

# Optimizing Cloudflare

Best Practices for Cloudflare's Integrated Global Cloud Platform

# Agenda

---

## Optimizing Cloudflare

Day 1 Review

Onboarding Best Practices

Security Best Practices

Performance Best Practices

---

# Instructor



Chrisanthy Carlane  
Partner Technical Enablement

# ACE - What we covered on Day 1

- Configured Cloudflare's Enterprise domain using CNAME setup at partner's bootcamp account.
- Understood the difference between Full setup and CNAME setup.
- Can guide the customer on the safe steps on onboarding.
- Can use tools like dig, curl, mtr, traceroute, chrome extensions to see if Cloudflare is well configured.
- Understood and can guide the customer on DNS/SSL decisions.



# Cloudflare ACE Training Webinar

Lab Handbook Online: <https://ace-training.cf/>

## Optimization: Prerequisite 1

- You have **a website you onboarded** on day 1.
- You have access to your Cloudflare demo account.

If not yet, please take **the day 1: implementation** webinar first.



# Cloudflare ACE Training Webinar

Lab Handbook Online: <https://ace-training.cf/>

## Optimization: Prerequisite 2

Please make sure you have terminal-friendly environment (MacOS, Linux) so we can play together with some basic terminal commands such as **dig**, **curl** etc.

Time: 5 ~ 15 mins

Prerequisite is also illustrated at the handbook: <https://ace-training.cf/>



// Best Practices: Onboarding

# Starting out with Cloudflare

1. Configure origin firewalls to allow access to web application from Cloudflare IPs
2. Set up Cloudflare Standalone Health Checks
3. Review HTTP headers and cookies added by Cloudflare
4. Restore your originating IP Address
5. Configure alerts
6. Enforce 2-Factor Authentication for your organization
7. Integrate with Customer SSO (if required)
8. Manage your brand by customizing Cloudflare error pages
9. Enable Cloudflare Logpush (or Logpull)

# 1. Allow access only from Cloudflare IPs

## Cloudflare IPs

- Configure firewalls to prevent access to your servers, load balancers, and other infrastructure from non-Cloudflare IP addresses to avoid attackers bypassing Cloudflare protection.
- This means allow-listing [Cloudflare IPs](#) in your Access Control List to prevent rate-limiting or false positives from any intrusion detection systems.
  - `cloudflare.com/ips`
  - API Call `/get ips`
- Prevents attackers from recording/recognizing the “fingerprints” of your hardware when probing your IPs

*Tips:  
If customer environment  
has multiple security  
appliances  
e.g F5, PaloAlto, etc.  
Make sure to whitelist CF IPs  
on all appliances*

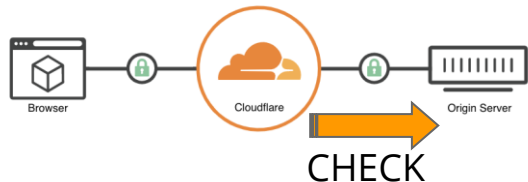
```
curl -X GET "https://api.cloudflare.com/client/v4/ips"
```



## 2. Set up Cloudflare Standalone Health Checks

Set up a criteria of health checks and get alerted when origin server is unreachable from Cloudflare edge.  
(under Traffic - Health Checks)

### Standalone Health Checks



### Health Checks

Monitor the health of your origin by creating a Health Check.


You have used 20 of 1000 Health Checks.

[Create](#)

[API](#) [Help](#)

### Manage Health Checks

Enable, disable, modify or delete configured Health Checks.

