NMAP: Scan avançado

Rede, Segurança

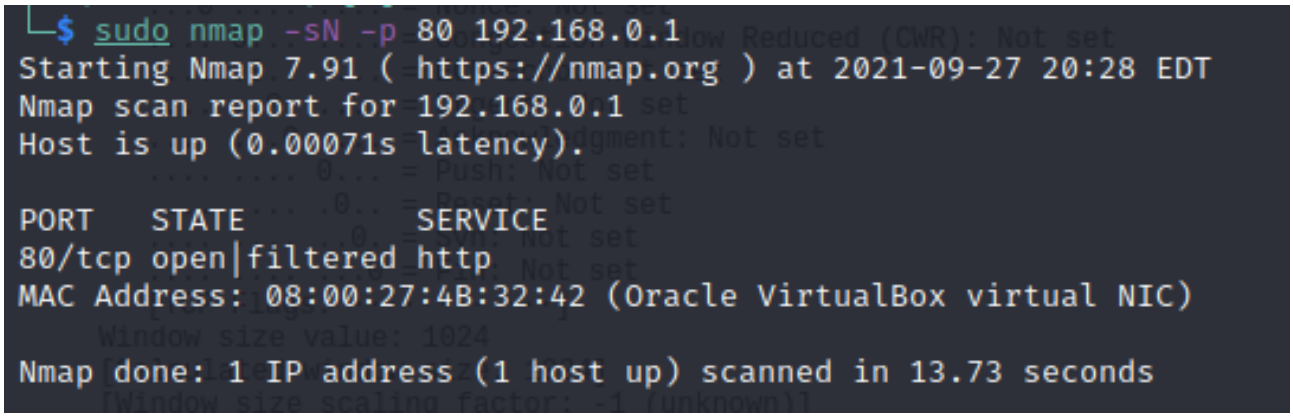
Vamos falar de como utilizar scan avançado no NMAP. A ideia aqui é apresentar outras formas de escaneamentos diferentes dos tradicionais apontados no post: NMAP: Mapeamento de portas TCP e UDP.

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.

