-type: none"> • RejectedByQueryArgs – CloudFront rejected the request and returned an HTTP 400 status code to the viewer because no query argument was specified in the configuration for field-level encryption. • ServerError – The server returned an error. |
| 26 | fle-encrypted-fields | The number of fields that CloudFront encrypted and forwarded to the origin. CloudFront streams the processed request to the origin as it encrypts data, so fle-encrypted-fields can have a value even if the value of fle-status is an error. If field-level encryption is not configured for the distribution, the value of fle-encrypted-fields is a hyphen (-). |

Note

Question marks (?) in URLs and query strings are not included in the log.

The following is an example log file for a web distribution:

```
#Version: 1.0
#Fields: date time x-edge-location sc-bytes c-ip cs-method cs(Host) cs-uri-stem sc-status
cs(Rerer) cs(User-Agent) cs-uri-query cs(Cookie) x-edge-result-type x-edge-request-id x-
host-header cs-protocol cs-bytes time-taken x-forwarded-for ssl-protocol ssl-cipher x-edge-
response-result-type cs-protocol-version fle-status fle-encrypted-fields
2014-05-23 01:13:11 FRA2 182 192.0.2.10 GET d11111abcdef8.cloudfront.net /
view/my/file.html 200 www.displaymyfiles.com Mozilla/4.0%20(compatible;
%20MSIE%205.0b1;%20Mac_PowerPC) - zip=98101 RefreshHit
MRVMF7KydIvxMWfJ1g1gwHQwZsbG2IhRJ07sn9AkKUFSHS9EXAMPLE== d11111abcdef8.cloudfront.net
http - 0.001 - - - RefreshHit HTTP/1.1 Processed 1
2014-05-23 01:13:12 LAX1 2390282 192.0.2.202 GET d11111abcdef8.cloudfront.net /soundtrack/
happy.mp3 304 www.unknownsingers.com Mozilla/4.0%20(compatible;%20MSIE%207.0;%20Windows
%20NT%205.1) a=b&c=d zip=50158 Hit xGN7KWPvEMB9Dp7ctcVFQC4E-nrcOcEKS3QyAez--06dV7TEXAMPLE==
d11111abcdef8.cloudfront.net http - 0.002 - - - Hit HTTP/1.1 - -
```

RTMP Distribution Log File Format

Each record in an RTMP access log represents a playback event, for example, connect, play, pause, stop, disconnect, and so on. As a result, CloudFront generates multiple log records each time a viewer watches a video. To relate log records that stem from the same stream ID, use the **x-sid** field.

Note

Some fields have values for all events, and some have values only for Play, Stop, Pause, Unpause, and Seek events. Usually, when the log file contains a hyphen (-) for a field, the field isn't relevant for the corresponding event.

The following table describes the fields that are present in each record in the RTMP distribution log file, regardless of the type of event. The fields appear in the log in the order listed.

| Field Number | Field Name | Description |
|--------------|------------|--|
| 1 | date | The date on which the event occurred in the format yyyy-mm-dd, for example, 2014-05-23. The date and time are in Coordinated Universal Time (UTC). |
| 2 | time | The time when the server received the request (in UTC), for example, 01:42:39. |

| Field Number | Field Name | Description |
|--------------|-----------------|---|
| 3 | x-edge-location | The edge location where the playback event occurred. Each edge location is identified by a three-letter code and an arbitrarily assigned number, for example, DFW3. The three-letter code typically corresponds with the International Air Transport Association airport code for an airport near the edge location. (These abbreviations might change in the future.) For a list of edge locations, see the Amazon CloudFront detail page, http://aws.amazon.com/cloudfront . |
| 4 | c-ip | Client IP, for example, 192.0.2.183. |
| 5 | x-event | The event type. This is a Connect, Disconnect, Play, Stop, Pause, Unpause, or Seek event. |
| 6 | sc-bytes | The running total number of bytes sent from the server to the client, up to the time of the event. |
| 7 | x-cf-status | A code indicating the status of the event. Currently, "OK" is the only value for this field. New functionality in the future could require new status codes. |
| 8 | x-cf-client-id | An opaque string identifier that can be used to differentiate clients. This value is unique for each connection. |
| 9 | cs-uri-stem | The stem portion of the URI, including the application and the application instance. This is sometimes referred to as the FMS <i>connect string</i> . For example, rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st. |
| 10 | cs-uri-query | The query string portion of the URI that is included on the connect string. |
| 11 | c-referrer | The URI of the referrer. |
| 12 | x-page-url | The URL of the page from which the SWF is linked. |
| 13 | c-user-agent | The value of the User-Agent header in the request. The User-Agent header identifies the type of device that submitted the request. For more information, see User-Agent Header (p. 199) . |

The following fields usually have values only for Play, Stop, Pause, Unpause, and Seek events. For other events, they contain a single hyphen (-). These fields appear in the log after the fields in the preceding table and in the order listed.

| Field | Description |
|---------------|--|
| x-sname | The stream name. |
| x-sname-query | The stream query string, if any. |
| x-file-ext | The stream type, for example, FLV. |
| x-sid | The stream ID. This is a unique integer identifier for the connection. |

Note

Question marks (?) in URLs and query strings are not included in the log.

The following is an example of a log file for an RTMP distribution:

```
#Version: 1.0
#Fields: date time x-edge-location c-ip x-event sc-bytes x-cf-status x-cf-client-id cs-uri-
stem cs-uri-query c-referrer x-page-url c-user-agent x-sname x-sname-query x-file-ext x-sid
2010-03-12 23:51:20 SEA4 192.0.2.147 connect 2014 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st key=value
http://player.longtailvideo.com/player.swf http://www.longtailvideo.com/support/jw-
player-setup-wizard?example=204 LNX%2010,0,32,18 - - -
2010-03-12 23:51:21 SEA4 192.0.2.222 play 3914 OK bfd8a98bee0840d9b871b7f6ade9908f
rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st key=value http://player.longtailvideo.com/
player.swf http://www.longtailvideo.com/support/jw-player-setup-wizard?example=204 LNX
%2010,0,32,18 myvideo p=2&q=4 flv 1
2010-03-12 23:53:44 SEA4 192.0.2.4 stop 323914 OK bfd8a98bee0840d9b871b7f6ade9908f
rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st key=value http://player.longtailvideo.com/
player.swf http://www.longtailvideo.com/support/jw-player-setup-wizard?example=204 LNX
%2010,0,32,18 dir/other/myvideo p=2&q=4 flv 1
2010-03-12 23:53:44 SEA4 192.0.2.103 play 8783724 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st key=value
http://player.longtailvideo.com/player.swf http://www.longtailvideo.com/support/jw-
player-setup-wizard?example=204 LNX%2010,0,32,18 dir/favs/myothervideo p=42&q=14 mp4 2
2010-03-12 23:56:21 SEA4 192.0.2.199 stop 429822014 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st key=value
http://player.longtailvideo.com/player.swf http://www.longtailvideo.com/support/jw-
player-setup-wizard?example=204 LNX%2010,0,32,18 dir/favs/myothervideo p=42&q=14 mp4 2
2010-03-12 23:59:44 SEA4 192.0.2.14 disconnect 429824092 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st key=value
http://player.longtailvideo.com/player.swf http://www.longtailvideo.com/support/jw-
player-setup-wizard?example=204 LNX%2010,0,32,18 - - -
```

Charges for Access Logs

Access logging is an optional feature of CloudFront. There is no extra charge for enabling access logging. However, you accrue the usual Amazon S3 charges for storing and accessing the files on Amazon S3 (you can delete them at any time). For more information about charges for CloudFront, see [CloudFront Reports \(p. 15\)](#).

Monitoring CloudFront Activity Using CloudWatch

Amazon CloudFront integrates with Amazon CloudWatch metrics so that you can monitor your website or application. CloudFront currently provides six free metrics, and these metrics don't count against [CloudWatch limits](#). When viewing metrics, you can specify a time interval of as little as one minute for time periods in the previous two weeks.

Viewing Metrics for a Distribution

To view metrics for a distribution in the CloudWatch console

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the navigation pane, click **Monitoring and Alarms**.
3. In the **CloudFront Metrics and Alarms From CloudWatch** pane, specify the following values:

From and To

Select the date and time range for which you want to display CloudWatch metrics.

Granularity

Specify the interval of the data points, for example, one per minute or one per hour. Note that the time period that you choose affects the available granularity. For example, if you choose to view data for two weeks, the finest granularity is one hour, and if you choose to view data for 24 hours, the finest granularity is one minute.

Web Distribution

Select the distribution that you want to display metrics for.

4. Click **Update Graph** to refresh the graph based on the settings that you specified.

Receiving Notifications

To receive an Amazon Simple Notification Service (Amazon SNS) notification based on a CloudFront metric

1. On the **CloudFront Metrics and Alarms From CloudWatch** page, expand the list of existing alarms to confirm that the alarm that you want to create doesn't already exist.
2. Click **Create Alarm**.
3. In the **Create Alarm** dialog box, specify the following values:

Metric

Choose the metric for which you want to create the alarm.

Distribution

Choose the CloudFront distribution for which you want to create the alarm.

Name of alarm

Enter a name for the alarm.

Send notification to

Choose the existing Amazon SNS topic that you want to send notification to if the status of this metric triggers an alarm.

Whenever metric *operator value*

Specify when CloudWatch should trigger an alarm and send a notification to the specified email list. For example, to receive notification when the 5xx error rate exceeds 1%, you'd specify the following:

Whenever Average of 5xxErrorRate > 1

Note the following about specifying values for *value*:

- Enter only whole numbers without punctuation. For example, to specify one thousand, enter **1000**.
- For 4xx, 5xx, and total error rates, the value that you specify is a percentage.
- For requests, bytes downloaded, and bytes uploaded, the value you specify is in units, for example, **1000000000** bytes.

For at least *x* consecutive periods of *time period*

Specify how many consecutive time periods of the specified duration the metric must meet the criteria before CloudWatch sends notification. When you choose a value, you need to find an appropriate balance between a value that produces frequent notifications for fleeting problems and delayed notifications for real problems.

4. If you created a new Amazon SNS topic, when you click **Create**, Amazon SNS sends you an email with information about the new topic. Follow the instructions in the email.

Downloading Data in CSV Format

You can download the CloudWatch Metrics report in CSV format. This section explains how to download the report and describes the values in the report.

To download the CloudWatch Metrics report in CSV format

1. While viewing the CloudWatch metrics, click **CSV**.
2. In the **Opening file name** dialog box, choose whether to open or save the file.

Information About the Report

The first few rows of the report include the following information:

Version

The CloudFront reporting version.

Report

The name of the report.

DistributionID

The ID of the distribution that you ran the report for.

StartDateUTC

The beginning of the date range for which you ran the report, in Coordinated Universal Time (UTC).

EndDateUTC

The end of the date range for which you ran the report, in Coordinated Universal Time (UTC).

GeneratedTimeUTC

The date and time on which you ran the report, in Coordinated Universal Time (UTC).

Granularity

The time period for each row in the report, for example, ONE_MINUTE.

Data in the CloudWatch Metrics Report

The report includes the following values:

DistributionID

The ID of the distribution that you ran the report for.

FriendlyName

An alternate domain name (CNAME) for the distribution, if any. If a distribution has no alternate domain names, the list includes an origin domain name for the distribution.

TimeBucket

The hour or the day that data applies to, in Coordinated Universal Time (UTC).

Requests

The total number of requests for all HTTP status codes (for example, 200 or 404) and all methods (for example, GET, HEAD, or POST) during the time period.

BytesDownloaded

The number of bytes that viewers downloaded for the specified distribution during the time period.

BytesUploaded

The number of bytes that viewers uploaded to your origin for the specified distribution during the time period.

TotalErrorRatePct

Requests for which the HTTP status code was a 4xx or 5xx error for the specified distribution during the time period.

4xxErrorRatePct

Requests for which the HTTP status code was a 4xx error for the specified distribution during the time period.

5xxErrorRatePct

Requests for which the HTTP status code was a 5xx error for the specified distribution during the time period.

Amazon CloudFront Metrics

The AWS/CloudFront namespace includes the following metrics.

