Resultados de un análisis de seguridad de los protocolos TCP e IP

Fernando Gont

(proyecto realizado para UK CPNI)

Congreso Seguridad en Cómputo
19 al 26 de septiembre de 2008, Ciudad de México, México

Agenda

- Motivación para la realización del proyecto (enunciado del problema)
- Breve descripción de algunas áreas de trabajo
- Resultados preliminares
- Discusión sobre algunos aspectos de seguridad en TCP

Enunciado del problema

- Durante los últimos veinte años, el descubrimiento de vulnerabilidades en implementaciones de los protocolos TCP/IP, y en los propios protocolos, han llevado a la publicación de un gran número de reportes de vulnerabilidad por parte de fabricantes y CSIRTs.
- Como resultado, la documentación de todas estas vulnerabilidades se encuentra esparcida en una gran cantidad de documentos que suelen ser difíciles de identificar.
- Asimismo, algunos de estos documentos proponen contramedidas a las mencionadas vulnerabilidades, sin realizar un análisis minucioso de las implicancia de las mismas sobre la interoperabilidad de los protocolos.
- Desafortunadamente, el trabajo de la comunidad en esta área no ha reflejado cambios en las especificaciones correspondientes de la IETF.

Situación actual

- Se hace notablemente dificultoso realizar una implementación segura de los protocolos TCP/IP a partir de las especificaciones de la IETF.
- Nuevas implementaciones de los protocolos re-implementan vulnerabilidades encontradas en el pasado.
- Nuevos protocolos re-implementan mecanismos o políticas cuyas implicancias de seguridad ya eran conocidas a partir de otros protocolos (por ejemplo, RH0 en IPv6).
- No existe ningún documento que apunte unificar criterios sobre las vulnerabilidades de los protocolos, y las mejores prácticas para mitigarlas.
- No existe ningún documento que sirva como complemento a las especificaciones oficiales, para permitir que la implementación segura de los protocolos TCP/IP sea una tarea viable.

Descripción del proyecto

- En los últimos años, UK CPNI (Centre for the Protection of National Infrastructure) – antes UK NISCC (National Infrastructure Security Co-ordination Centre) – se propuso llenar este vacío para los protocolos TCP e IP.
- El objetivo fue producir documentos que sirvieran de complemento a las especificaciones de la IETF, con el fin de que, mínimamente, nuevas implementaciones no posean vulnerabilidades ya conocidas, y que las implementaciones existentes puedan mitigar estas vulnerabilidades.
- Dichos documentos se irían actualizando en respuesta a los comentarios recibidos por parte de la comunidad y al descubrimiento de nuevas vulnerabilidades.
- Finalmente, se espera llevar este material al ámbito de la Internet Engineering Task Force (IETF), para promover cambios en los estándares correspondientes.

Algunas áreas de trabajo en IP

- Rango de valores aceptables para cada campo del encabezamiento
  - En algunos casos, los rangos aceptables dependen del valor de otros campos. Ejemplo: IHL (Internet Header Length), Total Length, *link-layer payload size.*
- Análisis de las posibles implicancias de seguridad de cada mecanismo y política del protocolo.
  - Ejemplo: El campo TTL se puede utilizar (al menos en teoría) para OS fingerprinting, physical-device fingerprinting, TTL-triangulation, evasión de NIDS, GTSM, etc.
- Procesamiento deseable de las distintas opciones IP
  - Ejemplo: source-routing? IP Security options?
- Analizar distintas políticas posibles para aplicar junto con distintos mecanismos. Por ej., en el caso del reensamble de fragmentos IP:
  - ¿Qué chequeos de validación podrían realizarse para evitar la evasión de NIDS? ¿Qué políticas se podrían implementar para minimizar ataques de DoS?