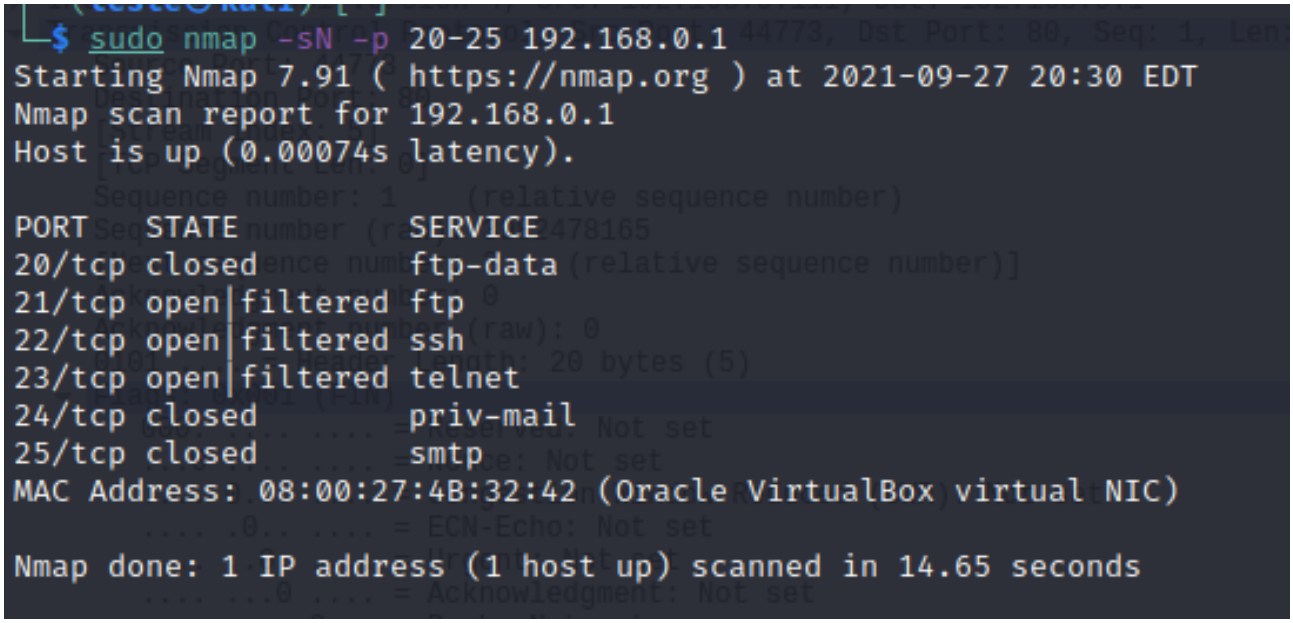
Escaneamento TCP NULL

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



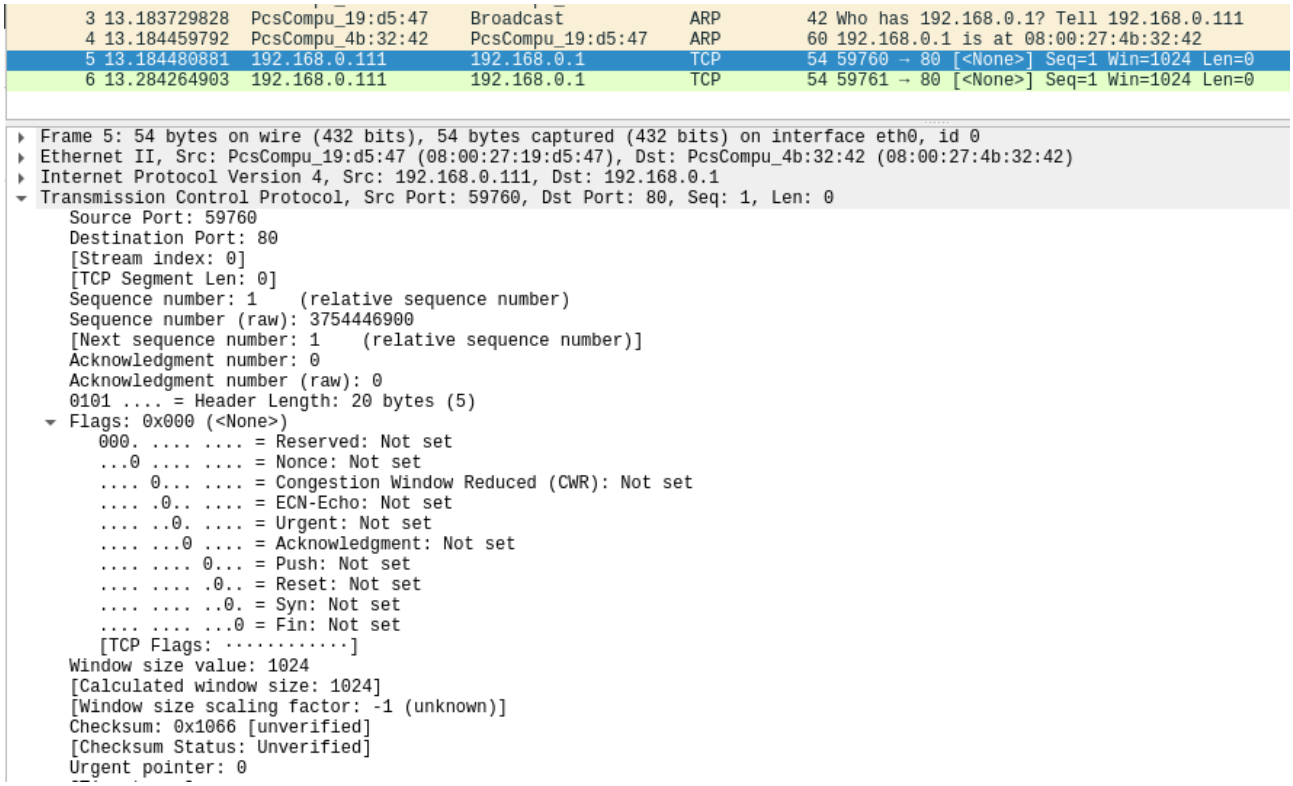
Escaneamento TCP NULL para a porta 80 do host 192.168.0.1



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.



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.

