Taller de Capa de Transporte

Teoría de las Comunicaciones

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

30.10.2012

Agenda

1. Discovery a nivel transporte
   - Técnicas de port scanning
   - Detección de servicios
   - Detección de sistemas operativos
   - Herramientas
2. Consignas

Agenda

1. Discovery a nivel transporte
   - Técnicas de port scanning
   - Detección de servicios
   - Detección de sistemas operativos
   - Herramientas
2. Consignas

Recopilación de info - information gathering

- Desde la óptica de usuario de red arbitrario, hasta ahora sabemos cómo reconocer hosts activos y al alcance en una red desconocida.
- El siguiente paso:
  - Determinar puertos disponibles,
  - Determinar servicios disponibles (y sus respectivas implementaciones), e
  - Identificar el sistema operativo subyacente.
- Permite, por ejemplo, armar un inventario de la red.
- Además, tiene implicancias importantes en seguridad informática.

A por los puertos!

- La identificación de los puertos disponibles se conoce como port scanning.
- Procedimiento que barre una secuencia de puertos en un host dado enviando paquetes y analizando las posibles respuestas.
- Se puede materializar de formas variadas, según la heurística que se desee utilizar.
- Hoy veremos:
  - SYN scanning,
  - Connect scanning,
  - Xmas, null y FIN scanning, y
  - UDP scanning.
- Se asume (en los primeros) compatibilidad con la especificación de TCP (RFC 793).

Clasificación de puertos

- Usulamente, un scan clasificará a un puerto de la siguiente manera:
- **open** (abierto) si pudo inferir de su heurística que hay un servicio escuchando en dicho puerto.
- **closed** (cerrado) si determinó que no hay un proceso escuchando allí.
- **filtered** (filtrado) si hay chances de que alguna entidad intermedia (e.g., firewall) descarte el tráfico correspondiente.
- Puede ocurrir que la heurística del scanner no pueda discernir completamente entre estas categorías.

SYN scan (o half-open scan)

- El scan más popular!
- Envía un paquete TCP con flag SYN y espera la respuesta:
  - SYN/ACK: indica que el puerto está abierto.
  - RST: indica que el puerto está cerrado.
  - Si no hay respuesta, se asume filtrado. Ídem si la respuesta es ICMP de tipo destination unreachable.
- No se completa la conexión.

SYN scan: ventajas y desventajas

- Suele ser rápido.
- Poco invasivo y discreto, al no completar las conexiones.
- Por otro lado, requiere enviar y recibir paquetes ad-hoc, lo cual requiere permisos elevados en el sistema.

Connect scan (o TCP scan)

- Usa la API del sistema operativo para establecer una conexión: primitiva `connect` de sockets.
- Así se inicia una conexión usualmente en clientes web, clientes de mail, clientes FTP, etc.
- Observar que se instancia por completo el algoritmo de three-way handshake!

Connect scan: ventajas y desventajas

- No requiere permisos elevados para correr.
- Pero tiene varias contras...
  - Más lento y con más volumen de tráfico, al completar conexiones.
  - Menos discreto: los hosts posiblemente registren en logs estas conexiones.

Scans bizarros (Xmas scan, null scan y FIN scan)

- Se apoyan en sutilezas del RFC.
- Si el estado de un puerto es CLOSED, la especificación dice (pág. 65):

```
 An incoming segment containing a RST is discarded.
An incoming segment not containing a RST causes a RST to be
sent in response.
```

- Si es LISTEN, afirma que todo paquete sin flags ACK, SYN ni RST (situación muy poco frecuente) **debe ser descartado** (pág. 66).

Cómo funcionan

- Si el host destino implementa coherentemente el RFC, enviar un paquete TCP que **no** incluya los flags SYN, ACK o RST resultará en:
  - Puerto cerrado $\Rightarrow$ respuesta con flag RST.
  - Puerto abierto $\Rightarrow$ respuesta inexistente.
- Los otros flags (PSH, URG y FIN) pueden tomar valores arbitrarios, lo cual determina las variantes de los scans:
  - Xmas scan: todos prendidos (como un arbolito de navidad!).
  - Null scan: ninguno prendido.
  - FIN scan: sólo FIN prendido.
- Observar que una falta de respuesta también puede indicar que el puerto está filtrado!