Algunas áreas de trabajo en TCP

- Establecer claramente el rango de valores aceptables para cada campo del encabezamiento y opciones
  - Ejemplo: Valores aceptables para la opción TCP MSS (Rose attack)
- Analizar posibles algoritmos para la selección de puertos efímeros.
- Reducir las posibilidades de abusar de los algoritmos de control de congestión de TCP.
- Analizar posibles algoritmos para el manejo del buffer de reensamblado, y del buffer de retransmisión de datos.
- Analizar como reducir la precisión de técnicas de “remote OS finger-printing”.
  - ¿No es **demasiada** la precisión de nmap? ¿Realmente necesita cada versión de un sistema operativo de cada fabricante hacer algo distinto? ¿No se pueden unificar criterios?

Resultados preliminares

- Para el caso del protocolo IP, se generó un documento de 50 páginas, con mas de 70 referencias a reportes de vulnerabilidad y papers relevantes. El mismo se encuentra disponible en: http://www.cpni.gov.uk/Products/technicalnotes/3677.aspx
- Para el caso del protocolo TCP, se generó un documento de más de 100 páginas, con más de 100 referencias a reportes de vulnerabilidad y papers relevantes. Este documento todavía no ha sido publicado.
- Los documentos se beneficiaron de los comentarios de desarrolladores de implementaciones TCP/IP, tanto abiertas como cerradas.

Cooperación

- Hemos tenido el agrado de contar con una variedad de ingenieros e investigadores del área para la revisión de los documentos sobre seguridad en TCP e IP de este proyecto, lo cual ha contribuido muy positivamente con nuestro trabajo.
- Lamentablemente, la situación no ha sido la misma en lo que respecta a fabricantes
  - En muchos casos, no se recibió respuesta alguna.
  - En algun caso, respuestas del tipo “como este proyecto no fue patrocinado por nuestra compañía, no podemos hacer comentarios técnicos” (¿?¡!)

Actividades relacionadas en la IETF (I)

- Algunas porciones de este proyecto ya se llevaron a la IETF. Ejemplos:
  - “Security Assessment of the Internet Protocol”: El mismo documento publicado por CPNI en agosto de este año fue publicado recientemente como Internet-Draft. Todavía no ha sido adoptado como elemento de trabajo del OPSEC WG.
  - Aleatorización de puertos: Se presentó un documento que fue adoptado por el TSVWG (BCP).
  - Ataques ICMP contra TCP: Se presentó un documento que fue adoptado por el TCPM WG (Informational) ☹.
- Algunas otras publicaciones en el ámbito de la IETF, no vinculadas con este proyecto:
  - “Recommendations for filtering ICMP messages” (draft-ietf-opsec-icmp-filtering-00.txt): Este documento fue adoptado como WG item por el OPSEC WG. (es increíble que en el 2008 todavía muchos crean que “se puede filtrar todo ICMP”).

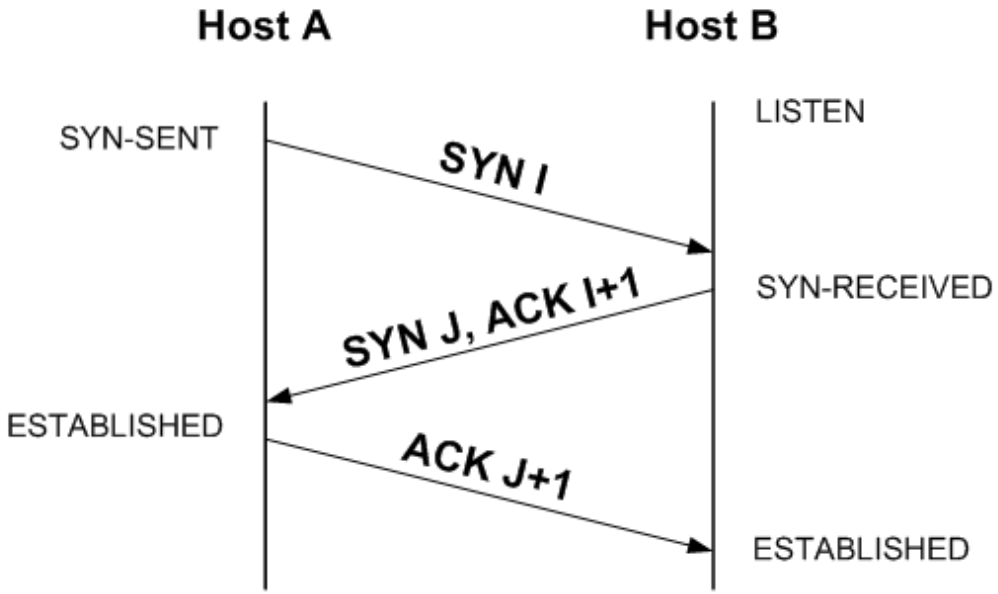
Actividades relacionadas en la IETF (II)