Status	Name	Failures last 24hr	Enabled
Healthy	https_test	0	<span>On</span> <span>↔</span> <span>⚙</span> <span>✕</span>
Healthy	19	0	<span>On</span> <span>↔</span> <span>⚙</span> <span>✕</span>
Healthy	21	0	<span>On</span> <span>↔</span> <span>⚙</span> <span>✕</span>
Healthy	Fail_every_10	 1.26k	<span>On</span> <span>↔</span> <span>⚙</span> <span>✕</span>

# 3. Review HTTP headers and cookies added by Cloudflare

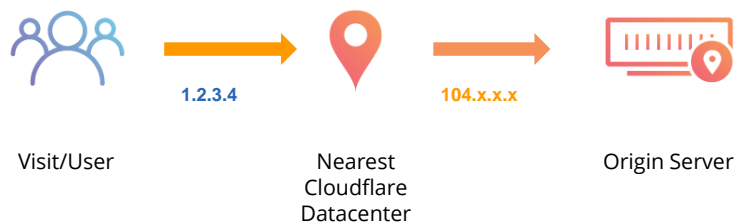
Cloudflare also provides custom headers and cookies for debugging, visitor information and improved security.

Header	How we use it	Example value
<b>CF-Request-Id</b> <i>(to be discontinued)</i>	This Cloudflare specific header is a unique identifier for every request passing through Cloudflare and is used primarily for debugging.	13b9eb04dff503dc...001
<b>CF-IPCountry</b>	The standard identifying header for the originating protocol of an HTTP request. Cloudflare may communicate with a web server using HTTP even if the request to the reverse proxy is HTTPS.	US
Cookie	How we use it	Example value
<b>cfduid</b>	This cookie is used by Cloudflare to apply security decisions to users that may share the same IP address (such as in a coffee shop). It does not correspond to any user id, nor does the cookie store any personally identifiable information.	d88dfb702206c2326978 0....115252

## 4. Restore the originating IP address

### Originating IPs

- HTTP requests will be coming from Cloudflare, instead of the actual users. Cloudflare adds “CF-Connecting-IP” and standard “X-Forwarded-For” headers to all requests.



- Nginx, Apache, and IIS configs to switch the logged IP are available.
  - [Restoring original visitor IPs - Option 1: Installing mod\\_cloudflare](#)
  - [Restoring original visitor IPs - Option 2: Installing mod\\_remoteip with Apache](#)
  - [Restoring original visitor IPs: Logging visitor IP addresses with mod\\_cloudflare](#)
  - [How does Cloudflare handle HTTP Request headers?](#)

## 4. Restore the originating IP address

### Nginx

```
log_format main ' "$http_cf_connecting_ip" $remote_addr - $remote_user [$time_local] '
                '$request' $status $body_bytes_sent "$http_referer" '
                '$http_user_agent' "$http_x_forwarded_for" ';
```

### Apache

```
LogFormat \
    "\ "%{CF-Connecting-IP}i\" %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\" \" \" \
    combined
```