Note

Only one statistic, Average or Sum, is applicable for each metric. However, all statistics are available through the console, API, and AWS Command Line Interface. In the following table, each metric specifies the statistic that is applicable to that metric.

| Metric | Description |
|-----------------|--|
| Requests | The number of requests for all HTTP methods and for both HTTP and HTTPS requests. Valid Statistics: Sum Units: None |
| BytesDownloaded | The number of bytes downloaded by viewers for GET, HEAD, and OPTIONS requests. Valid Statistics: Sum Units: None |
| BytesUploaded | The number of bytes uploaded to your origin with CloudFront using POST and PUT requests. Valid Statistics: Sum Units: None |
| TotalErrorRate | The percentage of all requests for which the HTTP status code is 4xx or 5xx. Valid Statistics: Average Units: Percent |
| 4xxErrorRate | The percentage of all requests for which the HTTP status code is 4xx. Valid Statistics: Average Units: Percent |
| 5xxErrorRate | The percentage of all requests for which the HTTP status code is 5xx. Valid Statistics: Average Units: Percent |

Dimensions for CloudFront Metrics

CloudFront metrics use the CloudFront namespace and provide metrics for two dimensions:

| Dimension | Description |
|----------------|--|
| DistributionId | The CloudFront ID of the distribution for which you want to display metrics. |

| Dimension | Description |
|-----------|---|
| Region | The region for which you want to display metrics. This value must be <code>Global</code> . The <code>Region</code> dimension is different from the region in which CloudFront metrics are stored, which is US East (N. Virginia). |

Using AWS CloudTrail to Capture Requests Sent to the CloudFront API

CloudFront is integrated with CloudTrail, an AWS service that captures information about every request that is sent to the CloudFront API by your AWS account, including your IAM users. CloudTrail periodically saves log files of these requests to an Amazon S3 bucket that you specify. CloudTrail captures information about all requests, whether they were made using the CloudFront console, the CloudFront API, the AWS SDKs, the CloudFront CLI, or another service, for example, AWS CloudFormation.

You can use information in the CloudTrail log files to determine which requests were made to CloudFront, the source IP address from which each request was made, who made the request, when it was made, and so on. To learn more about CloudTrail, including how to configure and enable it, see the [AWS CloudTrail User Guide](#).

Note

CloudFront is a global service. To view CloudFront requests in CloudTrail logs, you must update an existing trail to include global services. For more information, see [Updating a Trail](#) and [About Global Service Events](#) in the [AWS CloudTrail User Guide](#).

Topics

- [CloudFront Information in CloudTrail Log Files \(p. 335\)](#)
- [Understanding CloudFront Log File Entries \(p. 336\)](#)

CloudFront Information in CloudTrail Log Files

When you enable CloudTrail, CloudTrail captures every request that you make to every AWS service that CloudTrail supports. (For a list of supported services, see [Supported Services](#) in the [AWS CloudTrail User Guide](#).) The log files aren't organized or sorted by service; each log file might contain records from more than one service. CloudTrail determines when to create a new log file.

Note

CloudTrail supports all CloudFront API actions.

Every log file entry contains information about who made the request. The user identity information in the log file helps you determine whether the request was made using root or IAM user credentials, using temporary security credentials for a role or federated user, or by another AWS service. For more information, see [userIdentity Element](#) in the [AWS CloudTrail User Guide](#).

You can store log files for as long as you want. You can also define Amazon S3 lifecycle rules to archive or delete log files automatically.

By default, your log files are encrypted by using Amazon S3 server-side encryption (SSE).

You can choose to have CloudTrail publish Amazon SNS notifications when new log files are delivered if you want to take quick action upon log file delivery. For more information, see [Configuring Amazon SNS Notifications](#) in the [AWS CloudTrail User Guide](#).

You can also aggregate log files from multiple AWS regions and multiple AWS accounts into a single Amazon SNS bucket. For more information, see [Aggregating CloudTrail Log Files to a Single Amazon S3 Bucket](#) in the [AWS CloudTrail User Guide](#).

Understanding CloudFront Log File Entries

Each JSON-formatted CloudTrail log file can contain one or more log entries. A log entry represents a single request from any source and includes information about the requested action, including any parameters, the date and time of the action, and so on. The log entries are not guaranteed to be in any particular order; they are not an ordered stack trace of API calls.

The `eventName` element identifies the action that occurred and the API version that was used to perform that action. For example, the following `eventName` value indicates that a web distribution was updated, and the 2014-01-31 API version was used to perform the action:

```
UpdateDistribution2014_01_31
```

The following example shows a CloudTrail log entry that demonstrates five actions:

- Updating a web distribution configuration. The value of `eventName` is `UpdateDistribution`.
- Listing web distributions that are associated with the current account. The value of `eventName` is `ListDistributions`.
- Getting the configuration for a specific web distribution. The value of `eventName` is `GetDistribution`.
- Creating an invalidation batch request. The value of `eventName` is `CreateInvalidation`.
- Listing origin access identities that are associated with the current account. The value of `eventName` is `ListCloudFrontOriginAccessIdentities`.

```
{
  "Records": [
    {
      "eventVersion": "1.01",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "A1B2C3D4E5F6G7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/smithj",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "smithj"
      },
      "eventTime": "2014-05-06T18:00:32Z",
      "eventName": "UpdateDistribution2014_01_31",
      "sourceIPAddress": "192.0.2.17",
      "userAgent": "aws-sdk-ruby/1.39.0 ruby/1.9.3 x86_64-linux",
      "requestParameters": {
        "id": "EDFDVBD6EXAMPLE",
        "ifMatch": "E9LHASXEXAMPLE",
        "distributionConfig": {
          "restrictions": {
            "geoRestriction": {
              "quantity": 0,
              "restrictionType": "none"
            }
          },
          "customErrorResponses": {
            "quantity": 0
          },
          "defaultRootObject": "index.html",
          "aliases": {
            "quantity": 1,
            "items": ["example.com"]
          }
        },
        "logging": {
          "bucket": ""
        }
      }
    }
  ]
}
```

```

        "enabled": false,
        "prefix": "",
        "includeCookies": false
    },
    "viewerCertificate": {
        "iamCertificateId": "A1B2C3D4E5F6G7EXAMPLE",
        "sslSupportMethod": "sni-only"
    },
    "callerReference": "2014-05-06 64832",
    "defaultCacheBehavior": {
        "targetOriginId": "Images",
        "allowedMethods": {
            "items": ["GET",
                      "HEAD"],
            "quantity": 2
        },
        "forwardedValues": {
            "cookies": {
                "forward": "none"
            },
            "queryString": false
        },
        "minTTL": 300,
        "trustedSigners": {
            "enabled": false,
            "quantity": 0
        },
        "viewerProtocolPolicy": "redirect-to-https",
        "smoothStreaming": false
    },
    "origins": {
        "items": [
            {
                "customOriginConfig": {
                    "HTTPSPort": 443,
                    "originProtocolPolicy": "http-only",
                    "HTTPPort": 80
                },
                "domainName": "myawsbucket.s3-website-us-east-2.amazonaws.com",
                "id": "Web page origin"
            },
            {
                "customOriginConfig": {
                    "HTTPSPort": 443,
                    "originProtocolPolicy": "http-only",
                    "HTTPPort": 80
                },
                "domainName": "myotherawsbucket.s3-website-us-west-2.amazonaws.com",
                "id": "Images"
            }
        ],
        "quantity": 2
    },
    "enabled": true,
    "cacheBehaviors": {
        "allowedMethods": {
            "items": ["GET",
                      "HEAD"],
            "quantity": 2
        },
        "trustedSigners": {
            "enabled": false,
            "quantity": 0
        },
        "targetOriginId": "Web page origin",
        "smoothStreaming": false,
        "viewerProtocolPolicy": "redirect-to-https",
        "minTTL": 300,
        "forwardedValues": {
            "cookies": {
                "forward": "none"
            },
            "queryString": false
        }
    }
}

```

```

        "forwardedValues": {
            "cookies": {
                "forward": "none"
            },
            "queryString": false
        },
        "pathPattern": "*.html"
    }],
    "quantity": 1
},
"priceClass": "PriceClass_All",
"comment": "Added an origin and a cache behavior"
},
{
    "responseElements": {
        "eTag": "E2QWRUHEXAMPLE",
        "distribution": {
            "domainName": "d11111abcdef8.cloudfront.net",
            "status": "InProgress",
            "distributionConfig": {
                distributionConfig response omitted
            },
            "id": "EDFDVBD6EXAMPLE",
            "lastModifiedTime": "May 6, 2014 6:00:32 PM",
            "activeTrustedSigners": {
                "quantity": 0,
                "enabled": false
            },
            "inProgressInvalidationBatches": 0
        }
    },
    "requestID": "4e6b66f9-d548-11e3-a8a9-73e33example",
    "eventID": "5ab02562-0fc5-43d0-b7b6-90293example"
},
{
    "eventVersion": "1.01",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "A1B2C3D4E5F6G7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user smithj",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "smithj"
    },
    "eventTime": "2014-05-06T18:01:35Z",
    "eventName": "ListDistributions2014_01_31",
    "sourceIPAddress": "192.0.2.17",
    "userAgent": "aws-sdk-ruby/1.39.0 ruby/1.9.3 x86_64-linux",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "52de9f97-d548-11e3-8fb9-4dad0example",
    "eventID": "eb91f423-6dd3-4bb0-a148-3cdfbexample"
},
{
    "eventVersion": "1.01",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "A1B2C3D4E5F6G7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user smithj",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "smithj"
    },
    "eventTime": "2014-05-06T18:01:59Z",
    "eventName": "GetDistribution2014_01_31",
    "sourceIPAddress": "192.0.2.17",

```

```

"userAgent": "aws-sdk-ruby/1.39.0 ruby/1.9.3 x86_64-linux",
"requestParameters": {
    "id": "EDFDVBD6EXAMPLE"
},
"responseElements": null,
"requestID": "497b3622-d548-11e3-8fb9-4dad0example",
"eventID": "c32289c7-005a-46f7-9801-cba41example"
},
{
    "eventVersion": "1.01",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "A1B2C3D4E5F6G7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/smithj",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "smithj"
    },
    "eventTime": "2014-05-06T18:02:27Z",
    "eventName": "CreateInvalidation2014_01_31",
    "sourceIPAddress": "192.0.2.17",
    "userAgent": "aws-sdk-ruby/1.39.0 ruby/1.9.3 x86_64-linux",
    "requestParameters": {
        "invalidationBatch": {
            "callerReference": "2014-05-06 64947",
            "paths": {
                "quantity": 3,
                "items": ["/images/new.jpg",
                           "/images/logo.jpg",
                           "/images/banner.jpg"]
            }
        },
        "distributionId": "EDFDVBD6EXAMPLE"
    },
    "responseElements": {
        "invalidation": {
            "createTime": "May 6, 2014 6:02:27 PM",
            "invalidationBatch": {
                "callerReference": "2014-05-06 64947",
                "paths": {
                    "quantity": 3,
                    "items": ["/images/banner.jpg",
                               "/images/logo.jpg",
                               "/images/new.jpg"]
                }
            },
            "status": "InProgress",
            "id": "ISRZ85EXAMPLE"
        },
        "location": "https://cloudfront.amazonaws.com/2014-01-31/distribution/EDFDVBD6EXAMPLE/invalidation/ISRZ85EXAMPLE"
    },
    "requestID": "4e200613-d548-11e3-a8a9-73e33example",
    "eventID": "191ebb93-66b7-4517-a741-92b0eexample"
},
{
    "eventVersion": "1.01",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "A1B2C3D4E5F6G7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/smithj",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "smithj"
    },
    "eventTime": "2014-05-06T18:03:08Z",

```

```
        "eventName": "ListCloudFrontOriginAccessIdentities2014_01_31",
        "sourceIPAddress": "192.0.2.17",
        "userAgent": "aws-sdk-ruby/1.39.0 ruby/1.9.3 x86_64-linux",
        "requestParameters": null,
        "responseElements": null,
        "requestID": "42ca4299-d548-11e3-8fb9-4dad0example",
        "eventID": "7aeb434f-eb55-4e2a-82d8-417d5example"
    }]
}
```

Tagging Amazon CloudFront Distributions

Tags are words or phrases that you can use to identify and organize your AWS resources. You can add multiple tags to each resource, and each tag includes a key and a value that you define. For example, the key might be "domain" and the value might be "example.com". You can search and filter your resources based on the tags you add.