- Esta actividad suele requerir una gran cantidad de energía
  - Con el fin de lograr consenso, las propuestas presentadas en la IETF suelen tener que ser modificadas a niveles en los cuales el documento final termina difiriendo notablemente de la propuesta original.
  - Existe cierta resistencia a realizar modificaciones en IPv4 (ya que "*IPv6 reemplazará a IPv4*")
  - Los fabricantes suelen resistirse a implementar modificaciones vinculadas a seguridad

Breve revisión de algunos conceptos básicos de TCP

Establecimiento de conexión

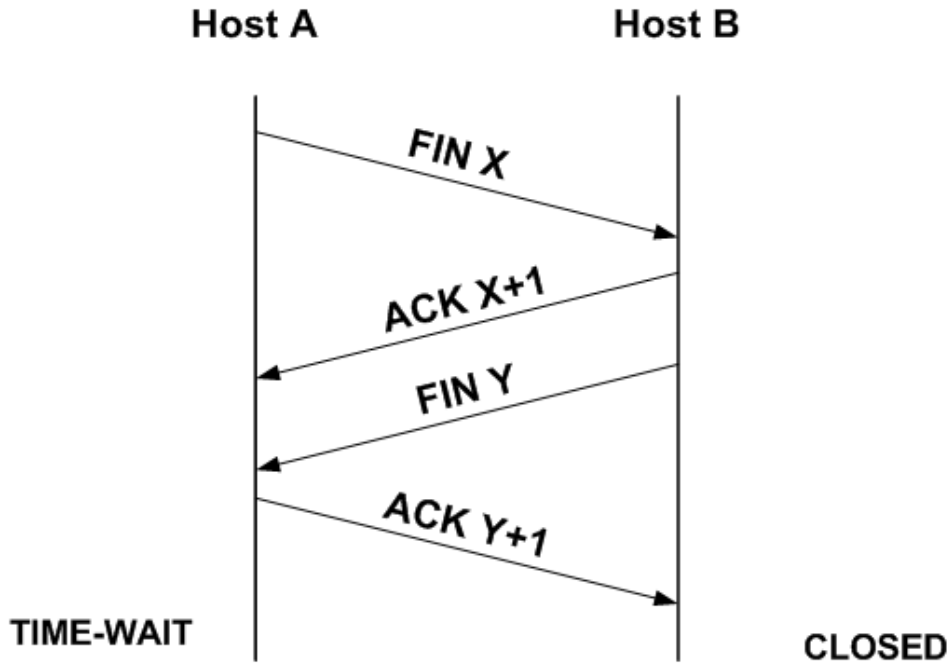
- El proceso de establecimiento de conexión consiste usualmente en el intercambio de tres segmentos.



- Una vez finalizado el “three-way handshake”, los números de secuencia (y demás parámetros) estará sincronizados, y se podrá comenzar la transferencia de información.