# Example

```
cc@DESKTOP-MBTMLKR:~$ curl http://httpbin.greyccloud.cf/headers
{
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "gzip",
    "Cdn-Loop": "cloudflare",
    "Cf-Connecting-Ip": "115.66.104.22",
    "Cf-Ipcountry": "SG",
    "Cf-Ray": "610d975b1dd01897-SIN",
    "Cf-Request-Id": "079c5cecee000018970339a000000001",
    "Cf-Visitor": "{\\"scheme\\":\\"http\\"}",
    "Connection": "Keep-Alive",
    "Host": "httpbin.greyccloud.cf",
    "True-Client-Ip": "115.66.104.22",
    "User-Agent": "curl/7.58.0"
  }
}
```

```
cc@ccarlane-sg:~$ sudo nginx -V
nginx version: nginx/1.10.3 (Ubuntu)
built with OpenSSL 1.0.2g 1 Mar 2016
TLS SNI support enabled

configure arguments: --with-cc-opt='-g -O2 -fPIE -fstack-protector-strong -Wformat -Werror=format-security -Wdate-time
D_FORTIFY_SOURCE=2' --with-ld-opt='-Wl,-Bsymbolic-functions -fPIE -pie -Wl,-z,relro -Wl,-z,now' --prefix=/usr/share/nginx
x --conf-path=/etc/nginx/nginx.conf --http-log-path=/var/log/nginx/access.log --error-log-path=/var/log/nginx/error.log
--lock-path=/var/lock/nginx.lock --pid-path=/run/nginx.pid --http-client-body-temp-path=/var/lib/nginx/body --http-fastc
gi-temp-path=/var/lib/nginx/fastcgi --http-proxy-temp-path=/var/lib/nginx/proxy --http-scgi-temp-path=/var/lib/nginx/scg
i --http-uwsgi-temp-path=/var/lib/nginx/uwsgi --with-debug --with-pcre-jit --with-ipv6 --with-http_ssl_module --with-http
p_stub_status_module --with-http_realip_module --with-http_auth_request_module --with-http_addition_module --with-http_d
av_module --with-http_geoip_module --with-http_gunzip_module --with-http_gzip_static_module --with-http_image_filter_mod
ule --with-http_v2_module --with-http_sub_module --with-http_xslt_module --with-stream --with-stream_ssl_module --with-m
ail --with-mail_ssl_module --with-threads
```

```
set_real_ip_from 2400:cb00::/32;
set_real_ip_from 2606:4700::/32;
set_real_ip_from 2803:f800::/32;
set_real_ip_from 2405:b500::/32;
set_real_ip_from 2405:8100::/32;
set_real_ip_from 2c0f:f248::/32;
set_real_ip_from 2a06:98c0::/29;
real_ip_header CF-Connecting-IP;
#real_ip_header X-Forwarded-For;

    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
                    '$status $body_bytes_sent "$http_referer" '
                    '"$http_user_agent" "$http_x_forwarded_for" "$http_cf_connecting_ip" "$http_cf_ray"';
    access_log /var/log/nginx/access.log main;
    error_log /var/log/nginx/error.log;
```

# 5. Configure Alerts

All Notifications

Destinations

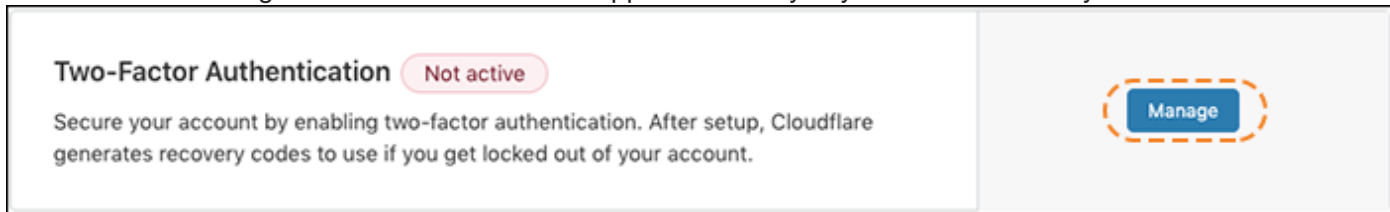
<div>Notifications</div> <div>Create notifications for your Cloudflare account(s).</div>			<div>Create</div>	
Name	Event	Method	Enabled	
Passive Origin Monitoring	Passive Origin Monitoring	Email	<div></div>	<div>Edit   Delete</div>
DDoS Notification	HTTP DDoS Attack Alerter	Email	<div></div>	<div>Edit   Delete</div>

## 6. Enforce Two-Factor Auth across an Account

### 2FA Enablement

Two-factor authentication (2FA) allows user account owners to add an additional layer of login security to Cloudflare accounts. This additional authentication step requires you to provide both something you know, such as a Cloudflare password, and something you have, such as an authentication code from a mobile device or security key ([Knowledge Base](#))

1. Log in to the Cloudflare dashboard.
2. Under the My Profile dropdown, click **My Profile**.
3. Click the **Authentication** tab.
4. Click **Manage** in the Two-Factor Authentication card.
5. Configure either a TOTP mobile app or a security key to enable 2FA on your account.