When you apply tags to CloudFront distributions or other AWS resources (such as Amazon EC2 instances or Amazon S3 buckets) and activate the tags, AWS generates a cost allocation report as a comma-separated value (CSV file) with your usage and costs aggregated by your active tags. You can apply tags that represent business categories (such as cost centers, application names, or owners) to organize your costs across multiple services. For more information about using tags for cost allocation, see [Use Cost Allocation Tags](#) in the [AWS Billing and Cost Management User Guide](#).

For the current limit on the number of tags that you can add to a distribution, see [Limits \(p. 359\)](#). To request a higher limit, [create a case](#) with the AWS Support Center.

Note the following:

- You can tag web and RTMP distributions, but you can't tag origin access identities or invalidations.
- [Tag Editor](#) and [Resource Groups](#) are currently not supported for CloudFront.

You can also apply tags to resources by using the CloudFront API, AWS CLI, SDKs, and AWS Tools for Windows PowerShell. For more information, see the following documentation:

- CloudFront API – See [Actions on Tags](#) in the [Amazon CloudFront API Reference](#)
- AWS CLI – See [cloudfront](#) in the [AWS Command Line Interface Reference](#)
- SDKs – See the applicable SDK documentation on the [AWS Documentation](#) page
- Tools for Windows PowerShell – See [Amazon CloudFront](#) in the [AWS Tools for Windows PowerShell Reference](#)

Topics

- [Tag Restrictions \(p. 341\)](#)
- [Adding, Editing, and Deleting Tags for Distributions \(p. 342\)](#)

Tag Restrictions

The following basic restrictions apply to tags:

- Maximum number of tags per resource – 10
- Maximum key length – 128 Unicode characters
- Maximum value length – 256 Unicode characters
- Valid values for key and value – a-z, A-Z, 0-9, space, and the following characters: _ . : / = + - and @
- Tag keys and values are case sensitive
- Don't use `aws :` as a prefix for keys; it's reserved for AWS use

Adding, Editing, and Deleting Tags for Distributions

The following procedure explains how to add, edit, and delete tags for your distributions in the CloudFront console.

To add tags, edit, or delete tags for a distribution

1. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose the ID for the distribution that you want to update.
3. Choose the **Tags** tab.
4. Choose **Add or edit tags**.
5. On the Add or edit tags page, you can do the following:

Add a tag

Enter a key and, optionally, a value for the tag.

Edit a tag

Change the key, the value, or both. You can also delete the value for a tag, but the key is required.

Delete a tag

Choose the X on the right side of the value field.

6. Choose Save.

Troubleshooting

Troubleshoot common problems you might encounter with Amazon CloudFront and find possible solutions.

Topics

- [Troubleshooting Distribution Issues \(p. 343\)](#)
- [Troubleshooting Error Responses from Your Origin \(p. 345\)](#)
- [Troubleshooting Lambda@Edge \(p. 348\)](#)

Troubleshooting Distribution Issues

Use the information here to help you diagnose and fix access-denied issues or other common issues that you might encounter when working with Amazon CloudFront distributions.

I Can't View the Files in My Web Distribution

If you can't view the files in your CloudFront web distribution, see the following topics for some common solutions.

Did You Sign Up for Both CloudFront and Amazon S3?

To use Amazon CloudFront with an Amazon S3 origin, you must sign up for both CloudFront and Amazon S3, separately. For more information about signing up for CloudFront and Amazon S3, see [Getting Started with CloudFront \(p. 38\)](#).

Are Your Amazon S3 Bucket and Object Permissions Set Correctly?

If you are using CloudFront with an Amazon S3 origin, the original versions of your content are stored in an Amazon S3 bucket. The easiest way to use CloudFront with Amazon S3 is to make all your objects publicly readable in Amazon S3. To do this, you must explicitly enable public read privileges for each object that you upload to Amazon S3.

If your content is not publicly readable, you must create a CloudFront origin access identity so that CloudFront can access it. For more information about CloudFront origin access identities, see [Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content \(p. 211\)](#).

Object properties and bucket properties are independent. You must explicitly grant privileges to each object in Amazon S3. Objects do not inherit properties from buckets, and object properties must be set independently of the bucket.

Is Your Alternate Domain Name (CNAME) Correctly Configured?

If you already have an existing CNAME record for your domain name, update that record or replace it with a new one that points to your distribution's domain name.

Also, make sure that your CNAME record points to your distribution's domain name, not your Amazon S3 bucket. You can confirm that the CNAME record in your DNS system points to your distribution's domain name. To do so, use a DNS tool like dig. For information about dig, see <http://www.kloth.net/services/dig.php>.

The following example shows a dig request for a domain name called `images.example.com` and the relevant part of the response. Under ANSWER SECTION, see the line that contains CNAME. The CNAME record for your domain name is set up correctly if the value on the right side of CNAME is your CloudFront distribution's domain name. If it's your Amazon S3 origin server bucket or some other domain name, then the CNAME record is set up incorrectly.

```
[prompt]>
dig images.example.com

; <>> DiG 9.3.3rc2 <>> images.example.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15917
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 2, ADDITIONAL: 0
;; QUESTION SECTION:
;images.example.com. IN A
;; ANSWER SECTION:
images.example.com. 10800 IN CNAME d111111abcdef8.cloudfront.net.
...
...
```

For more information about CNAMEs, see [Using Alternate Domain Names \(CNAMEs\) \(p. 50\)](#).

Are You Referencing the Correct URL for Your CloudFront Distribution?

Make sure that the URL that you're referencing uses the domain name (or CNAME) of your CloudFront distribution, not your Amazon S3 bucket or custom origin.

Do You Need Help Troubleshooting a Custom Origin?

If you need AWS to help you troubleshoot a custom origin, we probably will need to inspect the `X-Amz-Cf-Id` header entries from your requests. If you are not already logging these entries, you might want to consider it for the future. For more information, see [Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins \(p. 85\)](#). For further help, see the [AWS Support Center](#).

I Can't View the Files in My RTMP Distribution

If you can't view the files in your RTMP distribution, are your URL and your playback client correctly configured? RTMP distributions require you to use an RTMP protocol instead of HTTP, and you must make a few minor configuration changes to your playback client. For information about creating RTMP distributions, see [Task List for Streaming Media Files Using RTMP \(p. 132\)](#).

Error Message: Certificate: <certificate-id> Is Being Used by CloudFront

Problem: You're trying to delete an SSL/TLS certificate from the IAM certificate store, and you're getting the message "Certificate: <certificate-id> is being used by CloudFront."

Solution: Every CloudFront web distribution must be associated either with the default CloudFront certificate or with a custom SSL/TLS certificate. Before you can delete an SSL/TLS certificate, you must either rotate the certificate (replace the current custom SSL/TLS certificate with another custom SSL/TLS certificate) or revert from using a custom SSL/TLS certificate to using the default CloudFront certificate. To do that, perform the procedure in the applicable section:

- [Rotating SSL/TLS Certificates \(p. 294\)](#)
- [Reverting from a Custom SSL/TLS Certificate to the Default CloudFront Certificate \(p. 295\)](#)

Troubleshooting Error Responses from Your Origin

If CloudFront requests an object from your Amazon S3 bucket or custom origin server and your origin returns an HTTP 4xx or 5xx status code, there's a problem with communication between CloudFront and your origin. The following sections describe common causes for selected HTTP status codes and provides some possible solutions.

Topics

- [HTTP 502 Status Code \(Bad Gateway\) \(p. 345\)](#)
- [HTTP 503 Status Code \(Service Unavailable\) \(p. 347\)](#)

HTTP 502 Status Code (Bad Gateway)

An HTTP 502 status code (Bad Gateway) indicates that CloudFront wasn't able to serve the requested object because it couldn't connect to the origin server.

Topics

- [SSL/TLS Negotiation Failure Between CloudFront and a Custom Origin Server \(p. 345\)](#)
- [Origin Is Not Responding with Supported Ciphers/Protocols \(p. 346\)](#)
- [SSL/TLS Certificate on the Origin Is Expired, Invalid, Self-signed, or the Certificate Chain Is in the Wrong Order \(p. 347\)](#)
- [Origin Is Not Responding on Specified Ports in Origin Settings \(p. 347\)](#)

SSL/TLS Negotiation Failure Between CloudFront and a Custom Origin Server

If you use a custom origin and you configured CloudFront to require HTTPS between CloudFront and your origin, the problem might be mismatched domain names. The SSL/TLS certificate that is installed on your origin includes a domain name in the **Common Name** field and possibly several more in the **Subject Alternative Names** field. (CloudFront supports wildcard characters in certificate domain names.) One of the domain names in the certificate must match one or both of the following values:

- The value that you specified for **Origin Domain Name** for the applicable origin in your distribution.
- The value of the `Host` header if you configured CloudFront to forward the `Host` header to your origin. For more information about forwarding the `Host` header to your origin, see [Configuring CloudFront to Cache Objects Based on Request Headers \(p. 152\)](#).

If the domain names don't match, the SSL/TLS handshake fails, and CloudFront returns an HTTP status code 502 (Bad Gateway) to the viewer.

To determine whether domain names in the certificate match the **Origin Domain Name** in the distribution or the `Host` header, you can use an online SSL checker or OpenSSL. If the domain names don't match, you have two options:

- Get a new SSL/TLS certificate that includes the applicable domain names.

If you use AWS Certificate Manager (ACM), see [Request a Certificate](#) in the *AWS Certificate Manager User Guide* to request a new certificate.

- Change the distribution configuration so CloudFront no longer tries to use SSL to connect with your origin.

Online SSL Checker

To find an SSL test tool, search the Internet for "online ssl checker." Typically, you specify the name of your domain, and the tool returns a variety of information about your SSL/TLS certificate. Confirm that the certificate contains your domain name in the **Common Names** or **Subject Alternative Names** fields.

OpenSSL

To determine whether CloudFront is able to establish a connection with your origin, you can use OpenSSL to try to make an SSL/TLS connection to your origin and to verify that the certificate on your origin is correctly configured. If OpenSSL is able to make a connection, it returns information about the certificate on the origin server.

The command that you use depends on whether you use a client that supports **SNI (Server Name Indication)**.

Client supports SNI

```
openssl s_client -connect domainname:443 -servername domainname
```

Client doesn't support SNI

```
openssl s_client -connect domainname:443
```

Replace *domainname* with the applicable value:

- **If you aren't forwarding the Host header to the origin** – Replace *domainname* with your origin's domain name.
- **If you are forwarding the Host header to the origin** – Replace *domainname* with the CNAME that you're using with your CloudFront distribution.

Origin Is Not Responding with Supported Ciphers/Protocols

CloudFront connects to origin servers using ciphers and protocols. For a list of the ciphers and protocols that CloudFront supports, see [Supported Ciphers and Protocols \(p. 296\)](#). If your origin does not respond with one of these ciphers or protocols in the SSL/TLS exchange, CloudFront fails to connect. You can validate that your origin supports the ciphers and protocols by using SSL Labs:

- **SSL Labs**

Type the domain name of your origin in the **Hostname** field, and then choose **Submit**. Review the **Common names** and **Alternative names** fields from the test to see if they match your origin's domain name.

After the test is finished, find the **Protocols** and **Cipher Suites** sections in the test results to see which ciphers or protocols are supported by your origin. Compare them with the list of [Supported Ciphers and Protocols \(p. 296\)](#).

Note

If you're using Elastic Load Balancing, see [SSL Security Policies for Elastic Load Balancing](#) in the *Elastic Load Balancing User Guide* to learn how to set the ciphers and protocols. Using the Predefined Security Policy *ELBSecurityPolicy-2016-08* gives CloudFront access to your elastic

load balancer. If you want to restrict it further using a custom policy, you must allow the ciphers that CloudFront supports.

SSL/TLS Certificate on the Origin Is Expired, Invalid, Self-signed, or the Certificate Chain Is in the Wrong Order

If the origin server returns the following, CloudFront drops the TCP connection, returns HTTP status code 502 (Bad Gateway), and sets the X-Cache header to `Error from cloudfront`:

- An expired certificate
- Invalid certificate
- Self-signed certificate
- Certificate chain in the wrong order

Note

If the full chain of certificates, including the intermediate certificate, is not present, CloudFront drops the TCP connection.

For information about installing an SSL/TLS certificate on your custom origin server, see [Requiring HTTPS for Communication Between CloudFront and Your Custom Origin \(p. 280\)](#).

Origin Is Not Responding on Specified Ports in Origin Settings

When you create an origin on your CloudFront distribution, you can set the ports that CloudFront connects to the origin with for HTTP and HTTPS traffic. By default, these are TCP 80/443. You have the option to modify these ports. If your origin is rejecting traffic on these ports for any reason, or if your backend server isn't responding on the ports, CloudFront will fail to connect.