Cierre de conexión

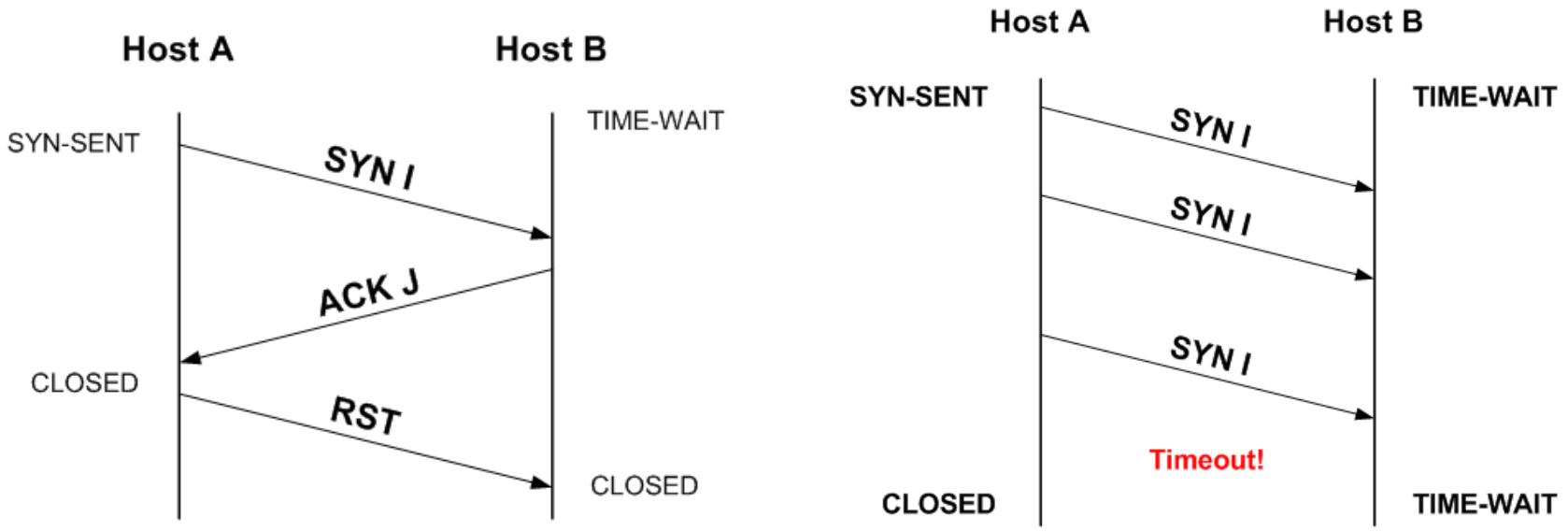
- El proceso de cierre de conexión consiste usualmente en el intercambio de cuatro segmentos:



- El extremo que inicia el proceso de cierre de conexión (Host A) usualmente queda en el estado TIME-WAIT por 4 minutos, mientras que el otro extremo queda en el estado ficticio CLOSED.

Colisiones de connection-id’s

- Debido al estado TIME-WAIT, puede suceder que al realizarse una petición de conexión exista una encarnación previa de la misma conexión en el sistema remoto. En tal caso, la petición de conexión fallará.



- Claramente, esta situación (colisión de connection-id’s) es indeseable, por lo cual, en la medida que sea posible, deberá ser evitada.

Discusión de algunos aspectos de seguridad de TCP

TCP Initial Sequence Numbers

TCP Initial Sequence Numbers (I)

- El RFC 793 menciona que los ISN's se han deben resultar en una secuencia monótonamente creciente, a partir de un timer global, con el fin de evitar la superposición de espacios de secuencia. Desde ahí (año 1981), se ha asumido que la generación de ISN's a de forma lineal es crucial para la confiabilidad de TCP (protección contra segmentos "viejos").
- Sin embargo, la verdadera protección contra segmentos viejos está provista por otros dos mecanismos que no tienen relación alguna con los ÍSN:
  - "Quiet time concept": Cuando se reinicia el sistema, se debe esperar 2*MSL antes de transmitir segmentos TCP.
  - TIME-WAIT state: Cuando se finaliza una conexión TCP, el extremo que realizó el "active close" debe permanecer en el estado TIME-WAIT por 2*MSL, asegurando así que los segmentos "viejos" desaparezcan de la red.