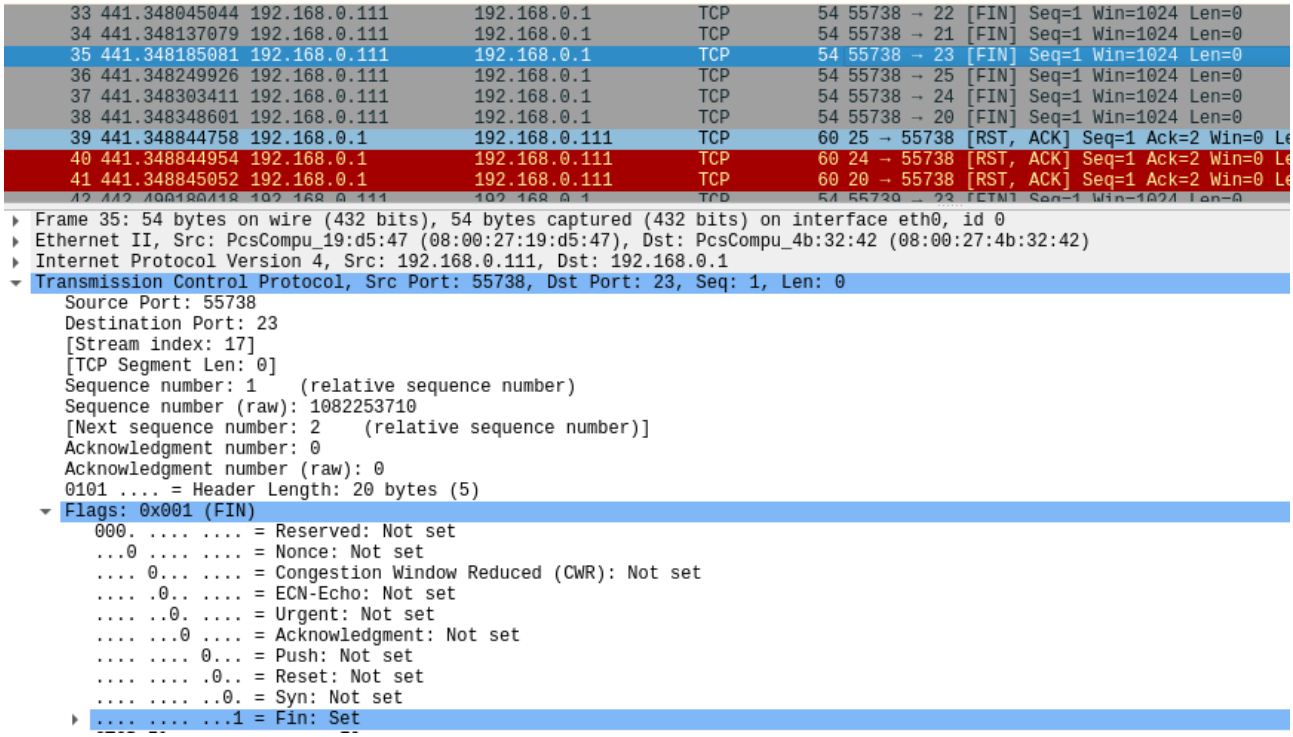
sudo -sF -p 20-25 192.168.0.1

```
└─$ sudo nmap -sF -p 20-25 192.168.0.1
Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-27 20:32 EDT
Nmap scan report for 192.168.0.1
Host is up (0.00059s latency).

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
```

Escaneamento usando TCP FIN para as portas 20 a 25.