To troubleshoot these issues, check any firewalls running in your infrastructure and validate that they are not blocking the supported IP ranges. For more information, see [AWS IP Address Ranges](#) in the *Amazon Web Services General Reference*. Additionally, verify whether your web server is running on the origin.

HTTP 503 Status Code (Service Unavailable)

An HTTP 503 status code (Service Unavailable) typically indicates a performance issue on the origin server. In rare cases, it indicates that CloudFront temporarily can't satisfy a request because of limited resources at an edge location.

Topics

- [Origin Server Does Not Have Enough Capacity to Support the Request Rate \(p. 347\)](#)
- [CloudFront Was Not Able to Resolve Your Origin Domain Due to DNS Issues \(p. 348\)](#)
- [CloudFront Caused the Error Due to Limited Resources at the Edge Location \(p. 348\)](#)

Origin Server Does Not Have Enough Capacity to Support the Request Rate

CloudFront generates this error when the origin server is overwhelmed with incoming requests. CloudFront then relays the error back to the user. To resolve this issue, try the following solutions:

- If you use Amazon S3 as your origin server, optimize the performance of Amazon S3 by following the best practices for key naming. For more information, see [Request Rate and Performance Considerations](#) in the *Amazon Simple Storage Service Developer Guide*.

- If you use Elastic Load Balancing as your origin server, see [503 Error Classic](#).
- If you use a custom origin, examine the application logs to ensure that your origin has sufficient resources, such as memory, CPU, and disk size. If you use Amazon EC2 as the backend, make sure that the instance type has the appropriate resources to fulfill the incoming requests. For more information, see [Instance Types](#) in the *Amazon EC2 User Guide for Linux Instances*.

CloudFront Was Not Able to Resolve Your Origin Domain Due to DNS Issues

When CloudFront receives a request for an object that is expired or is not stored in its cache, it makes a request to the origin to get the updated object. To make a successful request to the origin, CloudFront performs a DNS resolution on the origin domain name. However, when the DNS service that hosts your domain is experiencing issues, CloudFront cannot resolve the domain name to get the IP address, resulting in a 503 error. To fix this issue, contact your DNS provider, or, if you are using Amazon Route 53, see [Amazon Route 53 DNS](#).

To further troubleshoot this issue, ensure that the [authoritative name servers](#) of your origin's root domain or zone apex (such as example.com) are functioning correctly. Your authoritative name servers then receive the request and return the IP address that is associated with the domain, and are the same as the DNS servers that you used to set up your CloudFront distribution. Use the following commands to find the name servers for your apex origin:

```
dig OriginAPEXDomainName NS +short
nslookup -query=NS OriginAPEXDomainName
```

When you have the names of your name servers, use the following commands to query the domain name of your origin against them to make sure that each responds with an answer:

```
dig OriginDomainName @NameServerFromAbove
nslookup OriginDomainName NameServerFromAbove
```

CloudFront Caused the Error Due to Limited Resources at the Edge Location

You will receive this error in the rare case that CloudFront has the ability to route requests to the next best available edge location. In rare cases, CloudFront might be unable to satisfy the request. If this happens, contact [AWS Support](#).

This error is common when you perform load testing on your CloudFront distribution. To avoid this, follow the [Load Testing CloudFront \(p. 350\)](#) guidelines to avoid 503 (Capacity Exceeded) errors. This type of error typically means that the edge location didn't have enough capacity at the time of the request to serve the object.

Troubleshooting Lambda@Edge

If your Lambda@Edge function is returning one of the following errors, you're still using the preview version of Lambda@Edge:

- "The function has an invalid runtime ..."
- "The function code size is too big ..."
- "Invalid or unsupported region in the function ..."

To confirm that you're still using the preview version, check the following:

- Your existing functions are coded using the Edge Node.js 4.3 runtime, which is no longer supported.
- Your existing triggers reference the function only by name, which is no longer supported. Triggers must reference a numbered version of the function, for example:

arn:aws:lambda:us-east-1:123456789012:function:http-redirect:1

You can continue to use the preview version of Lambda@Edge for a distribution until you delete all the triggers that are associated with the distribution. To add new triggers to the distribution, you must migrate from the preview version to the general availability version; you can't revert to the preview version.

To migrate from the preview version to the general availability version of Lambda@Edge for a distribution

1. Use the CloudFront console to delete all the existing triggers for the distribution:
 - a. Sign in to the AWS Management Console and open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
 - b. Choose the ID for the distribution that you want to update.
 - c. Choose **Behaviors**.
 - d. Select the check box for the cache behavior that you want to delete triggers for, and choose **Edit**.
 - e. For **Lambda Function Associations**, choose the X next to each function association (trigger).
 - f. After you remove all triggers, choose **Yes, Edit**.
 - g. To delete triggers for other cache behaviors for this distribution, repeat steps 2d through 2f.

After you delete all triggers for a distribution and save your changes, the triggers that you add to the distribution provide the general availability functionality of Lambda@Edge instead of the preview functionality.

2. Use the Lambda console to create new versions of your functions. Note the following:
 - Create functions in the US East (N. Virginia) Region.
 - Create functions with the nodejs6.10 runtime property.
 - Update the event structure for headers. For more information, see [Event Structure \(p. 108\)](#).
 - For general information, see [Using CloudFront with Lambda@Edge \(p. 99\)](#).
 - For information about how to use the Lambda console to create functions, see [Creating a Lambda@Edge Function and Creating a Trigger for a CloudFront Event](#) in the *AWS Lambda Developer Guide*.
3. Use the Lambda console to add CloudFront triggers to the new functions. The Lambda console allows you to test the triggers as you create them; this option isn't available if you create triggers in the CloudFront console. For more information, see [Adding Triggers for a Lambda@Edge Function \(AWS Lambda Console\)](#) in the *AWS Lambda Developer Guide*.

Load Testing CloudFront

Traditional load testing methods don't work well with CloudFront because CloudFront uses DNS to balance loads across geographically dispersed edge locations and within each edge location. When a client requests content from CloudFront, the client receives a DNS response that includes a set of IP addresses. If you test by sending requests to just one of the IP addresses that DNS returns, you're testing only a small subset of the resources in one CloudFront edge location, which doesn't accurately represent actual traffic patterns. Depending on the volume of data requested, testing in this way may overload and degrade the performance of that small subset of CloudFront servers.

CloudFront is designed to scale for viewers that have different client IP addresses and different DNS resolvers across multiple geographic regions. To perform load testing that accurately assesses CloudFront performance, we recommend that you do all of the following:

- Send client requests from multiple geographic regions.
- Configure your test so each client makes an independent DNS request; each client will then receive a different set of IP addresses from DNS.
- For each client that is making requests, spread your client requests across the set of IP addresses that are returned by DNS, which ensures that the load is distributed across multiple servers in a CloudFront edge location.

CloudFront Streaming Tutorials

The following information and tutorials explain how to use CloudFront for live streaming, on-demand streaming, and RTMP streaming.

Live Streaming

- **Wowza Streaming Engine** – For information about using Wowza Streaming Engine and CloudFront for live streaming, see [How to bring your Wowza Streaming Engine license to CloudFront live HTTP streaming](#) on the Wowza website.
- [Live HTTP Streaming Using CloudFront and Any HTTP Origin \(p. 352\)](#)

On-Demand Streaming

Unified Streaming – For information about using Unified Streaming and CloudFront for on-demand streaming, see [Amazon Instances](#) on the Unified Streaming website.

RTMP Streaming

- [On-Demand Video Streaming Using CloudFront and JW Player \(p. 354\)](#)

Live HTTP Streaming Using CloudFront and Any HTTP Origin

With Amazon CloudFront live streaming, you can use any live encoder, such as Elemental Live, that outputs HTTP-based streams to stream live performances, webinars, and other events. This tutorial walks you through the process of configuring live streaming.

Note

This tutorial assumes that you have already set up your live encoder.

Topics

- [Creating a New CloudFront Distribution for Live Streaming \(p. 352\)](#)
- [Configuring Web Players to Play the Live Stream \(p. 353\)](#)
- [\(Optional\) Deleting an AWS CloudFormation Stack for Live Streaming \(p. 353\)](#)

Creating a New CloudFront Distribution for Live Streaming

The following procedure uses an AWS CloudFormation template to create a new CloudFront distribution that you will use for your live stream.

To create a new CloudFront distribution for live streaming

1. Go to the [Amazon Web Services](#) page, and sign in using your Amazon.com account or create a new account.
2. To start the wizard that creates a new CloudFront distribution, choose the following link:
[Create a CloudFront distribution for live HTTP streaming](#)
3. On the **Select Template** page, choose **Next**.
4. On the **Specify Details** page, type the following values:

Stack name

Type a descriptive name for your live streaming stack, or accept the default name.

CustomOriginDNSName

Type the DNS name of the custom origin that CloudFront will get the live feed from.

CustomOriginHTTPPort

Type the HTTP TCP port that you want CloudFront to use to communicate with your custom origin.

DistributionComment

Type a comment to help you identify your distribution, which is useful if you manage multiple distributions. The value that you type here appears in the CloudFront console.

5. Choose **Next**.
6. (Optional) On the **Options** page, add the key-value pairs for any tags you plan to use. In addition, to configure SNS notification, to specify how long you're willing to wait for the stack to be created, to choose whether to roll back changes if stack creation fails, and to enter a stack policy, choose **Advanced**, and adjust settings as desired. For more information, see [Setting AWS CloudFormation Stack Options](#).

7. Choose **Next**.
8. Review the settings for the stack, and then choose **Create**. AWS CloudFormation creates the stack. Creating your stack creation might take several minutes. To track the progress of stack creation, select the stack, and then choose the **Events** tab. If AWS CloudFormation cannot create the stack, the **Events** tab lists error messages.

When your stack is ready, in the list of stacks, the status for the stack changes to **CREATE_COMPLETE**.
9. When your stack is created, choose the **Outputs** tab, which will display the CloudFront domain name. You'll need this value when you set up the live stream playback in your web player.

Configuring Web Players to Play the Live Stream

To play the live stream, embed the manifest URL in the players that your users will play your live stream with. For example, to play a live stream for which the manifest file is **myStream/playlist.m3u8** and the CloudFront distribution is **d111111abcdef8.cloudfront.net**, you embed the following URL in players:

<http://d111111abcdef8.cloudfront.net/myStream/playlist.m3u8>

(Optional) Deleting an AWS CloudFormation Stack for Live Streaming

When your live event is over, delete the stack that you created for live streaming. This deletes the CloudFront distribution that you created for your live-streaming event.

Note

When you stop streaming, you also stop incurring CloudFront charges for this distribution. There are no charges for the AWS CloudFormation stack.

To delete an AWS CloudFormation stack for live streaming

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation>.
2. In the upper-right corner, choose the region in which you created your stack.
3. Select the stack, and then choose **Delete Stack**.
4. Choose **Yes, Delete** to confirm.
5. To track the progress of the stack deletion, select the stack, and then choose the **Events** tab.

On-Demand Video Streaming Using CloudFront and JW Player

When you stream media files using CloudFront, you provide both your media file and the media player with which you want end users to play the media file. To use the JW Player media player to stream media files with CloudFront, perform the procedures in the following topics:

1. [Uploading Media and JW Player Files to an Amazon S3 Bucket \(p. 354\)](#)
2. [Creating CloudFront Web and RTMP Distributions \(p. 355\)](#)
3. [Embedding Video in a Web Page \(p. 356\)](#)
4. [Uploading the HTML File and Playing the Video \(p. 358\)](#)

This tutorial is based on the free edition of JW Player version 7.3. For more information about JW Player, go to the [JW Player](#) website.

For more information about streaming media using CloudFront, see [Working with RTMP Distributions \(p. 131\)](#).

Uploading Media and JW Player Files to an Amazon S3 Bucket

You can upload your media files and your media player files to the same Amazon S3 bucket or to separate buckets. For this tutorial, you'll upload an .mp4 or .flv media file and the JW Player media player files to the same bucket.

To upload media and JW Player files to an Amazon S3 bucket

1. If you don't already have the files for the JW Player media player, download the player (JW Player 7) from the [Downloads](#) page on the JW Player website. Then extract the contents of the .zip file.
2. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
3. In the Amazon S3 console, choose **Create Bucket**.
4. In the **Create Bucket** dialog, enter a bucket name.

