Engenharia de Segurança

Portscan e Honeypots

Adaptado da Profa. Dra. Kalinka

Footprint

- Footprint é a técnica de coletar informações sobre sistemas de computadores e sobre as entidades os quais eles pertencem

Técnicas

- Consultas DNS
- Who is
- Consultas de rede
- Port-scaning
- Consultas SNMP
- Web Spidering
- Traceroutes
- Sniffing

Utilidade

- Falta de atualizações
- Pouca proteção do sistema
- Senha vulneráveis
- Conta de guest?

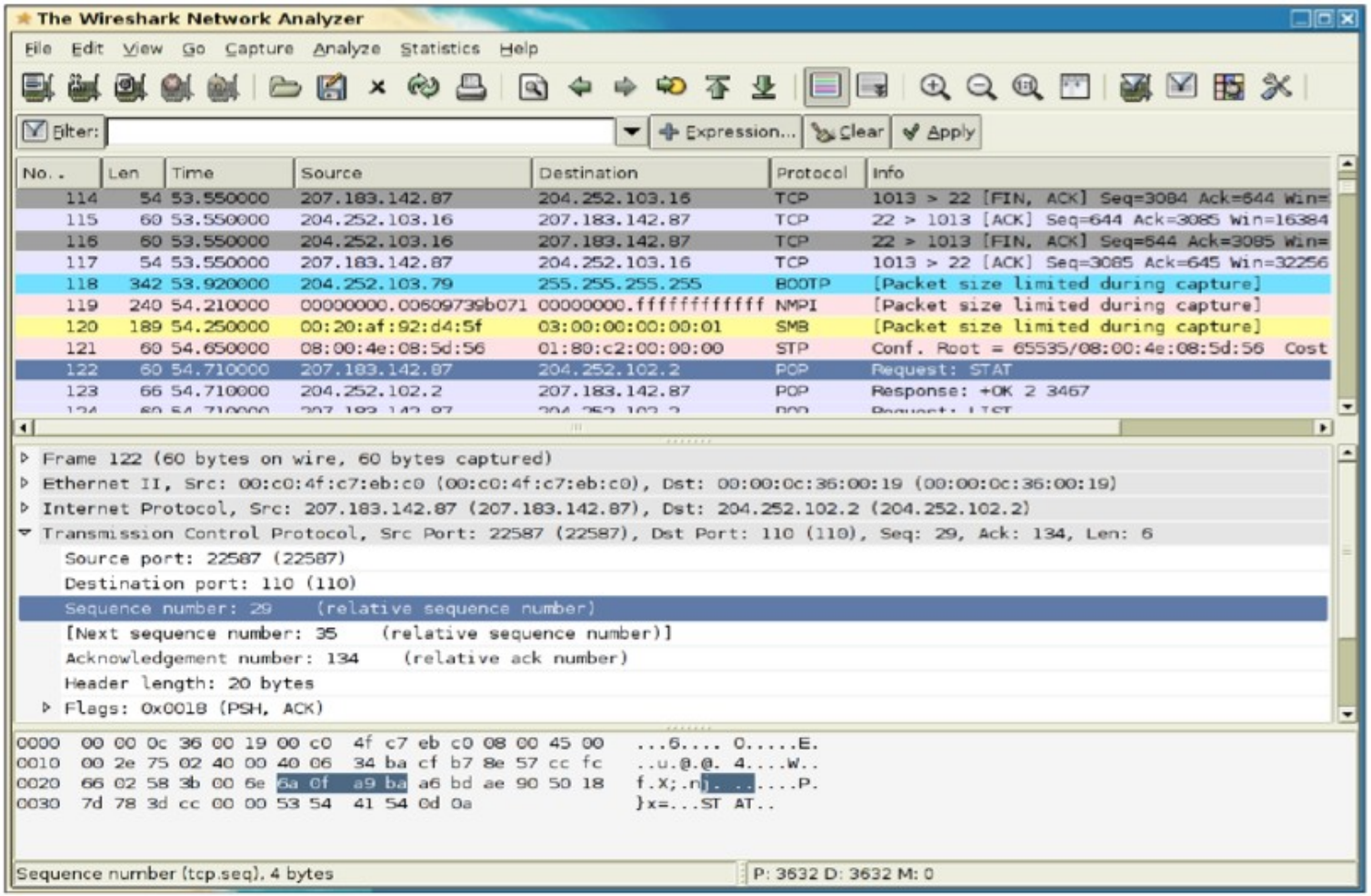
Levantamentos de Ativos e Passivos

- Footprinting passivo
  - Não deixa rastro, não são percebidos pelo alvos (sniffing)
  - Alguns sistemas anunciam a entrada em modo promíscuo (nos logs).
- Footprinting ativo
  - Utilizam conexões com o alvo, logo são rastreáveis.