# 7. Configure SSO Login

## SSO Setup

Cloudflare Enterprise customers with Premium Success Support Level can reach out to their Customer Success Manager to learn how to start using SSO to log-in to the Cloudflare dashboard. If you are interested in using SSO yourself and becoming a Cloudflare Enterprise customer, then please get in touch.

### Log in to Cloudflare

Email

Password

**Log In With SSO**

## 8. Protect your brand with custom error pages

### Custom Error Pages

Cloudflare uses a wide range of [error codes](#) to identify issues in handling request traffic. By default, these error pages mention Cloudflare; however, custom error pages help you provide a consistent brand experience for your users and avoid confusion.

#### Account Level

1. Log into your Cloudflare account.
2. Click the **Configurations** tab (Account-level error pages)
3. In the left navigation, click **Custom Pages**.
4. Identify your desired custom error page type, then click the **Custom Pages** button. A **Custom Page** dialog will appear.
5. Enter the URL of the custom error page you customized in your origin server, then click **Publish**.

#### Domain Level

Domain-level custom error pages can be configured within the domain's settings on the **Custom Pages** tab.

## 9. Enable Cloudflare Logpush

Logpush can be configured for Enterprise customers using one of the following methods:

- [Cloudflare UI \(Dashboard\)](#)
- [Cloudflare Logpush API](#)

Cloud Services supported by Cloudflare Logpush:

- [Enable Amazon S3](#)
- [Enable Google Cloud Storage](#)
- [Enable Microsoft Azure](#)
- [Enable Sumo Logic](#)
- [IBM Cloud Object Storage](#)

Full Documentation: <https://developers.cloudflare.com/logs/logpush/>

# ACE - Exercise - Best Practice - #errorpage

1. At Cloudflare dashboard, go to TLS/SSL - Overview and select “Strict (SSL only origin pull)”.
2. At the browser, visit your website: <https://www.yourdomain.cf/>
3. At the browser, access <http://www.ace-training.cf/error.html>
4. At Cloudflare dashboard, visit Custom Pages - 500 Class Errors, and try publishing the page: <http://www.ace-training.cf/error.html>
5. Give Cloudflare some time to provision, then visit your website again: <https://www.yourdomain.cf/> and confirm the difference.



# ACE - Exercise - Best Practice - #Alerts

1. At Cloudflare dashboard, go to Account Home - Notification.
2. Try setting 'HTTP DDoS Attack Alert' and 'Passive Origin Monitoring' to your work email.
3. Once alert is set, you will get alerted on HTTP DDoS events, and origin health (even if no active health check) based on limited set of passive error codes. (e.g. 521)



# ACE - Exercise - Best Practice - #nmap

1. Try access <https://pentest-tools.com/network-vulnerability-scanning/tcp-port-scanner-online-nmap>
2. Enter the test origin address **35.234.81.115** and run a light scan. Check the result.

After finishing the exercise, think about the answer to:

- a. Is this origin opening the web port to ANY?
- b. Even if the origin is set like this, if it's behind Cloudflare, is it immune to DDoS and security threats?



# ACE - Onboarding Summary

- Set up health checks
- Set up necessary notification email
- Use custom error pages
- Consider Cloudflare dashboard 2FA or SSO
- Origin to allow-list Cloudflare IPs (and block everything else)
- Review CF headers/cookies, restore the originating IP if needed
- Enable Cloudflare Logpush (or Logpull)



# Security Best Practices



# Security Best Practices



1. Secure origin IP addresses
2. Always use HTTPS
3. Enable Web Application Firewall
4. Configure your Security Level selectively
5. Set Rate Limiting Rules
6. Control bots visiting your site
7. Optimize application security with Firewall Rules

# 1. Secure Origin IP Addresses

## Proxy all DNS Records from your origin

When a subdomain is orange-clouded within our DNS application, Cloudflare will actively proxy that traffic by responding with Cloudflare IP addresses. These addresses cause the client to connect to Cloudflare first and obscure the origin IP address and provide Cloudflare solutions. To improve the security of your origin IP address, all HTTP(s) traffic should be orange-clouded to avoid exposure of your origin IP.

