

NMAP: Scan avançado

Rede, Segurança

Vamos falar de como utilizar scan avançado no NMAP. A ideia aqui é apresentar outras formas de escaneamentos diferentes dos tradicionais apontados no post: NMAP: Mapeamento de portas TCP e UDP.

Dessa forma, vamos apresentar formas de escaneamento que podem identificar serviços utilizando uma abordagem heterodoxa. Se essa é sua primeira vez no nmap, sugerimos que volte no post anterior sobre nmap.

Escaneamento TCP NULL

O primeiro tipo de scan que vamos ver é o TCP NULL. Nessa modalidade de escaneamento, vamos usar um cabeçalho de flag do TCP nulo. Ou seja, o campo referente as flags vai estar como zero. Para isso, vamos usar a opção -sN. Como no comando abaixo:

sudo nmap -sN -p 80 192.168.0.1

```
$ sudo nmap -sN -p 80 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-27 20:28 EDT
Nmap scan report for 192.168.0.1
Host is up (0.00071s latency).
... .. 0... = Push: Not set
... .. 0... = Reset: Not set
... .. 0... = Select: Not set
... .. 0... = Window: Not set
PORT      STATE SERVICE
80/tcp    open|filtered http
MAC Address: 08:00:27:4B:32:42 (Oracle VirtualBox virtual NIC)
Window size value: 1024
Nmap done: 1 IP address (1 host up) scanned in 13.73 seconds
```

Escaneamento TCP NULL para a porta 80 do host 192.168.0.1

```
$ sudo nmap -sN -p 20-25 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-27 20:30 EDT
Nmap scan report for 192.168.0.1
Host is up (0.00074s latency).
Sequence number: 1 (relative sequence number)
PORT      STATE SERVICE
20/tcp    closed ftp-data
21/tcp    open|filtered ftp
22/tcp    open|filtered ssh
23/tcp    open|filtered telnet
24/tcp    closed priv-mail
25/tcp    closed smtp
MAC Address: 08:00:27:4B:32:42 (Oracle VirtualBox virtual NIC)
... .. 0... = ECN-Echo: Not set
Nmap done: 1 IP address (1 host up) scanned in 14.65 seconds
```

Escaneamento TCP NULL para as portas 20 a 25 do host 192.168.0.1

Podemos observar no Wireshark que o campo flag dos segmentos TCP enviados para o alvo estão nulos. Nesse tipo de escaneamento a falta de resposta do alvo

Indica que a porta está aberta ou filtrada por algum firewall.

3	13.183729828	PcsCompu_19:d5:47	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.111
4	13.184459792	PcsCompu_4b:32:42	PcsCompu_19:d5:47	ARP	60	192.168.0.1 is at 08:00:27:4b:32:42
5	13.184480881	192.168.0.111	192.168.0.1	TCP	54	59760 → 80 [<None>] Seq=1 Win=1024 Len=0
6	13.284264903	192.168.0.111	192.168.0.1	TCP	54	59761 → 80 [<None>] Seq=1 Win=1024 Len=0


```
▶ Frame 5: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface eth0, id 0
▶ Ethernet II, Src: PcsCompu_19:d5:47 (08:00:27:19:d5:47), Dst: PcsCompu_4b:32:42 (08:00:27:4b:32:42)
▶ Internet Protocol Version 4, Src: 192.168.0.111, Dst: 192.168.0.1
▼ Transmission Control Protocol, Src Port: 59760, Dst Port: 80, Seq: 1, Len: 0
  Source Port: 59760
  Destination Port: 80
  [Stream index: 0]
  [TCP Segment Len: 0]
  Sequence number: 1 (relative sequence number)
  Sequence number (raw): 3754446900
  [Next sequence number: 1 (relative sequence number)]
  Acknowledgment number: 0
  Acknowledgment number (raw): 0
  0101 .... = Header Length: 20 bytes (5)
  ▼ Flags: 0x000 (<None>)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    ....0... = Congestion Window Reduced (CWR): Not set
    ....0... = ECN-Echo: Not set
    ....0... = Urgent: Not set
    ....0... = Acknowledgment: Not set
    ....0... = Push: Not set
    ....0... = Reset: Not set
    ....0... = Syn: Not set
    ....0... = Fin: Not set
    [TCP Flags: .....]
  Window size value: 1024
  [Calculated window size: 1024]
  [Window size scaling factor: -1 (unknown)]
  Checksum: 0x1066 [unverified]
  [Checksum Status: Unverified]
  Urgent pointer: 0
```