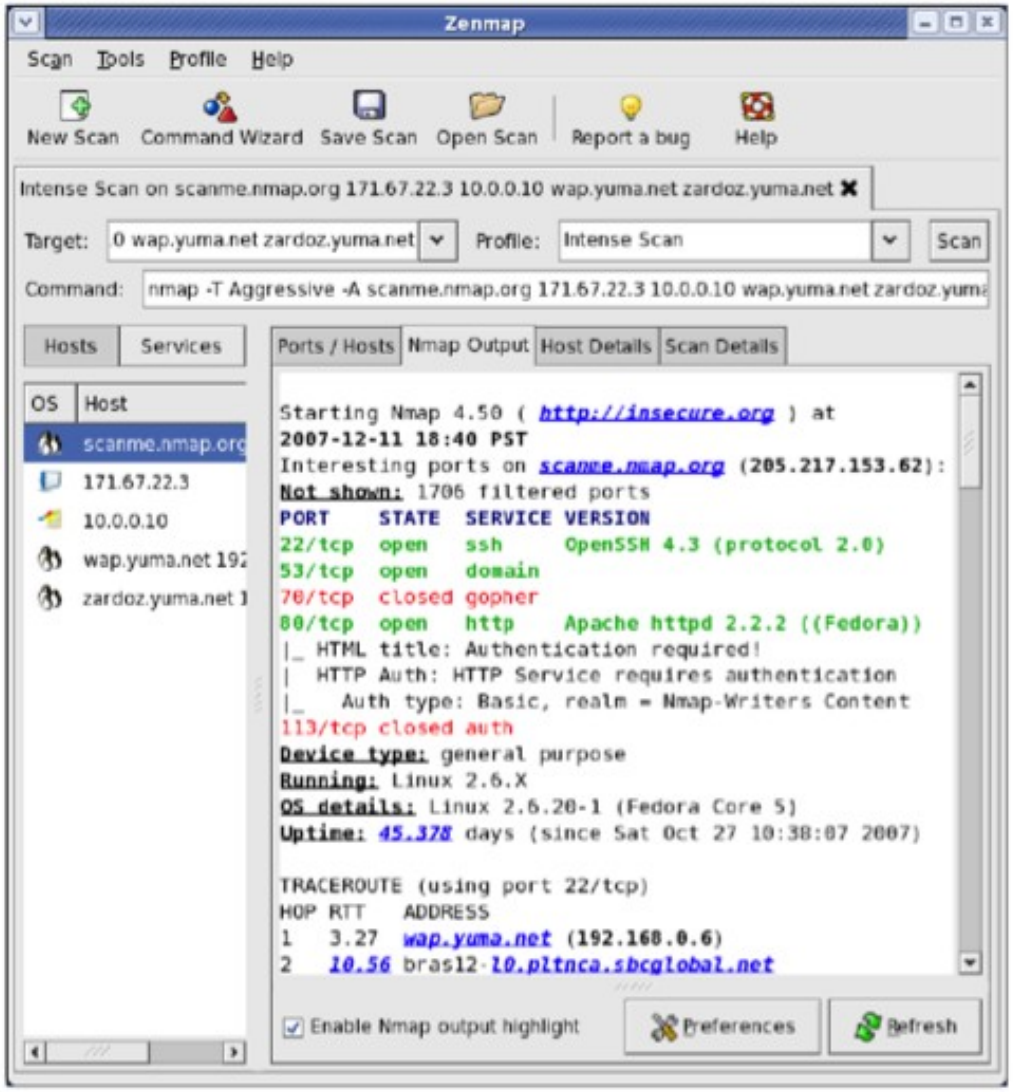
Ferramentas

- Wireshark
- Zenmap
- Scanners

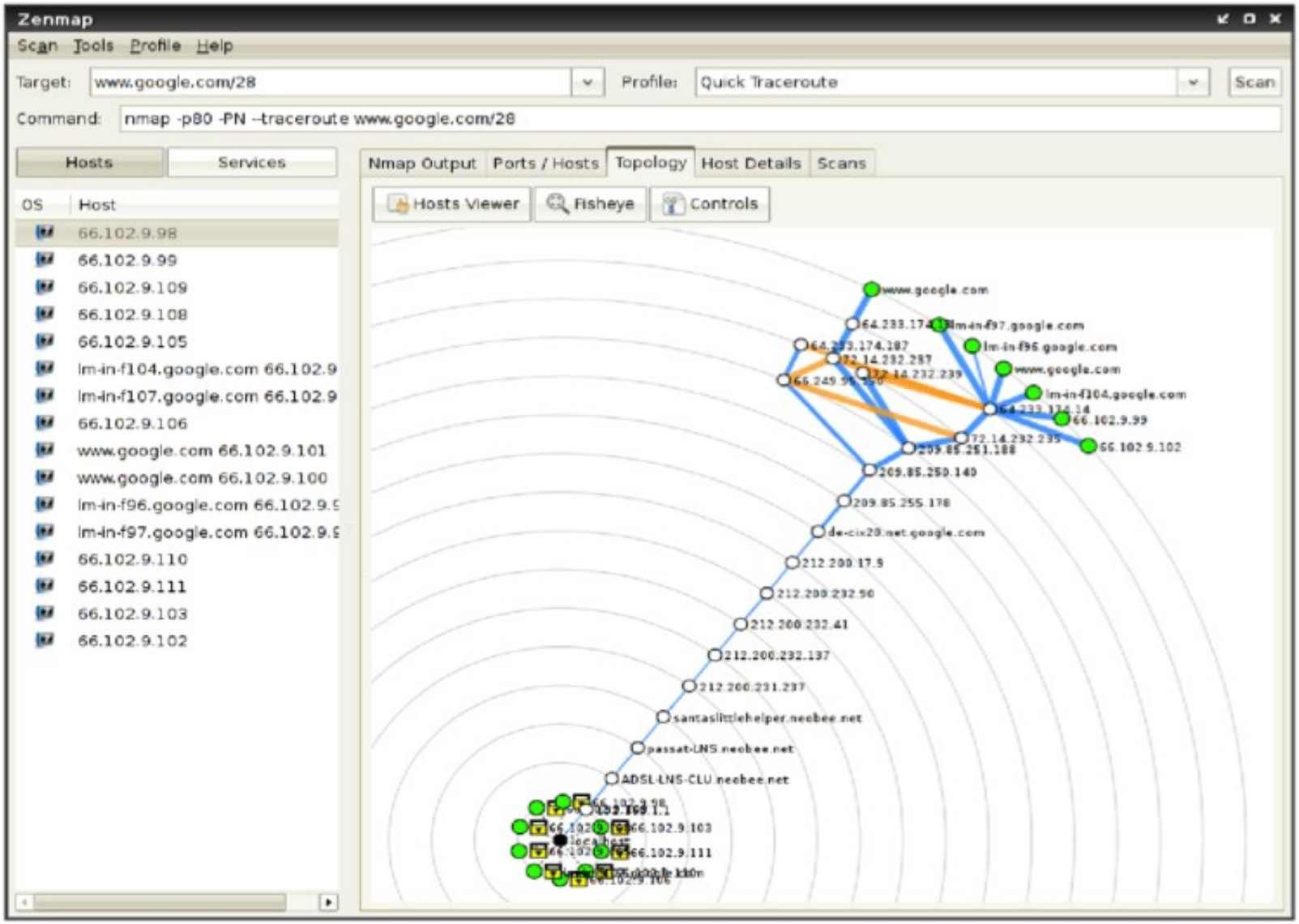
Wireshark


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply
No. . Len Time Source Destination Protocol Info
114 54 53.550000 207.183.142.87 204.252.103.16 TCP 1013 > 22 [FIN, ACK] Seq=3084 Ack=644 Win=
115 60 53.550000 204.252.103.16 207.183.142.87 TCP 22 > 1013 [ACK] Seq=644 Ack=3085 Win=16384
116 60 53.550000 204.252.103.16 207.183.142.87 TCP 22 > 1013 [FIN, ACK] Seq=644 Ack=3085 Win=
117 54 53.550000 207.183.142.87 204.252.103.16 TCP 1013 > 22 [ACK] Seq=3085 Ack=645 Win=32256
118 342 53.920000 204.252.103.79 255.255.255.255 BOOTP [Packet size limited during capture]
119 240 54.210000 00000000.0060973 9b071 00000000.ffffffffffff NMPI [Packet size limited during capture]
120 189 54.250000 00:20:af:92:d4:5f 03:00:00:00:00:01 SMB [Packet size limited during capture]
121 60 54.650000 08:00:4e:08:5d:56 01:80:c2:00:00:00 STP Conf. Root = 65535/08:00:4e:08:5d:56 Cost
122 60 54.710000 207.183.142.87 204.252.102.2 POP Request: STAT
123 66 54.710000 204.252.102.2 207.183.142.87 POP Response: +OK 2 3467
Frame 122 (60 bytes on wire, 60 bytes captured)
Ethernet II, Src: 00:c0:4f:c7:eb:c0 (00:c0:4f:c7:eb:c0), Dst: 00:00:0c:36:00:19 (00:00:0c:36:00:19)
Internet Protocol, Src: 207.183.142.87 (207.183.142.87), Dst: 204.252.102.2 (204.252.102.2)