Origin Access Control can also be set to only allow Cloudflare IPs.

grey-cloud	points to 1.2.3.4	Automatic	
orange-cloud	points to 1.2.3.4	Automatic	

## 2. Always use HTTPS

Remove HTTP-only path and endpoint, using Cloudflare's easy-to-use TLS

### ✓ Your SSL/TLS encryption mode is Full (strict)

This setting was last changed a few seconds ago



Learn more about [End-to-end encryption with Cloudflare](#)

[API](#) [Help](#)

### Always Use HTTPS

Redirect all requests with scheme "http" to "https". This applies to all http requests to the zone.

This setting was last changed a few seconds ago

On [↔](#)

[API](#) [Help](#)

### 3. Enable Web Application Firewall

<https://support.cloudflare.com/hc/en-us/articles/200172016-Understanding-the-Cloudflare-Web-Application-Firewall-WAF->

1. False positives can be reduced by setting your WAF in a logging mode and reviewing results after 1 to 2 weeks,
2. Cloudflare Specials and any platform-specific Rule groups your application may be using (Flash, Drupal, PHP) should be activated to begin.

#### Package: Cloudflare Rule Set

Make sure "Specials" is always enabled, and otherwise only enable rulesets for technologies you actually use.

3. Once your packages are configured, you can safely activate the WAF.

# ACE - Exercise - Security - #WAF

1. From a browser or terminal/POSTMAN, try access:  
[http://staging.yourdomain.cf/file.php?cmd=echo\(shell\\_exec\(%22ls%20/etc/var%22\)\)](http://staging.yourdomain.cf/file.php?cmd=echo(shell_exec(%22ls%20/etc/var%22)))
2. At Cloudflare dashboard, turn the WAF on.
3. Try the exploit again and check the result.
4. At Cloudflare dashboard, check the security logs.

You can do the similar thing for the customer using security scanning tools like SQLmap.



## 4. Configure a selective Security Level

Cloudflare's security can be applied differently across different paths

Cloudflare sees nearly 2 billion unique IPs every month from more than 25M+ million websites on our network. Each IP is assigned a threat score, and as an IP engages in malicious behavior on our network, its threat score increases. The scale of this data, and the automatic process of assigning threat scores, allows Cloudflare to quickly find bad actors and prevent them from reaching your assets like login pages or pages with sensitive information.

1. Select the domain
2. Create a Page Rule with the URL pattern of your API (i.e. [www.example.com/wp-login](http://www.example.com/wp-login))
3. Select the 'Security Level' setting
4. Mark the setting as 'High'
5. Select 'Save and Deploy'

# ACE - Exercise - Security - #IUAM

1. Adjust your test domain's security level to <I'm Under Attack Mode>.
2. At your browser, try access the domain and check the result and status codes.
3. At your terminal/POSTMAN, try access the domain and check the result and status codes.
4. After the exercise, turn off the <IUAM> mode so the next exercise won't be disturbed.



## 5. Set Rate Limiting Rules

Create Rate Limiting rules for hostname and/or specific endpoints such as login pages, API, etc.

Create a Rate Limiting Rule

Rule Settings

Rule Name

If Traffic Matching the URL

http & https

from the same IP address exceeds  requests per 

1 minute

Advanced Criteria

Then 

Block

 matching traffic from that visitor for 

1 hour

When "Block" is set, when the threshold is exceeded, the Client will receive a "429" error page until the Block time has expired.