Wireshark: TCP flags = NULL

Escaneamento TCP FIN

Esse escaneamento envia segmentos TCP com a flag FIN. Dessa forma, ao receber um flag FIN para uma porta que não está aberta o sistema tende a responder com um RST. Assim, se um alvo não responde a uma determinada porta, essa será considerada como aberta ou filtrada. A princípio o nmap não tem como saber se a porta está aberta ou se existe algum

mecanismo de segurança impedindo a resposta.

O comando abaixo utiliza a opção -sF para indicar que deverá ser enviado um segmento com a flag FIN.

sudo -sF -p 20-25 192.168.0.1

```
(root@kali) [~]
$ sudo nmap -sF -p 20-25 192.168.0.1 (ice number)
Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-27 20:32 EDT
Nmap scan report for 192.168.0.1 (relative sequence number)]
Host is up (0.00059s latency).
Acknowledge number (raw): 0
0101 = Header Length: 20 bytes (5)
PORT      STATE      SERVICE
20/tcp    closed    ftp-data
21/tcp    open|filtered ftp
22/tcp    open|filtered ssh
23/tcp    open|filtered telnet
24/tcp    closed    priv-mail
25/tcp    closed    smtp
MAC Address: 08:00:27:4B:32:42 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 14.71 seconds
[TCP Flags: ...E]
```

Escaneamento usando TCP FIN para as portas 20 a 25.

33	441.348045044	192.168.0.111	192.168.0.1	TCP	54	55738 → 22	[FIN]	Seq=1 Win=1024 Len=0
34	441.348137079	192.168.0.111	192.168.0.1	TCP	54	55738 → 21	[FIN]	Seq=1 Win=1024 Len=0
35	441.348185081	192.168.0.111	192.168.0.1	TCP	54	55738 → 23	[FIN]	Seq=1 Win=1024 Len=0
36	441.348249926	192.168.0.111	192.168.0.1	TCP	54	55738 → 25	[FIN]	Seq=1 Win=1024 Len=0
37	441.348303411	192.168.0.111	192.168.0.1	TCP	54	55738 → 24	[FIN]	Seq=1 Win=1024 Len=0
38	441.348348601	192.168.0.111	192.168.0.1	TCP	54	55738 → 20	[FIN]	Seq=1 Win=1024 Len=0
39	441.348844758	192.168.0.1	192.168.0.111	TCP	60	25 → 55738	[RST, ACK]	Seq=1 Ack=2 Win=0 Len=0
40	441.348844954	192.168.0.1	192.168.0.111	TCP	60	24 → 55738	[RST, ACK]	Seq=1 Ack=2 Win=0 Len=0
41	441.348845052	192.168.0.1	192.168.0.111	TCP	60	20 → 55738	[RST, ACK]	Seq=1 Ack=2 Win=0 Len=0
42	442.100180112	102.168.0.111	102.168.0.1	TCP	54	55720 → 22	[FIN]	Seq=1 Win=1024 Len=0

▶ Frame 35: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface eth0, id 0
 ▶ Ethernet II, Src: PcsCompu_19:d5:47 (08:00:27:19:d5:47), Dst: PcsCompu_4b:32:42 (08:00:27:4b:32:42)
 ▶ Internet Protocol Version 4, Src: 192.168.0.111, Dst: 192.168.0.1
 ▶ Transmission Control Protocol, Src Port: 55738, Dst Port: 23, Seq: 1, Len: 0

Source Port: 55738
 Destination Port: 23
 [Stream index: 17]
 [TCP Segment Len: 0]
 Sequence number: 1 (relative sequence number)
 Sequence number (raw): 1082253710
 [Next sequence number: 2 (relative sequence number)]
 Acknowledgment number: 0
 Acknowledgment number (raw): 0
 0101 = Header Length: 20 bytes (5)