Transmission Control Protocol, Src Port: 22587 (22587), Dst Port: 110 (110), Seq: 29, Ack: 134, Len: 6
Source port: 22587 (22587)
Destination port: 110 (110)
Sequence number: 29 (relative sequence number)
[Next sequence number: 35 (relative sequence number)]
Acknowledgement number: 134 (relative ack number)
Header length: 20 bytes
Flags: 0x0018 (PSH, ACK)
0000 00 00 0c 36 00 19 00 c0 4f c7 eb c0 08 00 45 00 ...6.... O.....E.
0010 00 2e 75 02 40 00 40 06 34 ba cf b7 8e 57 cc fc ..u.@.@. 4....W..
0020 66 02 58 3b 00 6e 6a 0f a9 ba a6 bd ae 90 50 18 f.X;.nj. ......P.
0030 7d 78 3d cc 00 00 53 54 41 54 0d 0a }x=...ST AT..
Sequence number (tcp.seq), 4 bytes
P: 3632 D: 3632 M: 0

Zenmap


Zenmap
Scan Tools Profile Help
New Scan
Command Wizard
Save Scan
Open Scan
Report a bug
Help
Intense Scan on scanme.nmap.org 171.67.22.3 10.0.0.10 wap.yuma.net zardoz.yuma.net
Target: 0 wap.yuma.net zardoz.yuma.net
Profile: Intense Scan
Scan
Command: nmap -T Aggressive -A scanme.nmap.org 171.67.22.3 10.0.0.10 wap.yuma.net zardoz.yuma
Hosts
Services
OS
Host
scanme.nmap.org
171.67.22.3
10.0.0.10
wap.yuma.net 192
zardoz.yuma.net 1
Ports / Hosts
Nmap Output
Host Details
Scan Details
Starting Nmap 4.50 (http://insecure.org) at
2007-12-11 18:40 PST
Interesting ports on scanme.nmap.org (205.217.153.62):
Not shown: 1706 filtered ports
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 4.3 (protocol 2.0)
53/tcp open domain
70/tcp closed gopher