Advanced Response

Bypass

Cancel

Save as Draft

Save and Deploy

Set tiered Rate Limiting rules to catch repeat offenders

RuleID	URL	Count	Timeframe	Matching Criteria	Action
1	/public/profile*	10	1 minute	Method: GET Status Code: 404	JavaScript Challenge
2	/public/profile*	25	1 minute	Method: GET Status Code: 200	Challenge
3	/public/profile*	50	10 minutes	Method: GET Status Code: 200, 404	Block for 4 hours



# ACE - Exercise - Security - #RateLimiting

## 1. Run this command

```
for i in {1..200}; do curl -svo /dev/null/
"https://www.yourdomain.cf/" 2>&1 | grep "<
HTTP"; done;
```

## 2. Create Rate Limiting Rule

## 3. Run the command again

## 4. Check the difference in response Find the log

**Rule Settings**

Rule Name

If Traffic Matching the URL

from the same IP address exceeds  requests per

[Advanced Criteria ▶](#)

Then  matching traffic from that visitor for

When "Block" is set, when the threshold is exceeded, the Client will receive a "429" error page until the Block time has expired.

[Advanced Response ▶](#)

# 6. Control Bots Visiting your Site

## Configure Super Bot Fight Mode

<b>Definitely automated</b> Definitely automated traffic typically consists of bad bots. Select an action for this traffic.	<input type="text" value="Challenge"/>
<b>Likely automated</b> Likely automated traffic can include bad bots, along with other traffic. Select an action for this traffic.	<input type="text" value="Allow"/>
<b>Verified bots</b> Verified bots are unique good bot identities validated by Cloudflare. Select an action for verified bots for this traffic.	<input type="text" value="Allow"/>
<b>Static resource protection</b> Enable if static resources on your application need bot protection. <b>Note:</b> Static resource protection can also result in legitimate traffic being blocked.	<input type="checkbox"/> On <input checked="" type="checkbox"/> Off
<b>JavaScript Detections</b> Use lightweight, invisible JavaScript detections to improve Bot Management. <a href="#">Learn more.</a>	<input checked="" type="checkbox"/> On <input type="checkbox"/> Off

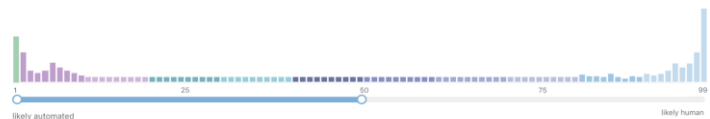
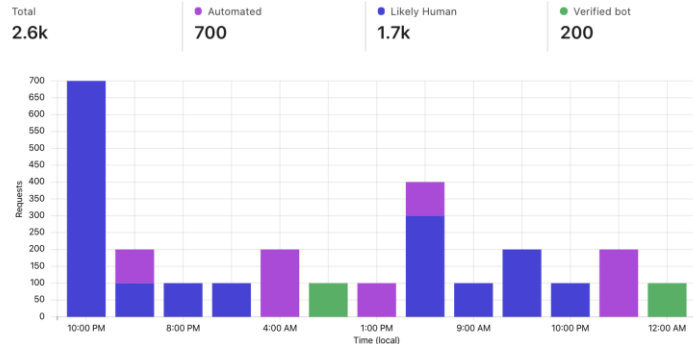
## Bot Analytics

[Configure Super Bot Fight Mode](#) [Print report](#)

[+ Add filter](#)

Last week

### Requests by traffic type



## Bot Management for Enterprise

Gain deeper control over your automated traffic with granular bot protection, anomaly detection, and path-level control.