▶ Flags: 0x001 (FIN)

000. = Reserved: Not set
 ...0 = Nonce: Not set
0... = Congestion Window Reduced (CWR): Not set
0... = ECN-Echo: Not set
0... = Urgent: Not set
0... = Acknowledgment: Not set
0... = Push: Not set
0... = Reset: Not set
0... = Syn: Not set
0... = Fin: Set

Wireshark: verificação da flag FIN do escaneamento TCP FIN

Escaneamento TCP ACK

Apresentamos agora um escaneamento diferente no seu funcionamento e no seu objetivo. Vamos falar do objetivo, nesse caso o scan deseja obter informação se o firewall é stateful. Dessa forma, não desejamos saber se as portas estão abertas ou não. O que desejamos é verificar se existe uma resposta RST a nossa consulta.

Caso o alvo responda com um RST poderemos presumir que não temos uma

regra de firewall stateful antes do serviço analisado. Isso é interessante para verificar se você configurou corretamente o firewall.

Exemplo de um scan utilizando ACK para um host no endereço 192.168.0.1 e portas 20 a 25:

sudo nmap -sA -p 20-25 192.168.0.1

```
(teste@kali)-[~]
└─$ sudo nmap -sA -p 20-25 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-27 20:38 EDT
Nmap scan report for 192.168.0.1
Host is up (0.00061s latency).
PORT      STATE      SERVICE
20/tcp    unfiltered ftp-data
21/tcp    unfiltered ftp
22/tcp    unfiltered ssh
23/tcp    unfiltered telnet
24/tcp    unfiltered priv-mail
25/tcp    unfiltered smtp
MAC Address: 08:00:27:4B:32:42 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 13.35 seconds
```

nmap scan tcp ack

Podemos observar que a resposta foi de que as portas não estavam filtradas. Dessa forma, podemos deduzir que não temos um firewall stateful entre origem e destino. Além disso, podemos verificar no Wireshark os

segmentos TCP com ACK e as respectivas respostas com RST. Isso comprova a tese de que o sistema está respondendo a ACK de forma indiscriminada.

49	785.524413964	192.168.0.111	192.168.0.1	TCP	54	56090 → 25	[ACK]	Seq=1 Ack=1 Win=1024 Len=0
50	785.524486407	192.168.0.111	192.168.0.1	TCP	54	56090 → 21	[ACK]	Seq=1 Ack=1 Win=1024 Len=0
51	785.524532545	192.168.0.111	192.168.0.1	TCP	54	56090 → 23	[ACK]	Seq=1 Ack=1 Win=1024 Len=0
52	785.525000052	192.168.0.111	192.168.0.1	TCP	54	56090 → 22	[ACK]	Seq=1 Ack=1 Win=1024 Len=0
53	785.525067336	192.168.0.111	192.168.0.1	TCP	54	56090 → 24	[ACK]	Seq=1 Ack=1 Win=1024 Len=0
54	785.525062595	192.168.0.1	192.168.0.111	TCP	60	25 → 56090	[RST]	Seq=1 Win=0 Len=0
55	785.525062798	192.168.0.1	192.168.0.111	TCP	60	21 → 56090	[RST]	Seq=1 Win=0 Len=0
56	785.525062884	192.168.0.1	192.168.0.111	TCP	60	23 → 56090	[RST]	Seq=1 Win=0 Len=0
57	785.525122955	192.168.0.111	192.168.0.1	TCP	54	56090 → 20	[ACK]	Seq=1 Ack=1 Win=1024 Len=0
58	785.525404531	192.168.0.1	192.168.0.111	TCP	60	22 → 56090	[RST]	Seq=1 Win=0 Len=0
59	785.525404647	192.168.0.1	192.168.0.111	TCP	60	24 → 56090	[RST]	Seq=1 Win=0 Len=0
60	785.525404732	192.168.0.1	192.168.0.111	TCP	60	20 → 56090	[RST]	Seq=1 Win=0 Len=0

Wireshark nmap scan tcp ack Escaneamento TCP Window

Agora vamos ver uma varredura semelhante à do TCP ACK que verifica se a resposta RST utiliza um tamanho de janela positivo. Esse comportamento é comum em alguns sistemas e pode ser usado para verificar se uma porta está aberta. Dessa forma, em alguns sistemas temos um RST com tamanho de janela positivo enquanto que para as portas fechadas temos um RST com tamanho 0. Portanto, trata-se de uma forma incomum de obter informações sobre serviços utilizando um Scan avançado no NMAP.