80/tcp open http Apache httpd 2.2.2 ((Fedora))
|_ HTML title: Authentication required!
| HTTP Auth: HTTP Service requires authentication
|_ Auth type: Basic, realm = Nmap-Writers Content
113/tcp closed auth
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.20-1 (Fedora Core 5)
Uptime: 45.378 days (since Sat Oct 27 10:38:07 2007)
TRACEROUTE (using port 22/tcp)
HOP RTT ADDRESS
1 3.27 wap.yuma.net (192.168.0.6)
2 10.56 bras12-l0.pltnca.sbcglobal.net
Enable Nmap output highlight
Preferences
Refresh

Zenmap como um Scaner



Portscan

- O ***Portscan***, que foi criado para que os administradores pudessem visualizar os serviços em sua rede, é como os atacantes geralmente começam a buscar informações em seu servidor.
- Verificam quais os serviços e portas que se encontram abertas e em uso no servidor. Capaz de localizar vulnerabilidades entre máquinas que se encontram na rede.

Portscan

- Bem, analogicamente, podemos comparar o Portscan com um ladrão, que vigia um bairro inteiro a procura de janelas e portas abertas, por onde possa entrar.
- Primeiro precisamos entender que, como manda a RFC do TCP, quando começamos uma negociação para conexão TCP com outro computador, mandamos um pacote com a *flag* SYN ativada, e ele deve então responder um pacote com as *flags* SYN+ACK ativadas, ou seja, quando há resposta a porta se encontra aberta, dependendo da porta sabemos qual o serviço que se encontra ativo nela.