Important

For your bucket to work with CloudFront, the name must conform to DNS naming requirements. For more information, go to [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

5. Select a region for your bucket. By default, Amazon S3 creates buckets in the US Standard region. We recommend that you choose a region close to you to optimize latency, minimize costs, or to address regulatory requirements.
6. Choose **Create**.
7. Select your bucket in the **Buckets** pane, and choose **Upload**.
8. On the **Upload - Select Files** page, choose **Add Files**, and add the following files:
 - jwplayer.flash.swf
 - jwplayer.js
 - Your .mp4 or .flv media file.



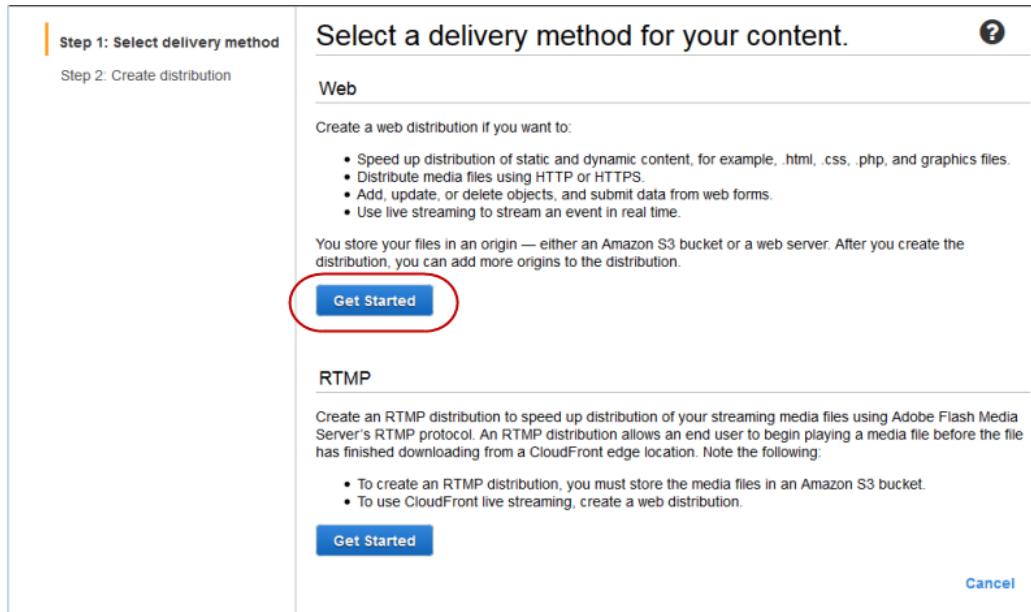
9. Grant public read permissions for the files that you added in the previous step.
 - a. Choose **Set Details**.
 - b. On the **Set Details** page, choose **Set Permissions**.
 - c. On the **Set Permissions** page, choose **Make everything public**.
10. Choose **Start Upload**.

Creating CloudFront Web and RTMP Distributions

To configure CloudFront to stream a media file, you need a CloudFront web distribution for the JW Player files and an HTML file, and an RTMP distribution for the media file. Perform the following two procedures to create a web distribution and an RTMP distribution.

To create a CloudFront web distribution for your JW Player files

1. Open the CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Choose **Create Distribution**.
3. On the first page of the **Create Distribution Wizard**, in the **Web** section, choose **Get Started**.



4. On the second page of the wizard, choose in the **Origin Domain Name** field, and select the Amazon S3 bucket that you created in the procedure [To upload media and JW Player files to an Amazon S3 bucket \(p. 354\)](#). If you have a lot of Amazon S3 buckets, you can type the first few characters of the bucket name to filter the list.
5. Accept the default values for the remaining fields, and choose **Create Distribution**.
6. After CloudFront creates your distribution, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. This should take less than 15 minutes.

The domain name that CloudFront assigns to your distribution appears in the list of distributions. (The domain name also appears on the Distribution Settings page for a selected distribution.)

To create a CloudFront RTMP distribution for your media file

1. In the CloudFront console, choose **Create Distribution**.
2. In the **Create Distribution Wizard**, in the **RTMP** section, choose **Get Started**.
3. On the second page of the wizard, choose in the **Origin Domain Name** field, and select the Amazon S3 bucket that you created in the procedure [To upload media and JW Player files to an Amazon S3 bucket \(p. 354\)](#). If you have a lot of Amazon S3 buckets, you can type the first few characters of the bucket name to filter the list.
4. Accept the default values for the remaining fields on the **Create Distribution** page, and choose **Create Distribution**.
5. After CloudFront creates your distribution, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. This should take less than 15 minutes.

The domain name that CloudFront assigns to your distribution appears in the list of distributions. (The domain name also appears on the Distribution Settings page for a selected distribution.)

Embedding Video in a Web Page

The following example shows you how to embed a video in a web page using the web and RTMP distributions that you created in [Creating CloudFront Web and RTMP Distributions \(p. 355\)](#).

Perform the following steps:

1. Sign in to the [JW Player](#) website. If you don't already have a JW Player account, create one.
2. On the [Downloads](#) page, get the license key for the player that you downloaded earlier in this tutorial.
3. Copy the HTML code below, and paste it into a text editor.
4. Review the comments in the HTML file, and replace the following placeholders with the applicable values:
 - WEB-DISTRIBUTION-DOMAIN-NAME
 - RTMP-DISTRIBUTION-DOMAIN-NAME
 - VIDEO-FILE-NAME
 - LICENSE-KEY
5. Save the file with a .html filename extension, for example, `jwplayer-example.html`.

```
<HTML>
<HEAD>
<TITLE>Amazon CloudFront Streaming with JW Player 7</TITLE>

<!-- Call the JW Player JavaScript file, jwplayer.js.
Replace WEB-DISTRIBUTION-DOMAIN-NAME with the domain name of your
CloudFront web distribution, for example, d1234.cloudfront.net
(begins with "d"). This causes a browser to download the JW Player file
before streaming begins.

Replace LICENSE-KEY with your personal license key from JW Player.
-->

<script type='text/javascript' src='WEB-DISTRIBUTION-DOMAIN-NAME/jwplayer.js'></script>
<script type='text/javascript' src='WEB-DISTRIBUTION-DOMAIN-NAME/jwplayer.flash.swf'></script>
<script>jwplayer.key="LICENSE-KEY";</script>

</HEAD>

<BODY>
<H1>This video is streamed by CloudFront and played by JW Player 7.</H1>

<!-- Replace RTMP-DISTRIBUTION-DOMAIN-NAME with the domain name of your
RTMP distribution, for example, s5678.cloudfront.net (begins with "s").

Replace VIDEO-FILE-NAME with the name of your .mp4 or .flv video file,
including the .mp4 or .flv filename extension. For example, if you uploaded
my-vacation.mp4, enter my-vacation.mp4. You might need to prepend "mp4:" to the
name of your video file, for example, mp4:my-vacation.mp4.

If the file is in a subdirectory, include the subdirectory name just before
the file name, for example:

"rtmp://RTMP-DISTRIBUTION-DOMAIN-NAME/cfx/st/sample-directory/VIDEO-FILE-NAME"
-->

<div id="my-video"></div>
<script type="text/javascript">
jwplayer("my-video").setup({
  'file': 'rtmp://RTMP-DISTRIBUTION-DOMAIN-NAME/cfx/st/VIDEO-FILE-NAME',
  'width': '720',
  'height': '480',
  'primary': 'flash',
  'autostart': 'true'
});
</script>
```

```
</BODY>  
</HTML>
```

Uploading the HTML File and Playing the Video

To play the video using the HTML file that you created in [Embedding Video in a Web Page \(p. 356\)](#), upload the file to your Amazon S3 bucket, and use the URL for your CloudFront distribution.

To upload the HTML file and play the video

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. Select your bucket, and choose **Upload**.
3. On the **Upload - Select Files** page, choose **Add Files**, and add your HTML file.
4. Grant public read permissions for the HTML file that you added in the previous step.
 - a. Choose **Set Details**.
 - b. On the **Set Details** page, choose **Set Permissions**.
 - c. On the **Set Permissions** page, choose **Make everything public**.
5. Choose **Start Upload**.
6. To play the video, enter the following URL in a web browser:
http://domain name of your CloudFront web distribution/your HTML file name

Limits

CloudFront entities are subject to the following limits.

Topics

- [General Limits \(p. 359\)](#)
- [General Limits on Web Distributions \(p. 359\)](#)
- [Limits on Whitelisted Cookies \(Web Distributions Only\) \(p. 360\)](#)
- [Limits on Whitelisted Query Strings \(Web Distributions Only\) \(p. 360\)](#)
- [Limits on Custom Headers \(Web Distributions Only\) \(p. 361\)](#)
- [Limits on SSL Certificates \(Web Distributions Only\) \(p. 361\)](#)
- [Limits on Invalidations \(p. 361\)](#)
- [Limits on Field-Level Encryption \(p. 362\)](#)
- [Limits on Lambda@Edge \(p. 362\)](#)
- [Limits on RTMP Distributions \(p. 363\)](#)

General Limits

| Entity | Limit |
|---|---|
| Data transfer rate per distribution | 40 Gbps Request a higher limit |
| Requests per second per distribution | 100,000 Request a higher limit |
| Tags that can be added to a CloudFront web or RTMP distribution | 50 |
| Objects that you can serve per distribution | Unlimited |
| Maximum length of a request, including headers and query strings | 20,480 bytes |
| Maximum length of a URL | 8,192 bytes |
| Active CloudFront key pairs for trusted signers | 2 |
| For more information, see Specifying the AWS Accounts That Can Create Signed URLs and Signed Cookies (Trusted Signers) (p. 216) . | |

General Limits on Web Distributions

| Entity | Limit |
|--|--|
| Web distributions per AWS account | 200 |
| For more information, see Working with Web Distributions (p. 59) . | Request a higher limit |

| Entity | Limit |
|--|--|
| Maximum file size for HTTP GET, POST, and PUT requests | 20 GB |
| Response timeout per origin For more information, see Origin Response Timeout (Amazon EC2, Elastic Load Balancing, and Other Custom Origins Only) (p. 69) . | 4-60 seconds Request a higher limit |
| File compression: range of file sizes that CloudFront compresses For more information, see Serving Compressed Files (p. 179) . | 1,000 to 10,000,000 bytes |
| Alternate domain names (CNAMEs) per distribution For more information, see Using Alternate Domain Names (CNAMEs) (p. 50) . | 100 Request a higher limit |
| Origins per distribution | 25 Request a higher limit |
| Cache behaviors per distribution | 25 Request a higher limit |

Limits on Whitelisted Cookies (Web Distributions Only)

| Entity | Limit |
|---|--|
| Whitelisted cookies per cache behavior For more information, see Configuring CloudFront to Cache Objects Based on Cookies (p. 150) . | 10 Request a higher limit |
| Total number of bytes in whitelisted cookie names (doesn't apply if you configure CloudFront to forward all cookies to the origin) | 512 minus the number of whitelisted cookies |

Limits on Whitelisted Query Strings (Web Distributions Only)

| Entity | Limit |
|---|--|
| Maximum number of characters in a whitelisted query string | 128 characters |
| Maximum number of characters total for all whitelisted query strings in the same parameter | 512 characters |
| Whitelisted query strings per cache behavior For more information, see Configuring CloudFront to Cache Based on Query String Parameters (p. 147) . | 10 Request a higher limit |

Limits on Custom Headers (Web Distributions Only)

| Entity | Limit |
|---|---|
| Whitelisted headers per cache behavior For more information, see Configuring CloudFront to Cache Objects Based on Request Headers (p. 152) . | 10 Request a higher limit |
| Custom headers: maximum number of custom headers that you can configure CloudFront to forward to the origin For more information, see Forwarding Custom Headers to Your Origin (Web Distributions Only) (p. 156) . | 10 name/value pairs Request a higher limit |
| Custom headers: maximum length of a header name | 256 characters |
| Custom headers: maximum length of a header value | 1,783 characters |
| Custom headers: maximum length of all header values and names combined | 10,240 characters |

Limits on SSL Certificates (Web Distributions Only)

| Entity | Limit |
|--|---------------------------------|
| SSL certificates per AWS account when serving HTTPS requests using dedicated IP addresses (no limit when serving HTTPS requests using SNI) For more information, see Using HTTPS with CloudFront (p. 278) . | 2 Request a higher limit |
| SSL certificates that can be associated with a CloudFront web distribution | 1 |

Limits on Invalidations

| Entity | Limit |
|--|-----------|
| Object invalidation: maximum number of objects allowed in active invalidation requests, excluding wildcard invalidations For more information, see Invalidate Objects (Web Distributions Only) (p. 165) . | 3,000 |
| Object invalidation: maximum number of active wildcard invalidations allowed | 15 |
| Object invalidation: maximum number of objects that one wildcard invalidation can process | Unlimited |

Limits on Field-Level Encryption

| Entity | Limit |
|---|-------|
| Maximum length of a field to encrypt For more information, see Using Field-Level Encryption to Help Protect Sensitive Data (p. 85) . | 16 KB |
| Maximum number of fields in a request body when field-level encryption encryption is configured | 10 |
| Maximum length of a request body when field-level encryption is configured | 1 MB |
| Maximum number of field-level encryption configurations that can be associated with one AWS account | 10 |
| Maximum number of field-level encryption profiles that can be associated with one AWS account | 10 |
| Maximum number of public keys that can be added to one AWS account | 10 |
| Maximum number of fields to encrypt that can be specified in one profile | 10 |
| Maximum number of CloudFront distributions that can be associated with a field-level encryption configuration | 20 |
| Maximum number of query argument profile mappings that can be included in a field-level encryption configuration | 5 |

Limits on Lambda@Edge

Due to the constrained execution environment, Lambda@Edge has limits that go beyond the default Lambda limits (described in the [Limits](#) section of the *AWS Lambda Developer Guide*).