[Add Bot Management](#)

## 7. Optimize application security with Firewall Rules

Trigger:

AS num	Cookie	Country	Continent	European Union
Hostname	IP Address	Referrer	Request Method	SSL/HTTPS
URI Full	URI	URI Path	URI Query String	HTTP Version
User Agent	X-Forwarded-For	Known Bots	Threat Score	And Custom

Action:

Block	JS Challenge	CAPTCHA	Allow	Bypass	Log
-------	--------------	---------	-------	--------	-----

# Example of Firewall Rules Usage

Zone  
Lockdown

## Create Firewall Rule

Changes to filter expressions may take up to 60 seconds to propagate globally. Changes to actions will take effect immediately. [×](#)

### Rule name

Give your rule a descriptive name

### When incoming requests match...

**Enter IP range and Hostname**

Field	Operator	Value	
IP Address <a href="#">×</a>	is not in	<input type="text" value="Enter multiple values"/>	And <a href="#">×</a>
And			
Hostname <a href="#">×</a>	equals	<input type="text" value="admin.example.com"/>	And Or <a href="#">×</a>

### Expression Preview

[Edit expression](#)

```
(not ip.src in {} and http.host eq "admin.example.com")
```

**Set Action to Block**

### Then...

Choose an action

Block <a href="#">▼</a>	<a href="#">Test rule</a>
-------------------------	---------------------------

# Example of Firewall Rules Usage

Require  
specific header  
to access

## Rule name

Give your rule a descriptive name

Require specific header

## When incoming requests match...

[Use expression builder](#)

```
not any(lower(http.request.headers.names[*])[*] contains "x-csrf-token") and (http.request.full_uri eq "https://www.theburritobot.com/admin")
```



## Then...

Choose an action

Block

Test rule

# Example of Firewall Rules Usage

Inspection of  
POST Body  
(add-on)

## Edit Firewall Rule

Changes to filter expressions may take up to 60 seconds to propagate globally. Changes to actions will take effect immediately. ✕

### Rule name

Give your rule a descriptive name

forbid special characters from username

### When incoming requests match...

[Use expression builder](#)

```
any(url_decode(http.request.body.form["username"])[*])[*] contains "'|'|/')
```



### Then...

Choose an action

Block

Test rule

# ACE - Exercise - Security - #FWRules

1. Try set a rule to only allow one client IP '1.2.3.4' when people access `<http://yourtestdomain.com>/admin.html`.
2. Try set a rule to give JS challenge to everyone when the request to `<http://yourtestdomain.com>/country.html` is NOT coming from your country.

<Hint> You may want to check “Firewall Rules”



# ACE - Security Summary

- Avoid HTTP, redirect/rewrite to HTTPS
- Hide origin IP/port
- Turn on WAF
- Use Security Level at Med or High, IUAM if needed.
- Set Rate Limiting Rules
- Get bots visibility and control them
- Deploy their security needs at Cloudflare Firewall Rules





// Performance Best Practices

# ACE - Exercise - Performance Comparison

1. Access <https://webpagetest.org>
2. Enter <https://www.<yourtestcomain.cf>> using origin **35.234.81.115** (from e.g. Singapore EC2)
3. Check the performance and load time
4. Run another test to <https://www.ace-training.cf>.  
This is a same page proxying to the same origin. Use the same location (e.g. Singapore EC2) when testing
5. Compare two results. Content is same. Is the result similar or different? Let's think about why.



# Performance Best Practices

1. Use Page Rules to maximise Caching for Static Contents
2. Turn on Performance and Optimization Features
3. Purge Cache when needed

Note: During Planning/Scoping stage, discuss with your customer on their cache and refresh requirements and workflow, this includes cache TTL, identifying static assets, cache purge workflow  
In depth cache configuration and behaviour will be covered in ASA course



# Content Cached by Default

Static file types is automatically cached.

Any other file types, like static HTML, need to be cached via a Page Rule.

bmp	ejs	jpeg	pdf	ps	ttf
class	eot	jpg	pict	svg	webp
css	eps	js	pls	svgz	woff
csv	gif	mid	png	swf	woff2
doc	ico	midi	ppt	tif	xls
docx	jar	otf	pptx	tiff	xlsx

# 1. Use Page Rules to maximise Caching for Static Contents

## How do I know what has been cached?

Cloudflare adds the response header **"CF-Cache-Status"** if attempting to cache the object. The value of this header indicates if successful:

- **MISS:** Not yet in the cache or the TTL expired (i.e. cache-control max age of 0)
- **HIT:** Asset delivered from cache
- **EXPIRED:** Resource was in cache but has since expired, served from origin server
- **REVALIDATED:** Delivered from cache. The TTL was expired, but a "If-Modified-Since" request to the origin indicated the asset has not changed so the version in cache is considered valid again.