Xmas, null y FIN scans: ventajas y desventajas

- Pueden escabullirse mejor que un SYN scan a través de firewalls.
- Sin embargo, pueden resultar más sospechosos.
- Además, no es confiable con SOs que no implementen al pie de la letra estos detalles. Ejemplo: Windows.
- Y, como ya vimos, no hay una distinción clara entre puertos abiertos y filtrados.

UDP scan

- Es más complicado hacer scans por UDP: protocolo muy simple y sin conexiones!
- Pero es necesario dado que hay servicios que lo usan (como DNS o DHCP).
- Funciona enviando un paquete UDP al puerto destino:
  - Si la respuesta es ICMP de tipo port unreachable $\Rightarrow$ puerto cerrado.
  - Si no hay respuesta $\Rightarrow$ puede estar abierto o filtrado.
- En algunos casos pueden regresar paquetes UDP como respuesta, lo cual indica que el puerto sí está abierto.

UDP scan: dificultades

- No es fácil hacerlo rápido: requiere retransmisiones por si los paquetes se extraviaron.
- Además, los hosts a veces restringen la cantidad de mensajes ICMP de tipo port unreachable.

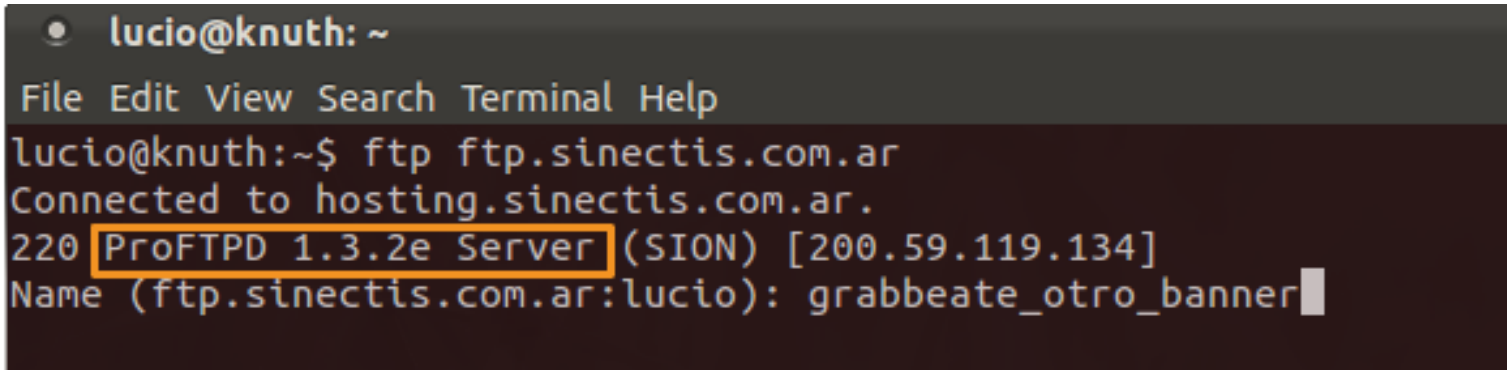
Banner grabbing

- Para identificar fehacientemente las aplicaciones detrás de los puertos, una técnica posible es banner grabbing.
- Consiste en conectarse al servicio y observar en la información intercambiada si hay trazas del nombre y/o versión del programa en cuestión.
- Para ello, pueden utilizarse distintas herramientas: `telnet` , `netcat` , web browsers, etc.

Banner grabbing: un ejemplo

- Observar que en este caso obtenemos también gratis el sistema operativo!