In addition, be aware that there are some other restrictions when using Lambda@Edge functions. For more information, see [Requirements and Restrictions on Lambda Functions \(p. 101\)](#).

Limits that differ by event-type

| Entity | Origin request and response event limits | Viewer request and response event limits |
|--|--|--|
| Function resource allocation | Same as Lambda limits | 128 MB |
| Function timeout. The function can make network calls to resources such as Amazon S3 buckets, DynamoDB tables, or Amazon EC2 instances in AWS Regions. | 30 seconds | 5 seconds |
| Size of a response that is generated by a Lambda function, including headers and body | 1 MB | 40 KB |
| Maximum compressed size of a Lambda function and any included libraries | 50 MB | 1 MB |

Other limits

| Entity | Limit |
|--|--|
| Distributions per AWS account that you can create triggers for | 25 Request a higher limit |
| Triggers per distribution | 25 Request a higher limit |
| Requests per second | 10,000 Request a higher limit |
| Concurrent executions For more information, see Lambda Function Concurrent Executions in the AWS Lambda Developer Guide . | 1000 Request a higher limit |

Limits on RTMP Distributions

| Entity | Limit |
|--|--|
| RTMP distributions per AWS account | 100 |
| For more information, see Working with RTMP Distributions (p. 131) . | Request a higher limit |

Amazon CloudFront Resources

Although fairly simple to use, CloudFront is rich in functionality. The resources listed here can help you learn more about CloudFront.

Topics

- [Additional Amazon CloudFront Documentation \(p. 364\)](#)
- [Getting Support \(p. 364\)](#)
- [CloudFront Developer Tools and SDKs \(p. 365\)](#)
- [Using CloudFront Logging \(p. 365\)](#)
- [Additional Tips from the Amazon Web Services Blog \(p. 365\)](#)
- [Invalidating Objects \(p. 365\)](#)
- [Distributing Streaming Media \(p. 366\)](#)
- [Tools and Code Examples for Configuring Private Content \(p. 366\)](#)
- [Using CloudFront with a Content Management System \(p. 366\)](#)

Additional Amazon CloudFront Documentation

The following related resources can help you as you work with this service.

- [Amazon CloudFront API Reference](#) – Gives complete descriptions of the API actions, parameters, and data types, and a list of errors that the service returns.
- [Document History \(p. 368\)](#) – A high-level overview of current and previous releases with specific attention to new features, corrections, known issues, and documentation improvements.
- [Technical documentation for the Amazon Simple Storage Service \(S3\)](#) – A detailed discussion of the Amazon S3 service, including the basics of getting started, an overview of the service, a programming reference, and an API reference.
- [Amazon CloudFront product information](#) – The primary web page for information about CloudFront, including features and pricing information.
- [Terms of Use](#) – Detailed information about our copyright and trademark; your account, license, and site access; and other topics.

Getting Support

Support for CloudFront is available in a number of forms.

- [Discussion forums](#) – A community-based forum for developers to discuss technical questions related to CloudFront.
- [AWS Support Center](#) – This site brings together information about your recent support cases and results from AWS Trusted Advisor and health checks, as well as providing links to discussion forums, technical FAQs, the service health dashboard, and information about AWS support plans.
- [AWS Premium Support Information](#) – The primary web page for information about AWS Premium Support, a one-on-one, fast-response support channel to help you build and run applications on AWS Infrastructure Services.
- [Contact Us](#) – Links for inquiring about your billing or account. For technical questions, use the discussion forums or support links above.

CloudFront Developer Tools and SDKs

See the [Developer Tools](#) page for links to developer resources that provide documentation, code samples, release notes, and other information to help you build innovative applications with AWS.

In addition, Amazon Web Services provides software development kits for accessing CloudFront programmatically. The SDK libraries automate a number of common tasks, including cryptographically signing your service requests, retrying requests, and handling error responses.

- [AWS SDK for Java](#) – Setup and other documentation
- [AWS SDK for .NET](#) – Setup and other documentation
- [AWS SDK for PHP](#) – Setup and other documentation
- [AWS SDK for Ruby](#) – Setup and other documentation

Using CloudFront Logging

The following AWS blog posts discuss enhancements to CloudFront logging as well as some ways to analyze access logs.

- AWS Blog – [Amazon CloudFront Request Logging](#) (for content delivered via HTTP)
- AWS Blog – [Amazon CloudFront Now Supports Streaming Access Logs](#) (for content delivered via RTMP)
- AWS Blog – [Enhanced CloudFront Logs, Now With Query Strings](#)

Additional Tips from the Amazon Web Services Blog

The AWS Blog has a number of posts to help you use CloudFront:

- Improving website performance – [Improving Global Application Performance](#)
- Creating secure connections using HTTPS – [Amazon CloudFront: HTTPS Access, Another Edge Location, Price Reduction](#)
- Using custom origins – [New Amazon CloudFront Feature: Custom Origins](#)
- Learning more about third-party tools for Amazon CloudFront – [CloudFront Management Tool Roundup](#)

Invalidating Objects

In addition to the invalidation methods provided by CloudFront, you can use the following third-party tools to invalidate objects.

Note

These tools were developed by third-party vendors who are not associated with Amazon Web Services. For information on how to use these tools, please refer to the vendor's documentation or contact the vendor.

- CloudBerry Explorer – <http://cloudberrylab.com>
- Ylastic – <http://ylastic.com>
- Cyberduck – <http://cyberduck.io>

- CloudFront Invalidator – <https://github.com/swook/jQuery-CloudFront-Invalidator>
- CDN Planet CloudFront Purge Tool – <http://www.cdnplanet.com/tools/cloudfront-purge-tool/>

You can also search for code samples on Github, <https://github.com>. Search for the phrase *CloudFront invalidation*.

Distributing Streaming Media

The following third-party sources provide additional information on distributing streaming media.

- StreamingMedia.com – [How To Get Started With Amazon CloudFront Streaming](#)
- IONCANNON.NET –
 - [iPhone Windowed HTTP Live Streaming Using Amazon S3 and CloudFront Proof of Concept](#)
 - [HTTP Live Video Stream Segmenter and Distributor](#)
 - [iPhone Windowed HTTP Live Streaming Server](#)
- Flowplayer.org – [Flowplayer demos for streaming video](#)
- JW Player – [About RTMP Streaming](#)

Tools and Code Examples for Configuring Private Content

In addition to the methods provided by CloudFront, the following third-party tools provide web forms for configuring your distribution for private content. Some of the tools also provide web forms for creating signed URLs.

- **CloudBerry** – Supports configuring a distribution for private content and supports creating signed URLs.

For information about using [CloudBerry](#) for CloudFront private content, go to [How to Configure Private Content for CloudFront Streaming with CloudBerry](#).

For information on setting a default root object, see [How to set CloudFront Default Object with CloudBerry S3 Explorer](#).

For more information about private content, see the blog post [New Amazon CloudFront Feature: Private Content](#) on the AWS blog.

For an example of how to use signed cookies, use your domain name in object URLs, and still use the SSL certificate for the cloudfront.net domain, see the Space Vatican blog post [Using CloudFront Signed Cookies](#). This allows you to use an alternate domain name with HTTPS without incurring the expense of dedicated IP addresses or the limitations of SNI, as documented in [Using Alternate Domain Names and HTTPS \(p. 283\)](#).

Using CloudFront with a Content Management System

You can use CloudFront with several popular content management systems. The following links tell you how.

Drupal

- Drupal.org – [CloudFront Installation](#)
- DrupalModules.com – [CloudFront Drupal Module](#)

WordPress

- om4.com – [Using Amazon CloudFront with WordPress and WordPress MU](#)
- WordPress.org – [W3 Total Cache](#)
- WordPress.org – [Simple Amazon S3 Upload Form](#)
- WordPress.org – [OSSDL CDN Off-linker](#)
- WordPress.org – [My CDN](#)
- Inquisiter.com – [Amazon CloudFront CDN with a WordPress Blog](#)

Document History

The following table describes the important changes to the documentation since the last release of CloudFront.

- **API Version:** 2016-11-25
- **Latest documentation update:** November 30, 2016

| Change | Description | Date Changed |
|--------------|--|--------------------|
| New Features | You can now Lambda edge locations in conjunction with CloudFront as Lambda@Edge. For more information, see Using CloudFront with Lambda@Edge (p. 99) . Regional Edge Caches are edge locations between your web server and global edge locations. For more information, see How CloudFront Works with Regional Edge Caches (p. 6) . | November 30, 2016 |
| New Feature | You can now secure communication between viewers and CloudFront, and between CloudFront and ELB load balancers, by importing third-party certificates. For more information, see Using HTTPS with CloudFront (p. 278) . | October 13, 2016 |
| New Feature | The account that you use to create or update a web or RTMP distribution now requires additional permissions if you want to enable access logs for the distribution. For more information, see Permissions Required to Configure Logging and to Access Your Log Files (p. 316) . | September 19, 2016 |
| New Feature | You can now choose the HTTP version, HTTP/2 or HTTP/1.1, that you want viewers to use to communicate with CloudFront. Viewers use the latest version that you configure CloudFront to use. If you choose HTTP/2, viewers that don't support HTTP/2 automatically use an earlier version. For more information, see the description for the Supported HTTP Versions (p. 80) field. | September 7, 2016 |
| New Feature | For web distributions, you can now choose the query string parameters that CloudFront uses as a basis for caching your objects. For more information, see Configuring CloudFront to Cache Based on Query String Parameters (p. 147) . | August 30, 2016 |
| New Feature | You can now assign tags, which are commonly used for cost allocation, to CloudFront web and RTMP distributions. For more information, see Tagging Amazon CloudFront Distributions (p. 341) . | August 9, 2016 |
| New Feature | You can now configure CloudFront to use SSL/TLS certificates that you provisioned by using the new AWS Certificate Manager service. Note that CloudFront still supports using certificates that you obtained from a third-party certificate authority and uploaded to the IAM certificate store. For more information, see Using HTTPS with CloudFront (p. 278) . To specify an SSL/TLS certificate by using the CloudFront API, use the new <code>Certificate</code> and <code>CertificateSource</code> elements. For more information, see the | January 21, 2016 |