Cache-Keys can also be customized to ensure content is being cached appropriately.

```
< Set-Cookie: __cfduid=d64e283e5e56319b47401dd2cf2327e5c1466531627; expires=Wed, 21-Jun-17 17:53:47 GMT; path=/; domain=.whiskytango.us; HttpOnly
< Last-Modified: Tue, 18 Aug 2015 09:05:34 GMT
< ETag: "55d2f55e-365ce"
< CF-Cache-Status: HIT - Cache Status
< Expires: Tue, 21 Jun 2016 21:53:47 GMT
< Cache-Control: public, max-age=14400
< Accept-Ranges: bytes
```

# Page Rules Example

1. Create a Page Rule with the URL pattern of your static HTML (i.e. [www.example.com/static-html/\\*](http://www.example.com/static-html/*))
2. Set Cache Level to **'Cache Everything'**
3. Set Edge and Browser TTLs
4. Press 'Save and Deploy'

test.cloudflare100.cf/\*.html

Then the settings are:

Browser Cache TTL	an hour	×
Cache Level	Cache Everything	×
Edge Cache TTL	4 hours	×

+ Add a Setting

Cancel Save

## 2. Turn on Performance and Optimization Features

- Enable Brotli & Image Compression using Polish
- Enable Minification for HTML & CSS
- Enable Cache Everything for all static content and HTML
- Enable Rocket Loader & Mirage
- Enable Argo Smart Routing
- Enable TLS 1.3 & 0-RTT Connection Resumption
- Enable Enhanced HTTP/2 Prioritization

### 3. Purge the cache when needed

- Purge the cache via API for event-driven content
- Purge the cache for individual files
  - Purging individual objects is a great way to maintain your cache-hit ratio while still ensuring certain objects are re-validated in our cache. [API Documentation](#)
- Purge the cache using the Cache-Tag
  - Cache-Tags allow you to define buckets of content that you wish to purge all together. This is an excellent way to combine objects that are commonly changed together. So an HTML blog post, for example, and all of its image content could be tagged together. Mobile-only content could also be bundled using cache-tags to purge everything when you push a new update to your mobile domain.
- Purge all files globally or via a matching Page Rule
- During Planning/Scoping stage, discuss with your customer on their content refresh requirements and workflow



# ACE - Exercise - Performance - #cache

Try creating a caching rule for all HTML pages under the domain.

- At the Edge:
  - If status code == 200(OK), **cache 1 minute**
  - If status code == 403(Forbidden) or 404(Not Found), **cache 10 minutes**
- At client browser: Recommend cache for 12 hours



# ACE - Performance Summary

- Cache as much as possible
- Maximise use of optimization features
  - a. Use Brotli compression
  - b. Use Argo Smart Routing (add-on)
  - c. Use newer TLS and better technologies
  - d. the other described optimization features
- Keep the customer's staging subdomain so you can always use it for origin compatibility test vs. CF optimization feature



# Thank you for your focus today!!

What we have learned so far:

Core Product Implementation Steps

Staging Test Tools & Hands-on

DNS/SSL Configurations

Useful Tooling for Cloudflare

Onboarding Best Practice

Security Best Practice

Performance Best Practice



# Next topic : Troubleshooting

Common Troubleshooting Issues & Reasons

Understand Errors

Cloudflare Logs

Cloudflare API

Working with Cloudflare Support





**THANK YOU**

**See you tomorrow at  
“Troubleshooting Cloudflare”!**