Banner grabbing: otro ejemplo



OS fingerprinting

- Ésta es la técnica para identificar la “huella característica” de un sistema operativo.
- Una forma posible de hacerlo es analizando cómo el host reacciona ante distintos paquetes.
- Por lo general se usan paquetes TCP o ICMP para los cuales las respuestas particulares de distintos sistemas operativos pueden variar, aprovechando ambigüedades en las especificaciones.
- Como vimos recién, un banner también puede darnos información acerca del sistema operativo subyacente.

Variantes

- Esencialmente existen dos variantes: detección **activa** y detección **pasiva**.
- En la primera, se envía un conjunto de paquetes al host destino y luego se procesan las respuestas obtenidas.
- La otra alternativa, más discreta, escucha la red y captura el tráfico, realizando el procesamiento posterior del mismo de manera similar a su contraparte activa.
- `nmap` es el ejemplo paradigmático de detección de SO activa.
- La herramienta `p0f` implementa detección pasiva.
- Y otra popular, `SinFP`, provee sendas estrategias.

Cómo funciona (a alto nivel)

- Usualmente, se dispone de una base de datos almacenando los posibles resultados de distintos tests para una serie de SOs.
- Luego de enviar paquetes con ciertas características, los tests consisten en estudiar los campos de las respuestas y utilizar la base para calcular el SO más probable.
- Algunos posibles tests:
  - Análisis de números de secuencia de TCP.
  - Análisis del campo ID en el header IP.
  - Análisis de opciones TCP (orden y disponibilidad variables).
- Más info acá: `http://nmap.org/book/osdetect-methods.html`

telnet

- En realidad es un protocolo de nivel de aplicación (sobre TCP)...
- Está especificado en el RFC 854.
- Provee acceso a una interfaz de línea de comandos en la máquina destino.
- Muy usado, no obstante, para testear conectividad (éste es el uso que le daremos en el contexto del trabajo).
- Hoy por hoy su uso para acceso remoto quedó obsoleto por ser poco seguro (a diferencia de SSH).

telnet : un ejemplo

- Veamos cómo usar telnet para conectarnos a un servidor de mails:

```
lucio@knuth:~$ telnet proxymail1.sion.com 25
Trying 200.81.186.15...
Connected to proxymail1.sion.com.
Escape character is '^]'.
220 mx2.sion.com ESMTP (SION)
QUIT
221 2.0.0 Bye
Connection closed by foreign host.
```

netstat

- Sirve para ver y analizar las conexiones TCP (...y UDP).
- Puede indicar los procesos ligados a cada socket.
- Muestra el estado de las conexiones TCP (según el famoso diagrama de estados que ya conocemos).

netstat : fragmento de una corrida

```
lucio@knuth:~$ netstat -an | more
(...)
Proto   Local Address         Foreign Address          State
tcp     127.0.0.1:631          0.0.0.0:*               LISTEN
tcp     127.0.0.1:5432         0.0.0.0:*               LISTEN
tcp     0.0.0.0:902            0.0.0.0:*               LISTEN
tcp     0.0.0.0:80             0.0.0.0:*               LISTEN
tcp     192.168.1.101:46670    65.55.71.176:1863     ESTABLISHED
tcp     192.168.1.101:37853    74.125.130.125:5222   ESTABLISHED
tcp     192.168.1.101:57385    192.168.1.100:22      ESTABLISHED
(...)
```

`netcat`

- La navaja suiza para TCP/IP.
- Múltiples y variadas funcionalidades:
  - Hablar con servidores (como hicimos con `telnet`)
  - Escaneo de puertos
  - Transferencia de archivos
  - Escuchar en un puerto dado

netcat : ejemplo de port scanning

```
lucio@knuth:~$ nc -vz 192.168.1.100 77-83
nc: connect to 192.168.1.100 port 77 (tcp) failed
nc: connect to 192.168.1.100 port 78 (tcp) failed
nc: connect to 192.168.1.100 port 79 (tcp) failed
Connection to 192.168.1.100 80 port [tcp/www] succeeded!
nc: connect to 192.168.1.100 port 81 (tcp) failed
nc: connect to 192.168.1.100 port 82 (tcp) failed
nc: connect to 192.168.1.100 port 83 (tcp) failed
```