Wireshark: verificação da flag FIN do escaneamento TCP FIN

Escaneamento TCP ACK

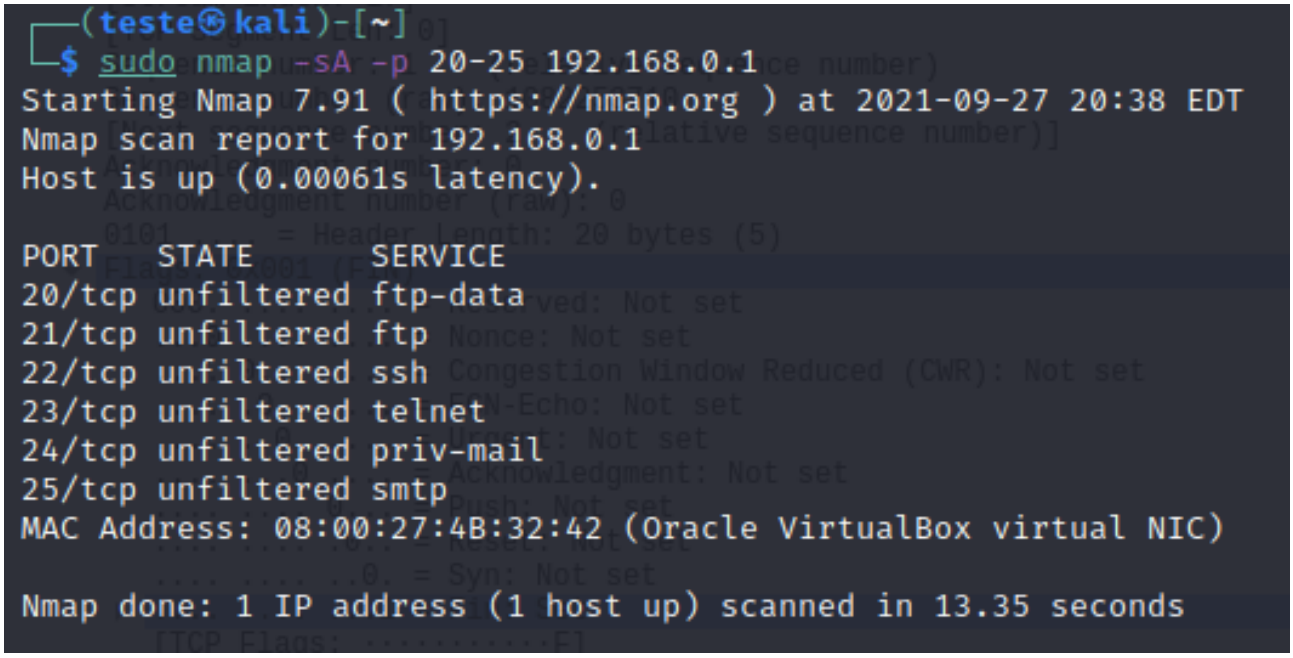
Apresentamos agora um escaneamento diferente no seu funcionamento e no seu objetivo. Vamos falar do objetivo, nesse caso o scan deseja obter informação se o firewall é stateful. Dessa forma, não desejamos saber se as portas estão abertas ou não. O que desejamos é verificar se existe uma resposta RST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

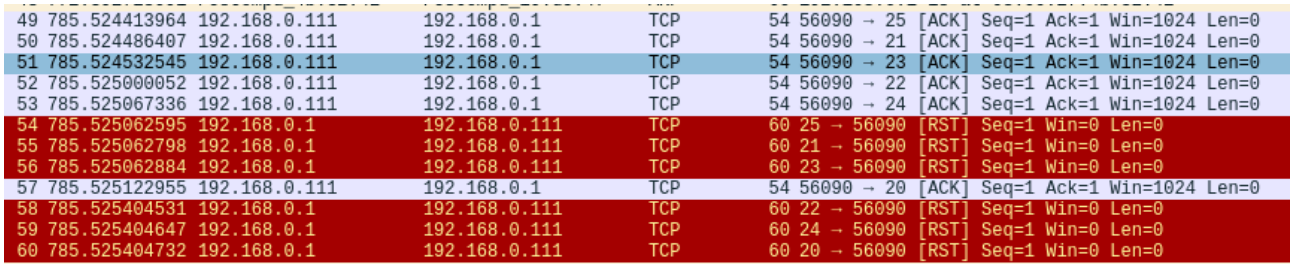
sudo nmap -sA -p 20-25 192.168.0.1



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.