| Change | Description | Date Changed |
|--------------|---|-------------------|
| | descriptions of these elements in the DistributionConfig Complex Type in the <i>Amazon CloudFront API Reference</i> . | |
| New Features | <p>For web distributions, you can now further protect communication between CloudFront and your origin server:</p> <ul style="list-style-type: none"> • Enforce HTTPS-only connection between CloudFront and your origin web server – You can configure CloudFront to connect to your origin server using HTTPS regardless of whether the viewer made the request by using HTTP or HTTPS. • Support for TLSv1.1 and TLSv1.2 between CloudFront and your origin web server – CloudFront now supports TLSv1.1 and TLSv1.2 for communication between CloudFront and your origin. In addition, you can choose the protocols that you want CloudFront to use when communicating with your origin so you can, for example, choose not to allow CloudFront to communicate with your origin by using SSLv3, which is less secure than TLS. <p>For more information, see Using HTTPS with CloudFront (p. 278).</p> | January 13, 2016 |
| New Feature | <p>For web distributions, you can now configure CloudFront to include custom headers when it forwards requests to your origin. Custom headers have a variety of uses, such as the following:</p> <ul style="list-style-type: none"> • You can distinguish the requests that are forwarded to your custom origin by CloudFront from requests that come from other sources. • If you've configured more than one CloudFront distribution to use the same origin, you can distinguish between the requests that CloudFront forwards for each distribution. • You can use custom headers to control access to content on a custom origin. <p>For more information, see Forwarding Custom Headers to Your Origin (Web Distributions Only) (p. 156).</p> | December 28, 2015 |
| New Feature | <p>For web distributions, you can now configure CloudFront to automatically compress files of certain types for both Amazon S3 and custom origins, so downloads are faster and your web pages render faster. Compression also reduces your CloudFront data transfer cost because you pay for the total amount of data served. For more information, see Serving Compressed Files (p. 179).</p> | December 17, 2015 |

| Change | Description | Date Changed |
|-------------|--|-----------------|
| New Feature | <p>You can now integrate CloudFront with AWS WAF, a web application firewall that lets you monitor the HTTP and HTTPS requests that are forwarded to CloudFront and lets you control access to your content. Based on conditions that you specify, such as the IP addresses that requests originate from or the values of query strings, CloudFront responds to requests either with the requested content or with an HTTP 403 status code (Forbidden). You can also configure CloudFront to return a custom error page when a request is blocked.</p> <p>For more information about AWS WAF, see the AWS WAF Developer Guide. For information about how to add the ID for an AWS WAF web ACL to a CloudFront distribution, see AWS WAF Web ACL (p. 78) in the topic Values That You Specify When You Create or Update a Web Distribution (p. 63).</p> | October 6, 2015 |
| New Feature | <p>CloudFront access logs for web distributions now include four new columns:</p> <ul style="list-style-type: none"> • x-forwarded-for – The originating IP address of the client that made a request. • ssl-protocol – The SSL protocol (for example, <code>TLSv1.1</code>) that the client and CloudFront negotiated for transmitting the request and response. • ssl-cipher – The SSL cipher (for example, <code>ECDHE-RSA-AES128-GCM-SHA256</code>) that the client and CloudFront negotiated for transmitting the request and response. • x-edge-response-result-type – The classification of the response (for example, <code>Hit</code> or <code>Miss</code>) just before CloudFront begins to return the response to the viewer. In some cases, this value can differ from the value of the existing log field <code>x-edge-result-type</code>, which shows how CloudFront classified the response after the last byte left the edge location. <p>For more information about access logs, see Access Logs (p. 315). For a complete listing of all of the fields that appear in access logs for web distributions, see Web Distribution Log File Format (p. 319).</p> | June 30, 2015 |
| New Feature | <p>You can now configure a maximum time-to-live (TTL) and a default TTL to specify how long CloudFront caches your objects in edge locations. You can set the TTL at the edge when your origin doesn't include a <code>Cache-Control max-age</code>, <code>Cache-Control s-maxage</code>, or <code>Expires</code> header in the response, or you can override the header value. For more information, see Specifying How Long Objects Stay in a CloudFront Edge Cache (Expiration) (p. 160).</p> | June 17, 2015 |
| New Feature | <p>The invalidation feature, which allows you to remove objects from CloudFront edge caches before they expire, now supports the <code>*</code> wildcard character. You can add a <code>*</code> wildcard character at the end of an invalidation path to remove all objects that match this path. For more information, see Invalidating Objects (Web Distributions Only) (p. 165).</p> | May 21, 2015 |

| Change | Description | Date Changed |
|--------------|--|----------------|
| New Features | <p>This release of CloudFront includes the following improvements to CloudFront reports:</p> <ul style="list-style-type: none"> • A new Devices report lets you see what types of devices your users are using to view your content. • You can now download the data for CloudFront reports in comma-separated values (CSV) format. • You can now view CloudWatch metrics in the CloudFront console, which lets you more easily view all of the metrics for a distribution. • For Devices, Browsers, and Operating Systems reports, we created separate categories for requests from bots and crawlers, for requests from custom viewers, and for requests for which the value of the User-Agent header is empty. • The Popular Objects report now lets you view up to 500 characters of the URL for each object. <p>For more information about CloudFront reports, see CloudFront Reports (p. 15).</p> | March 25, 2015 |
| New Feature | <p>When you have Amazon S3 buckets in regions that require signature version 4 for authentication and you're using an origin access identity to restrict access to your Amazon S3 bucket, you can now submit <code>PUT</code> requests to CloudFront to upload objects to your bucket. For more information, see Using an Origin Access Identity in Amazon S3 Regions that Support Only Signature Version 4 Authentication (p. 215).</p> | March 18, 2015 |
| New Features | <p>This release of CloudFront introduces the following new features:</p> <ul style="list-style-type: none"> • For web distributions, you can now use signed cookies instead of signed URLs to control who can access your content. Signed cookies are useful when you don't want to change your current URLs or when you want to provide access to multiple restricted files, for example, all of the files in the subscribers' area of a website. For information about using signed cookies to protect your private content, see Serving Private Content through CloudFront (p. 207). • For web distributions, you can configure CloudFront to cache different versions of your objects based on the device a user is using to view your content. With this release, we add support for caching a different version of your objects when the device is a smart TV. For more information, see User-Agent Header (p. 199). | March 12, 2015 |

| Change | Description | Date Changed |
|--------------|---|-------------------|
| New Features | <p>This release of CloudFront introduces the following new features:</p> <ul style="list-style-type: none"> • Adding a path to the origin – For web distributions, you can now specify a path in addition to a domain name when you configure the origin. For example, if you're using an Amazon S3 bucket as your origin, you can specify <code>bucket-name.s3.amazonaws.com/production</code> instead of just <code>bucket-name.s3.amazonaws.com</code>. This allows you to use a single bucket to serve content for multiple distributions. This feature works both for Amazon S3 origins and for custom origins. For information about specifying an origin path in the CloudFront console, see Origin Path (p. 67). • Top-referrers report and viewer reports – For web distributions, you can now display a list of the top referrers. You can also display information about the viewers that are accessing your content, including the browsers that your users are using, the operating systems that the browsers are running on, and the locations of viewers. For information about these reports, see CloudFront Top Referrers Report (p. 23) and CloudFront Viewers Reports (p. 30). | December 15, 2014 |
| New Feature | <p>For web distributions, you can now choose the minimum SSL protocol version, SSLv3 or TLSv1, that you want CloudFront to allow when responding to requests from your users. If a user is using a browser or device that doesn't support the minimum protocol version that you specify, CloudFront won't serve your objects to the user. For information about choosing the minimum SSL protocol version in the CloudFront console, see Minimum SSL Protocol Version (p. 80).</p> | October 24, 2014 |
| New Features | <p>This release of CloudFront introduces the following new features:</p> <ul style="list-style-type: none"> • Cache Statistics charts – You can now view a graphical representation of statistics related to CloudFront edge locations. The following statistics are available for a specified time period over the last 60 days: total number of requests; hits, misses, and errors as a percentage of total requests; total bytes transferred to viewers and bytes transferred for cache misses; viewer requests by HTTP status code (2xx, 3xx, 4xx, and 5xx); and the percentage of GET requests that didn't finish downloading. For more information, see CloudFront Cache Statistics Reports (p. 16). • Faster delivery of access logs – CloudFront access log files are now delivered several times per hour, and the files are available within an hour of viewer requests. For more information about access logs, see Access Logs (p. 315). • Popular Objects report – The Popular Objects report lists the number of requests, cache hits, and cache misses, as well as error rates for the 50 most popular objects during a specified period. For more information, see CloudFront Popular Objects Report (p. 20). | October 21, 2014 |

| Change | Description | Date Changed |
|--------------|---|--------------------|
| New Feature | For web distributions, you can now monitor six CloudFront metrics in near real time using CloudFront. This lets you quickly spot trends in usage and availability. You can also set alarms based on the metrics, so you can get notification when a specific event occurs. For more information, see Monitoring CloudFront Activity Using CloudWatch (p. 330) . | October 9, 2014 |
| New Features | <p>This release of CloudFront introduces the following new features:</p> <ul style="list-style-type: none"> • For web distributions, you can now configure CloudFront to cache the response to OPTIONS requests. The response includes information about the options provided by a web server and can apply to a specific resource or to the server as a whole. For more information, see Allowed HTTP Methods (p. 73) and Cached HTTP Methods (p. 73). • For web distributions, when you configure CloudFront to forward whitelisted cookies to your origin and to cache objects based on cookie values, you can now use * and ? wildcards in cookie names. For more information, see Configuring CloudFront to Cache Objects Based on Cookies (p. 150). | September 29, 2014 |
| New Feature | CloudFront now supports more ciphers for forwarding HTTPS requests to custom origin servers. For more information, see Encryption (p. 193) . | August 20, 2014 |
| New Feature | For web distributions, CloudFront lets you choose whether you want CloudFront to forward headers to your origin and to cache separate versions of a specified object based on the header values in viewer requests. This allows you to serve different versions of your content based on the device the user is using, the location of the viewer, the language the viewer is using, and a variety of other criteria. For more information, see Configuring CloudFront to Cache Objects Based on Request Headers (p. 152) . | June 26, 2014 |
| New Feature | Amazon CloudFront now works with AWS CloudTrail to capture information about every request that your AWS account (including your IAM users) sends to the CloudFront API. Integrating CloudFront and CloudTrail lets you determine which requests were made to the CloudFront API, the source IP address from which each request was made, who made the request, when it was made, and more. For more information about using CloudFront with CloudTrail, see Using AWS CloudTrail to Capture Requests Sent to the CloudFront API (p. 335) . | May 28, 2014 |
| New Feature | With this release, for HTTPS viewer requests that CloudFront forwards to a custom origin, CloudFront validates that one of the domain names in the SSL certificate on your origin server matches the domain name that you specify for Origin Domain Name . If the domain names don't match, CloudFront responds to viewer requests with an HTTP status code 502 (bad gateway) instead of the requested objects. To enable this functionality, you must specify an Origin Protocol Policy of Match Viewer . For more information, see Using HTTPS with CloudFront (p. 278) . | May 16, 2014 |

| Change | Description | Date Changed |
|--------------|---|-------------------|
| New Feature | This release of CloudFront introduces a new field in CloudFront access logs for web distributions. The <code>time-taken</code> field shows the number of seconds between the time a CloudFront edge server receives a viewer's request and the time that CloudFront writes the last byte of the response to the server's output queue as measured on the server. For more information about the file format of CloudFront access logs for web distributions, see Web Distribution Log File Format (p. 319) . | April 28, 2014 |
| New Feature | This release of CloudFront introduces usage charts that contain a subset of data from the CloudFront usage report. For more information, see CloudFront Usage Reports (p. 26) . | March 13, 2014 |
| New Features | <p>This release of CloudFront introduces the following new features:</p> <ul style="list-style-type: none"> • Redirect viewer HTTP requests to HTTPS: You can now configure CloudFront to redirect viewer HTTP requests to HTTPS. For more information, see Using HTTPS with CloudFront (p. 278). • Server Name Indication (SNI) for HTTPS requests with alternate domain names: If you're using your domain name in the URLs for your objects, you can now configure CloudFront to serve HTTPS requests to users whose browsers support Server Name Indication (SNI). For more information, see Using Alternate Domain Names and HTTPS (p. 283). | March 5, 2014 |
| New Feature | This release of CloudFront introduces support for HTTP on-demand streaming of media files in the Microsoft Smooth Streaming format. For more information, see Configuring On-Demand Smooth Streaming (p. 96) . | February 20, 2014 |
| New Feature | <p>This release of CloudFront introduces support for HTTP 1.1. For more information, see Transfer Encoding (p. 202).</p> <p>In addition, we added documentation about on-demand progressive downloads and on-demand Apple HTTP live streaming. For more information, see Configuring On-Demand Progressive Downloads and Configuring On-Demand Apple HTTP Live Streaming (HLS) in the <i>Amazon CloudFront Developer Guide</i>.</p> | February 7, 2014 |
| New Features | <p>This release of CloudFront introduces geo restriction. If you need to prevent users in selected countries from accessing your content, you can configure a CloudFront web distribution to do one of the following:</p> <ul style="list-style-type: none"> • Allow users to access content only if they're in a whitelist of specified countries. • Prevent users from accessing content if they're in a blacklist of specified countries. <p>For more information, see Restricting the Geographic Distribution of Your Content (p. 94).</p> | December 18, 2013 |