- Ha vários tipos de *portscanner*, uns mais efetivos e mais "seguros" para os atacantes, ou seja, que deixam menos rastros:
  - **Método connect() ou "Dumb Scan"** – E um método antigo e simples, utiliza o processo que foi descrito acima. Felizmente, quando se utiliza esse método, o atacante, quando recebe o SYN+ACK, devolve um pacote ACK onde inicia a conexão, porem nesse mesmo momento é possível se identificar a origem. O ataque é facilmente detectado.
  - **Método Half-open** – Vendo o problema do método connect(), surgiu à ideia de se fechar a conexão mandando um RST ao receber o pacote SYN+ACK, uma vez que já era possível somente com ele saber se a porta estava aberta ou não. E como somente no ultimo ACK estava a origem do atacante não seria possível identifica-lo. Esse ataque foi considerado muito eficiente e por um momento indetectável. Porém os sistemas foram se aprimorando e começaram a identificar a origem no SYN inicial da conexão e não mais no ACK final. Tornando esse ataque facilmente identificável.

Portscan

- Começaram então a aparecer métodos obscuros de ataque, que são considerados mais efetivos:
  - **Fin Scan** – Baseados mais uma vez na RFC do TCP, que diz que toda porta fechada deve responder com a flag RST ao receber um pacote com a flag FIN ativada e as portas abertas simplesmente ignoram o pacote com a flag. Sendo assim, ao invés de mandar um SYN, começaram a mandar um FIN que não continha a origem do atacante, aquelas portas que respondessem com um RST estavam fechadas, as que não respondessem nada estavam provavelmente abertas. A desvantagem dessa técnica e que não possibilita a identificação de algum tipo de filtragem por um *firewall*. Além disso, a Microsoft não segue as recomendações da RFC e responde com RST em todas as portas.

Portscan

- **Null Scan** – Bem similar ao Fin *scan*, Assim como quando enviamos a flag FIN, ao enviar a flag NULL (onde desligamos todas as *flags* do pacote) as portas fechadas devem responder com a *flag* RST e as portas abertas simplesmente os ignoram. Porém os mesmos problemas do Fin scan também são aplicados aqui. Outro problema e que se o scan não identificar que a máquina esta "unreachable" ou tem algum tipo de filtragem ele pode entender que a porta que não responda está aberta, retornando uma informação que não é correta.
- **Decoy Scan** – Este é o método mais completo, pois envolve uma junção com uma da técnicas anteriores (menos a *connect*), tornando o ataque mais poderoso e trazendo informações possivelmente mais relevantes. Ele disfarça a origem real do ataque, enviado diversas origens como se fossem vários computadores fazendo o *port scan* ao mesmo tempo, sendo que somente alguns dos pacotes foram enviados pelo ip real do atacante confundindo o computador.

Honeypots e Honeynets

- A ideia é entender o conceito de honeypots e honeynets e saber como montá-las e quais informações podem fornecer.

Honeypots

- Em computação, honeypot é uma armadilha montada para refletir, analizar e contra atacar acessos não-autorizados a sistemas de informação. Em geral, consiste de um computador, dados falsos ou de uma pequena parte da rede que parece fazer parte da rede principal do alvo, mas na verdade se trata de uma rede isolada e constantemente monitorada. Um honeypot parece ter informações valiosas ao atacante.

Honeypots – como montar?

- Usando computadores reais;
- Usando máquinas virtuais;
- Usando softwares falsos
  - Fakehttp
  - Fakeproxy
- Usando o deamon honeyd
- Usando o próprio Netkit

Elementos importantes

- Ele deve passar informações ao atacante
- Ele deve ser totalmente isolado
- Ele deve ser continuamente monitorado

Honeypots vs Honenets

- Um honeypot é um conjunto ou serviço falso que passa informações ao receptor, agindo como um único computador.
- Uma honeynet é uma rede de captura de informações sobre o ataque
  - O conceito pode ser expandido para *honeyclouds*

Honeycloud (era *free*, não mais)

Como funciona a prática

- Utilização da ferramenta honeyd
  - Opera através de uma configuração e um conjunto de scripts
  - O deamon captura os pacotes em modo promíscuo, interpreta e responde com o conteúdo adequado, de acordo com os scripts inteligentes.