netcat : ejemplo de transferencia de archivos

```
lucio@knuth:~$ nc -l localhost 12345 > file &
[1] 5557
lucio@knuth:~$ echo "eh amigo" > eh_amigo
lucio@knuth:~$ nc localhost 12345 < eh_amigo
[1]+  Done                    nc -l localhost 12345 > file
lucio@knuth:~$ cat file
eh amigo
```

nmap

- Es una herramienta de discovery muy famosa y muy completa:
  - Discovery de hosts (similar a lo que hicimos en el taller anterior, aunque por supuesto más completo)
  - Escaneo de puertos (implementa todas las técnicas mencionadas hace un rato y algunas otras más esotéricas)
  - Detección de servicios
  - Detección de sistemas operativos
- Además tiene soporte para scripts custom.
- Apareció por 1997 y tiene soporte para múltiples plataformas.
- El nombre viene de network mapper.

nmap : fragmento de una corrida

```
lucio@knuth:~$ sudo nmap -sS -sV 192.168.1.100

Starting Nmap 5.21 ( http://nmap.org ) at 2012-10-30 01:56 ART
Nmap scan report for 192.168.1.100
Host is up (0.013s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 5.3p1 Debian 3ubuntu6 (protocol 2.0)
80/tcp open  http?
1 service unrecognized despite returning data.
MAC Address: 00:24:8C:96:57:8B (Asustek Computer)
Service Info: OS: Linux

Service detection performed. Please report any incorrect results
at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 134.77 seconds
```

Agenda

1. Discovery a nivel transporte
   - Técnicas de port scanning
   - Detección de servicios
   - Detección de sistemas operativos
   - Herramientas
2. Consignas

Primera parte: implementación de port scanning

- Implementar SYN scan y Connect scan.
- ¿Cuál es la complejidad algorítmica de estos métodos? Comparar tiempo contra cantidad de operaciones.
- Discutir sobre cuán inocuos resultan estos métodos.
- ¿Es posible tomar conciencia de que una detección de puertos está siendo realizada?
- ¿En qué escenarios son efectivos? ¿En cuáles carecen de utilidad?
- Considerar el siguiente escenario: se desconoce la existencia de un firewall intermedio. ¿Qué se podría determinar con esta técnica?

Segunda parte: banner grabbing

- Documentar al menos tres servicios susceptibles a la técnica de banner grabbing.
- Dar una implementación para la captura de los banners en tales servicios.

Tercera parte: OS fingerprinting vía `nmap`

- Correr `nmap` especificando la opción `-O` (OS fingerprinting) contra al menos dos hosts conocidos con distinto sistema operativo (e.g., Windows y Linux). Analizar la precisión de los resultados arrojados por `nmap` y sacar conclusiones.
- Repetir la actividad anterior capturando el tráfico generado con Wireshark. A partir de lo observado, contestar las siguientes preguntas:
  - ¿Puede identificarse un proceso de escaneo de puertos en las capturas? Si es así, indicar qué algoritmo se utiliza y qué resultados se obtuvieron.
  - Identificar alguna de las pruebas de `nmap` mencionadas en [2]. ¿Qué tipo de paquetes se envían? ¿Qué características tienen? (i.e., flags, opciones, etc.).
  - En caso de haber una prueba en común para los hosts analizados, estudiar si las respuestas generadas contienen diferencias.

Referencias

Nmap Port Scanning Techniques
`http://nmap.org/book/man-port-scanning-techniques.html`

TCP/IP Fingerprinting Methods Supported by Nmap
`http://nmap.org/book/osdetect-methods.html`

S. McClure, J. Scambray, G. Kurtz (2012)
Hacking Exposed 7: Network Security Secrets & Solutions - Cap. 2: Scanning
New York: McGraw-Hill.