Exemplo de um scan utilizando TCP WINDOW para um host no endereço 192.168.0.1 e portas 20 a 25:

sudo nmap -sW -p 20-25 192.168.0.1

```
└─$ sudo nmap -sW -p 20-25 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-27 20:45 EDT
Nmap scan report for 192.168.0.1
Host is up (0.00042s latency).
PORT      STATE SERVICE
20/tcp    closed ftp-data
21/tcp    closed ftp
22/tcp    closed ssh
23/tcp    closed telnet
24/tcp    closed priv-mail
25/tcp    closed smtp
MAC Address: 08:00:27:4B:32:42 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 13.33 seconds
```

nmap RST Window

67	1184.8163208...	192.168.0.111	192.168.0.1	TCP	54	37265 → 22	[ACK]	Seq=1	Ack=1	Win=1024	Len=0
68	1184.8163883...	192.168.0.111	192.168.0.1	TCP	54	37265 → 23	[ACK]	Seq=1	Ack=1	Win=1024	Len=0
69	1184.8164288...	192.168.0.111	192.168.0.1	TCP	54	37265 → 21	[ACK]	Seq=1	Ack=1	Win=1024	Len=0
70	1184.8165258...	192.168.0.111	192.168.0.1	TCP	54	37265 → 25	[ACK]	Seq=1	Ack=1	Win=1024	Len=0
71	1184.8165540...	192.168.0.111	192.168.0.1	TCP	54	37265 → 24	[ACK]	Seq=1	Ack=1	Win=1024	Len=0
72	1184.8165900...	192.168.0.111	192.168.0.1	TCP	54	37265 → 20	[ACK]	Seq=1	Ack=1	Win=1024	Len=0
73	1184.8169008...	192.168.0.1	192.168.0.111	TCP	60	22 → 37265	[RST]	Seq=1	Win=0	Len=0	
74	1184.8169010...	192.168.0.1	192.168.0.111	TCP	60	23 → 37265	[RST]	Seq=1	Win=0	Len=0	
75	1184.8169011...	192.168.0.1	192.168.0.111	TCP	60	21 → 37265	[RST]	Seq=1	Win=0	Len=0	
76	1184.8169011...	192.168.0.1	192.168.0.111	TCP	60	25 → 37265	[RST]	Seq=1	Win=0	Len=0	
77	1184.8169012...	192.168.0.1	192.168.0.111	TCP	60	24 → 37265	[RST]	Seq=1	Win=0	Len=0	
78	1184.8169013...	192.168.0.1	192.168.0.111	TCP	60	20 → 37265	[RST]	Seq=1	Win=0	Len=0	

```

[Stream index: 34]
[TCP Segment Len: 0]
Sequence number: 1 (relative sequence number)
Sequence number (raw): 0
[Next sequence number: 1 (relative sequence number)]
Acknowledgment number: 1 (relative ack number)
Acknowledgment number (raw): 1006149232
0101 .... = Header Length: 20 bytes (5)
▼ Flags: 0x010 (ACK)
  000. .... = Reserved: Not set
  ...0 .... = Nonce: Not set
  ....0... = Congestion Window Reduced (CWR): Not set
  ....0... = ECN-Echo: Not set
  ....0... = Urgent: Not set
  ....1... = Acknowledgment: Set
  ....0... = Push: Not set
  ....0... = Reset: Not set
  ....0... = Syn: Not set
  ....0... = Fin: Not set
[TCP Flags: .....A....]
Window size value: 1024
[Calculated window size: 1024]
[Window size scaling factor: -1 (unknown)]
Checksum: 0xbe01 [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0

```

nmap RST Window. Podemos verificar que esse sistema não responde com tamanho de janela diferente.