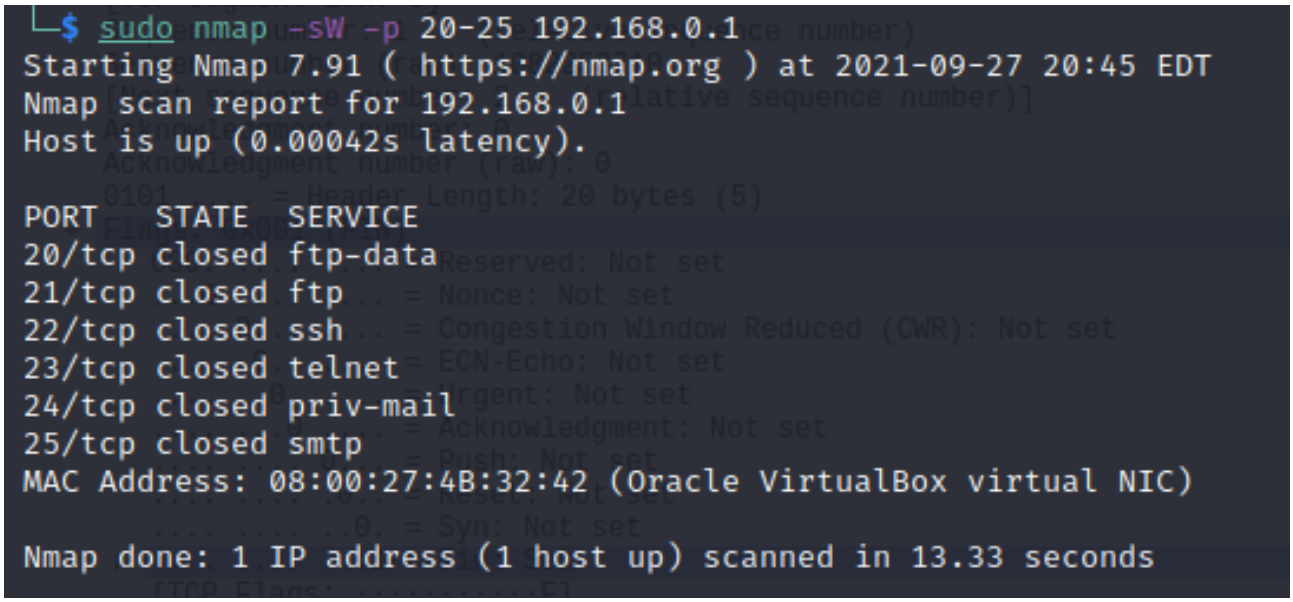
49	785.524413964	192.168.0.111	192.168.0.1	TCP	54 56090 → 25 [ACK] Seq=1 Ack=1 Win=1024 Len=0
50	785.524486407	192.168.0.111	192.168.0.1	TCP	54 56090 → 21 [ACK] Seq=1 Ack=1 Win=1024 Len=0
51	785.524532545	192.168.0.111	192.168.0.1	TCP	54 56090 → 23 [ACK] Seq=1 Ack=1 Win=1024 Len=0
52	785.525000052	192.168.0.111	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