| Change | Description | Date Changed |
|-----------------------|---|--------------------|
| New Features | <p>This release of CloudFront introduces the following features:</p> <ul style="list-style-type: none"> • DELETE, OPTIONS, PATCH, POST, and PUT support: You can now use the <code>DELETE</code>, <code>OPTIONS</code>, <code>PATCH</code>, <code>POST</code>, and <code>PUT</code> HTTP methods in requests that you send to CloudFront. For more information, see Allowed HTTP Methods (p. 73). <p>For information about how to specify HTTP methods by using the CloudFront API, see <code>Method</code> in the topic DistributionConfig Complex Type in the <i>Amazon CloudFront API Reference</i>.</p> <ul style="list-style-type: none"> • Distribution types renamed: CloudFront download distributions are now known as web distributions, and streaming distributions are now known as RTMP distributions. • New columns in access logs for web distributions: Access logs for CloudFront web distributions now include three additional columns for each request: <code>x-host-header</code>, <code>cs-protocol</code>, and <code>cs-bytes</code>. For more information, see Web Distribution Log File Format (p. 319). | October 15, 2013 |
| New Features | <p>This release of CloudFront introduces the following features:</p> <ul style="list-style-type: none"> • Custom error pages: You can now serve error pages with your own branding and content instead of the default HTTP error messages, such as "404, page not found." You can also use custom error pages to serve a static page when your web server is unavailable. For more information, see Customizing Error Responses (p. 172). <p>For information about how to specify custom error pages by using the CloudFront API, see <code>CustomErrorResponse</code>s in the topic DistributionConfig Complex Type in the <i>Amazon CloudFront API Reference</i>.</p> <ul style="list-style-type: none"> • Configurable cache duration for error responses: Also known as error caching minimum TTL, this feature lets you specify how long you want CloudFront to cache each error at CloudFront edge locations. CloudFront previously cached all error responses for five minutes; now you can specify any duration and thereby control how frequently CloudFront checks with your origin after an error. For more information, see Customizing Error Responses (p. 172). <p>For information about how to specify the cache duration for error responses by using the CloudFront API, see <code>CustomErrorResponse</code>s in the topic DistributionConfig Complex Type in the <i>Amazon CloudFront API Reference</i>.</p> | September 23, 2013 |
| New Feature | You can now include the <code>*</code> wildcard in a CloudFront alternate domain name (CNAME), such as <code>*.example.com</code> . This is useful when you want to route all requests for objects in a domain and its subdomains to a CloudFront distribution. For more information, see Using Alternate Domain Names (CNAMEs) (p. 50) . | September 18, 2013 |
| Updated Documentation | Documentation about live streaming with Wowza Media Server 3.6 was added. | September 10, 2013 |

| Change | Description | Date Changed |
|--------------|---|--------------------|
| New Features | <p>This release of CloudFront introduces the following features:</p> <ul style="list-style-type: none"> • Authentication with AWS Signature Version 4: If you are using CloudFront API version 2013-05-12 or later, you must authenticate requests by using AWS Signature version 4. For more information, see Authenticating REST Requests in the <i>Amazon CloudFront API Reference</i>. • SSL for CloudFront alternate domain names: CloudFront now supports using HTTPS and using your own domain name in the URLs for your objects (for example, <code>http://www.example.com/image.jpg</code>). For more information, see Using Alternate Domain Names and HTTPS (p. 283). <p>In addition, a simultaneous release of Amazon Route 53 introduces the following CloudFront-related feature:</p> <ul style="list-style-type: none"> • Route 53 aliases to CloudFront distributions: Amazon Route 53 now supports creating alias resource record sets that route DNS queries to alternate domain names for CloudFront distributions. You can use this feature both for alternate domain names at the zone apex (<code>example.com</code>) and alternate domain names for subdomains (<code>www.example.com</code>). For more information, see Routing Queries to an Amazon CloudFront Distribution in the <i>Amazon Route 53 Developer Guide</i>. | June 11, 2013 |
| New Features | <p>This release of CloudFront introduces the following features:</p> <ul style="list-style-type: none"> • Fields for private content in the AWS Management Console: Settings for private content, which previously could be configured or changed only using the CloudFront API, can now be configured or changed in the AWS Management Console. This includes settings for origin access identities and trusted signers. In addition, the documentation about private content was reorganized and clarified. <p>For more information, see Serving Private Content through CloudFront (p. 207).</p> <ul style="list-style-type: none"> • Improvements to the AWS Management Console: Wizards and dialog boxes in the AWS Management Console have been resized to simplify viewing on tablet computers without compromising the appearance for other viewers. In addition, the number of pages in the Create Distribution wizard was reduced to simplify the process of creating a new distribution. | September 27, 2012 |

| Change | Description | Date Changed |
|--------------|---|-------------------|
| New Features | <p>This release of CloudFront introduces the following features:</p> <ul style="list-style-type: none"> • Access log improvements for web distributions: For web distributions, CloudFront access logs now include fields for: <ul style="list-style-type: none"> • The cookie header in each viewer request, including name-value pairs and attributes. This field is optional. • The result type of a request (for example, Hit, RefreshHit, or Miss). • An identifier that uniquely identifies each request (the CloudFront request ID). <p>For more information, see Web Distribution Log File Format (p. 319).</p> <p>For information about how to configure a CloudFront distribution to include cookies in access logs by using the CloudFront API, see IncludeCookies in the topic DistributionConfig Complex Type in the Amazon CloudFront API Reference.</p> <ul style="list-style-type: none"> • Cookie support for web distributions: You can now choose whether you want CloudFront to forward cookies and the associated cookie attributes to your origin. If so, you can also choose whether to forward all cookies or just a selected list of cookies. For more information, see Configuring CloudFront to Cache Objects Based on Cookies (p. 150). <p>For information about how to configure a CloudFront distribution to forward cookies to your origin by using the CloudFront API, see Cookies in the topic DistributionConfig Complex Type in the Amazon CloudFront API Reference.</p> <ul style="list-style-type: none"> • Price classes for web and RTMP distributions: You can now choose a price class that corresponds with the maximum price that you want to pay for CloudFront service. If you're willing to accept higher latency for your viewers in some geographic regions in return for lower cost, you can choose a price class that doesn't include all CloudFront regions. For more information, see Choosing the Price Class for a CloudFront Distribution (p. 55). <p>For information about how to specify the price class for a CloudFront web distribution by using the CloudFront API, see PriceClass in the topic DistributionConfig Complex Type in the Amazon CloudFront API Reference.</p> <p>For information about how to specify the price class for a CloudFront RTMP distribution by using the CloudFront API, see PriceClass in the topic StreamingDistributionConfig Complex Type in the Amazon CloudFront API Reference.</p> | September 5, 2012 |

| Change | Description | Date Changed |
|-----------------------|---|----------------|
| New Features | <p>This release of CloudFront introduces the following features:</p> <ul style="list-style-type: none"> • You can now invalidate objects using the CloudFront console. For more information, see Invalidating Objects (Web Distributions Only) (p. 165). • The CloudFront console was updated to better support viewing on tablet devices. | June 22, 2012 |
| New Features | <p>This release of CloudFront introduces the following features for web distributions:</p> <ul style="list-style-type: none"> • You can forward query strings to your origin. For more information, see Configuring CloudFront to Cache Based on Query String Parameters (p. 147). • You can specify up to 10 origins. For more information, see Values That You Specify When You Create or Update a Web Distribution (p. 63). • You can specify path patterns. For more information, see Values That You Specify When You Create or Update a Web Distribution (p. 63). <p>For information about how to specify these values by using the CloudFront API, see DistributionConfig Complex Type in the Amazon CloudFront API Reference.</p> <p>In addition, the CloudFront console has been updated. For more information, see Task List for Creating a Web Distribution (p. 59) and Task List for Streaming Media Files Using RTMP (p. 132).</p> <p>The <i>Amazon CloudFront Getting Started Guide</i> was merged into the <i>Amazon CloudFront Developer Guide</i>, and the <i>Amazon CloudFront Developer Guide</i> was reorganized to enhance usability.</p> | May 13, 2012 |
| Updated Documentation | The documentation about working with objects was reorganized and clarified. For the revised documentation, see Working with Objects (p. 143) . | April 4, 2012 |
| New Feature | <p>This release of CloudFront reduces the minimum TTL value for a web distribution. If you don't specify a minimum TTL when you create a distribution, CloudFront sets the minimum TTL to 0 seconds. For more information, see the following documentation:</p> <ul style="list-style-type: none"> • Specifying How Long Objects Stay in a CloudFront Edge Cache (Expiration) (p. 160) • The <code>CachingBehavior</code> element in the topic DistributionConfig Complex Type in the Amazon CloudFront API Reference. | March 15, 2012 |

| Change | Description | Date Changed |
|-------------|--|-------------------|
| New Feature | <p>This release of CloudFront introduces AWS Management Console support for creating a distribution with a custom origin, restricting your distribution to HTTPS exclusively, and specifying a default root object. For more information, go to the Amazon CloudFront product page or see any of the following topics in the <i>Amazon CloudFront Developer Guide</i>:</p> <ul style="list-style-type: none"> • Task List for Creating a Web Distribution (p. 59) • Using HTTPS with CloudFront (p. 278) • Specifying a Default Root Object (Web Distributions Only) (p. 177) | April 27, 2011 |
| New Feature | <p>This release of CloudFront includes integration with AWS Identity and Access Management (IAM). For more information, see Authentication and Access Control for CloudFront (p. 300).</p> | March 10, 2011 |
| New Feature | <p>This release of CloudFront includes new APIs to support custom origins. For more information, go to the Amazon CloudFront product page or Task List for Creating a Web Distribution (p. 59) in the <i>Amazon CloudFront Developer Guide</i>.</p> | November 9, 2010 |
| New Feature | <p>This release of CloudFront includes new APIs for object invalidation. For more information, go to the Amazon CloudFront product page or Invalidate Objects (Web Distributions Only) (p. 165) in the <i>Amazon CloudFront Developer Guide</i>.</p> | August 31, 2010 |
| New Feature | <p>CloudFront now supports the ability to assign a default root object to your distribution. For more information, see Specifying a Default Root Object (Web Distributions Only) (p. 177).</p> | August 5, 2010 |
| New Feature | <p>Access logging for HTTP distributions now includes a field for query string parameters. For more information, see Web Distribution Log File Format (p. 319).</p> | July 14, 2010 |
| New Feature | <p>Added support for secure connections using HTTPS. For more information, see Using HTTPS with CloudFront (p. 278).</p> | June 7, 2010 |
| New Feature | <p>Added logging for RTMP content. For more information, see RTMP Distribution Log File Format (p. 327).</p> | May 13, 2010 |
| New Feature | <p>Reduced the minimum amount of time an object can be on an edge server from 24 hours to 1 hour. The default, however, remains 24 hours. For more information, see Specifying How Long Objects Stay in a CloudFront Edge Cache (Expiration) (p. 160).</p> | April 13, 2010 |
| New Feature | <p>Added feature to serve private streaming content over a Real-Time Messaging Protocol (RTMP) and prevent the content from being downloaded. For more information, see Serving Private Content through CloudFront (p. 207).</p> | March 28, 2010 |
| New Feature | <p>Added feature to deliver streaming content over a Real-Time Messaging Protocol (RTMP) connection. For more information, see Task List for Streaming Media Files Using RTMP (p. 132).</p> | December 15, 2009 |
| New Feature | <p>Added feature to restrict access to your content delivered over HTTP. For more information, see Serving Private Content through CloudFront (p. 207).</p> | November 11, 2009 |

| Change | Description | Date Changed |
|-----------|--|-------------------|
| New Guide | We've separated the API reference material into its own guide. The <i>Amazon CloudFront Developer Guide</i> contains general information about how to use CloudFront, and the Amazon EC2 Auto Scaling API Reference contains detailed information about the control API requests, responses, and errors. | November 11, 2009 |

AWS Glossary

For the latest AWS terminology, see the [AWS Glossary](#) in the *AWS General Reference*.