NMAP Scan avançado com várias flags

Outra opção de escaneamento é enviar um segmento TCP com todos os flags marcados ou a combinação de mais de um flag. Dessa forma, verifica-se como o sistema destino responderá a esse tipo de segmento. Vale ressaltar que não importa a ordem em que as flags são informadas no comando. Isso porque, o comando apenas marca quais serão as flags que estarão marcadas no segmento TCP.

Exemplo de um scan utilizando várias flags TCP para um host no endereço 192.168.0.1 e portas 20 a 25:

sudo nmap -scanflags URGACKPSHRSTSYNFIN -p 20-25 192.168.0.1

```
$ sudo nmap --scanflags URGACKPSHRSTSYNFIN -p 20-25 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-27 20:55 EDT
Nmap scan report for 192.168.0.1
Host is up (0.00048s latency).
    ... = Push: Not set
    ... = Set: Not set
    ... = Fin: Not set
    ... = A....]
    ... size: 1024]
    ... stor: -1 (unknown)]
    ... unverified]
MAC Address: 08:00:27:4B:32:42 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 14.73 seconds
```

NMAP Escaneamento com várias flags

83	1780.1075343...	192.168.0.111	192.168.0.1	TCP	58 43828 → 25	[FIN, SYN, RST, PSH, ACK, URG]	Seq
84	1780.1076061...	192.168.0.111	192.168.0.1	TCP	58 43828 → 23	[FIN, SYN, RST, PSH, ACK, URG]	Seq
85	1780.1076557...	192.168.0.111	192.168.0.1	TCP	58 43828 → 22	[FIN, SYN, RST, PSH, ACK, URG]	Seq
86	1780.1076997...	192.168.0.111	192.168.0.1	TCP	58 43828 → 21	[FIN, SYN, RST, PSH, ACK, URG]	Seq
87	1780.1077525...	192.168.0.111	192.168.0.1	TCP	58 43828 → 20	[FIN, SYN, RST, PSH, ACK, URG]	Seq
88	1780.1078048...	192.168.0.111	192.168.0.1	TCP	58 43828 → 24	[FIN, SYN, RST, PSH, ACK, URG]	Seq
89	1781.2107863...	192.168.0.111	192.168.0.1	TCP	58 43829 → 24	[FIN, SYN, RST, PSH, ACK, URG]	Seq
90	1781.2108696...	192.168.0.111	192.168.0.1	TCP	58 43829 → 20	[FIN, SYN, RST, PSH, ACK, URG]	Seq
91	1781.2109171...	192.168.0.111	192.168.0.1	TCP	58 43829 → 21	[FIN, SYN, RST, PSH, ACK, URG]	Seq
92	1781.2109613...	192.168.0.111	192.168.0.1	TCP	58 43829 → 22	[FIN, SYN, RST, PSH, ACK, URG]	Seq
93	1781.2110129...	192.168.0.111	192.168.0.1	TCP	58 43829 → 23	[FIN, SYN, RST, PSH, ACK, URG]	Seq
94	1781.2110636...	192.168.0.111	192.168.0.1	TCP	58 43829 → 25	[FIN, SYN, RST, PSH, ACK, URG]	Seq

```
[Stream index: 40]
[TCP Segment Len: 0]
Sequence number: 0 (relative sequence number)
Sequence number (raw): 4036171604
[Next sequence number: 1 (relative sequence number)]
Acknowledgment number: 1 (relative ack number)
Acknowledgment number (raw): 2700866330
0110 .... = Header Length: 24 bytes (6)
▼ Flags: 0x03f (FIN, SYN, RST, PSH, ACK, URG)
  000. .... = Reserved: Not set
  ...0 .... = Nonce: Not set
  ....0... = Congestion Window Reduced (CWR): Not set
  ....0... = ECN-Echo: Not set
  ....1.1. = Urgent: Set
  ....1... = Acknowledgment: Set
  ....1... = Push: Set
  ► ....1. = Reset: Set
  ► ....1. = Syn: Set
  ► ....1. = Fin: Set
[TCP Flags: .....UAPRSF]
Window size value: 1024
[Calculated window size: 1024]
```

Wireshark Escaneamento com várias flags