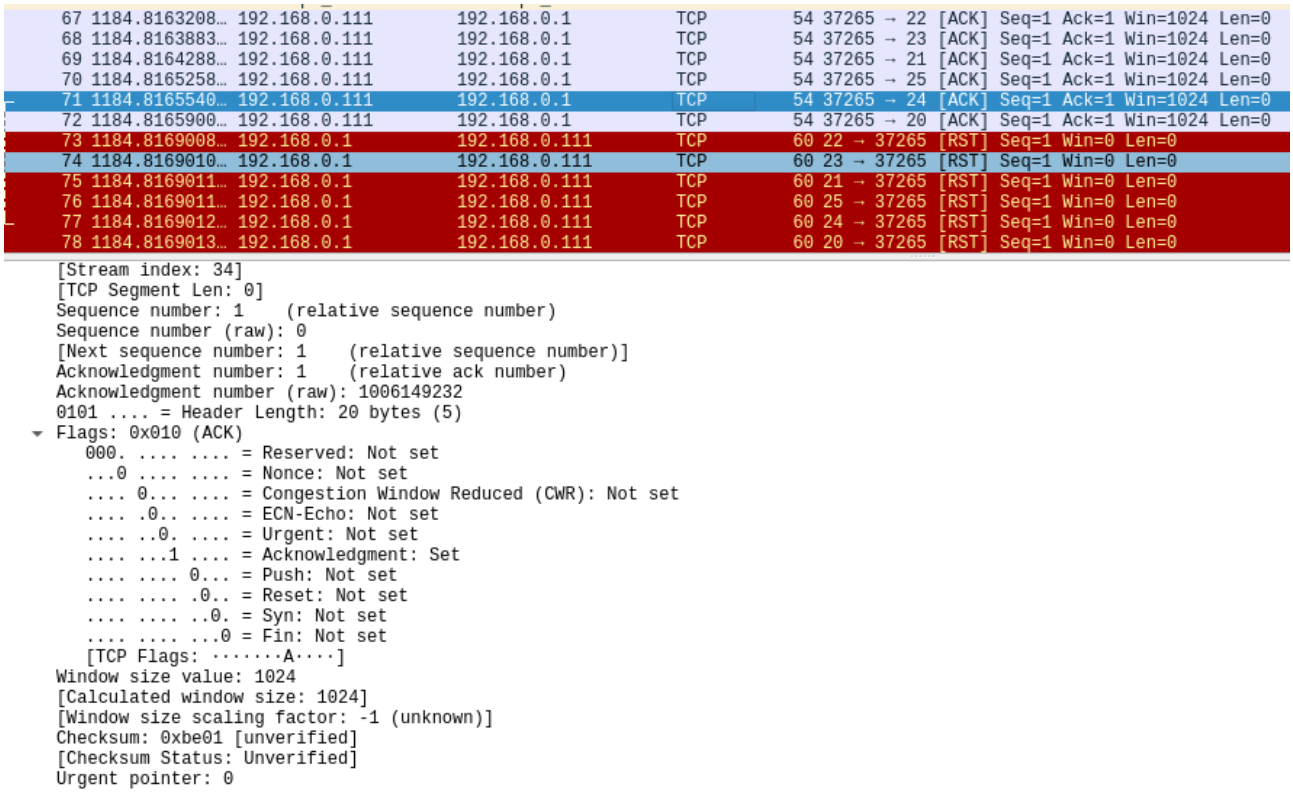
Agora vamos ver uma varredura semelhante à do TCP ACK que verifica se a resposta RST utiliza um tamanho de janela positivo. Esse comportamento é comum em alguns sistemas e pode ser usado para verificar se uma porta está aberta. Dessa forma, em alguns sistemas temos um RST com tamanho de janela positivo 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



nmap RST Window



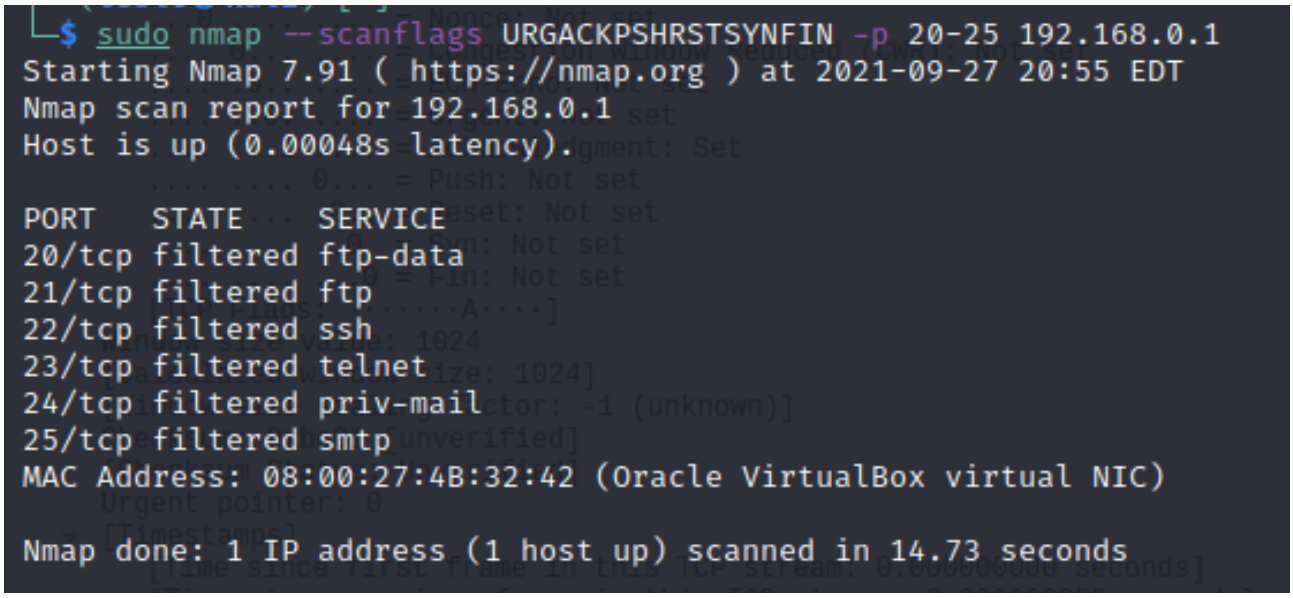
nmap RST Window. Podemos verificar que esse sistema não responde com tamanho de janela diferente.

NMAP Scan avançado com várias flags

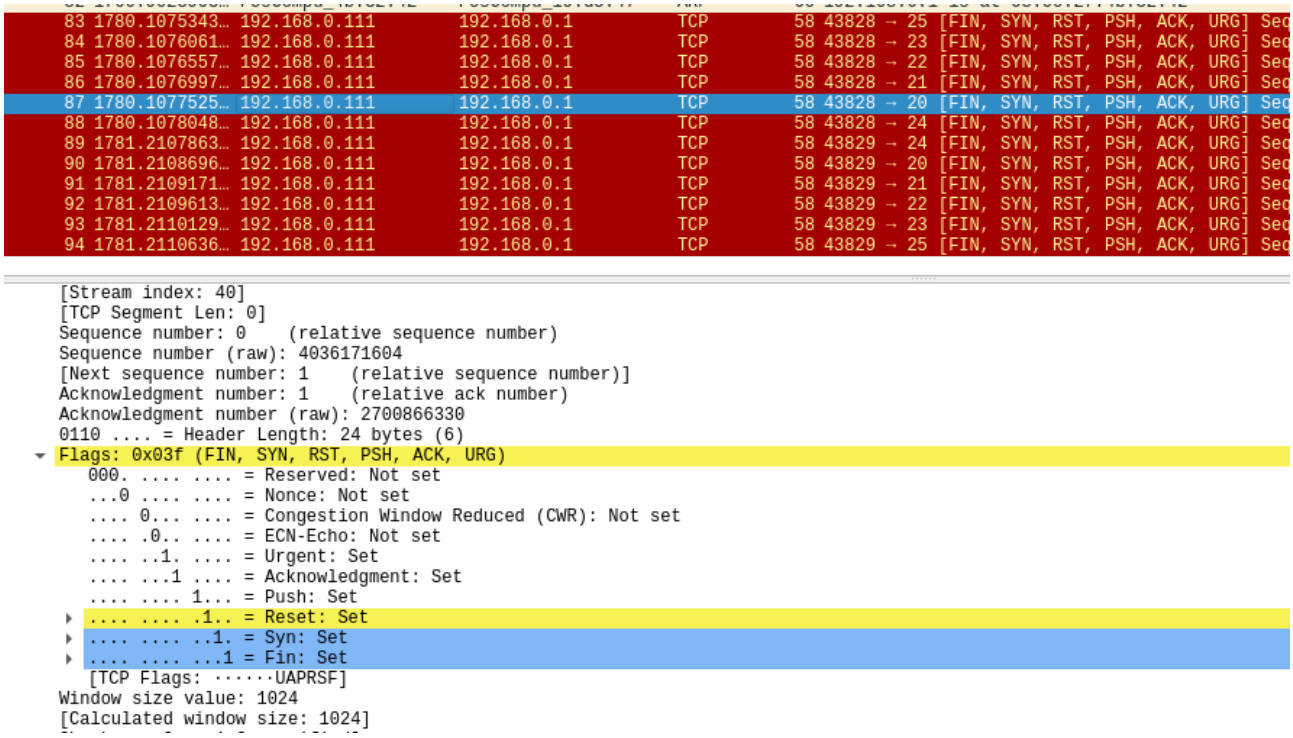
Outra opção de escaneamento é enviar um segmento TCP com todos os flags marcados ou a combinação de mais de um flag. Dessa forma, verifica-se como o sistema destino responderá a esse tipo de segmento. Vale ressaltar que não importa a ordem em que as flags são informadas no comando. Isso porque, o comando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



NMAP Escaneamento com várias flags



Wireshark Escaneamento com várias flags