TCP Initial Sequence Numbers (II)

- En la implementación tradicional BSD, el generador de ISN se inicializaba en 1 al iniciar el sistema, y se incrementaba en 64000 cada medio segundo, y en 64000 por cada conexión establecida.
- Basado en la suposición de que los ISNs son generados linealmente, BSD implementó una modificación al comportamiento estandar de TCP, con el fin de permitir altas tasas de petición de conexión. Si se recibe un SYN para una conexión que se encuentra en el estado TIME-WAIT, entonces,
  - Si el ISN del SYN es mayor que el último número de secuencia recibido (SEG.SEQ > RCV.NXT), se remueve el TCB en estado TIME-WAIT, y se crea uno en el estado SYN-RECEIVED.
  - En caso contrario, se siguen las reglas del RFC 793.
- Es interesante mencionar que este “fix” fue motivado por el uso de los comandos r*, es decir, para conexiones cortas, que transmiten **poca** información, y/o a una baja tasa de transferencia. En caso contrario, esta “heurística” falla.

Implicancias de seguridad de los ISNs

- Las implicancias de seguridad de la predictibilidad de ISNs fue descripta por primera vez por Morris en 1985, en el trabajo "*A Weakness in the 4.2BSD Unix TCP/IP Software*".
- La explotación de dicha vulnerabilidad para spoofing the conexiones TCP fue ampliamente publicitada 10 años mas tarde por T. Shimomura en "*Technical details of the attack described by Markoff in NYT*" (el famoso ataque de K. Mitnick).
- Asimismo, una gran cantidad de ataques "ciegos" contra TCP dependen de la habilidad del atacante para para adivinar números de secuencia TCP.
- Debido a ello, es deseable la introducción de aleatorización en la selección de ISNs.

Aleatorización de ISN’s

- Algunas implementaciones (por ej., OpenBSD) decidieron simplemente aleatorizar la generación de ISNs, con el fin de mitigar aquellos ataques basados en la predicción de números de secuencia. Lamentablemente, esto llevó a que en determinados escenarios, las peticiones de conexión fallaran, cuando antes esto no sucedía.
- S. Bellovin propuso, en RFC1948, un algoritmo para la selección de ISNs, que permite que la heurística de las implementaciones BSD pueda seguir funcionando con éxito. Dicho algoritmo propone la selección de ISNs de acuerdo a la expresión:

  ISN = M + F(localhost, localport, remotehost, remoteport, secret)
- Esta función permite que los ISN para conexiones a un determinado end-point sean generados linealmente, a partir de un offset aleatorio, separando así los espacios de números de secuencia.
- Considerando tanto los aspectos de seguridad como de interoperabilidad, es aconsejable implementar un esquema como el propuesto en RFC1948, y **no** un esquema de aleatorización “trivial”.

TCP timestamps

TCP Timestamps

- Las opciones TCP timestamp fueron introducidas por RFC 1323 con dos propósitos:
  - Proveer un mecanismo para medir el Round-Trip Time de la conexión
  - Proteger a las conexiones de segmentos “viejos” de la misma conexión (PAWS), en caso de que se utilicen altas tasas de transferencia de información (PAWS).
- Desde el punto de vista de PAWS, los TCP timestamps vienen a ser una extensión del número de secuencia.
- Para permitir la función de PAWS, los timestamps deben ser monotónicamente crecientes. El RFC 1323 sugiere una frecuencia para los timestamps de entre 1/1s y 1/1/ms.
- Basándose en esta premisa, algunas implementaciones (ej., Linux) permiten la destrucción del estado TIME-WAIT si el SYN entrante tiene un timestamp mayor al último recibido (es decir, un fix similar al de los BSD para el caso de los ISN).

Implicancias de seguridad de los TCP timestamps

- Asimismo, los TCP timestamps tienen proveen al menos dos vectores de ataque previamente inexistentes:
  - Si se logra inyectar un segmento con un timestamp válido, pero mayor al recibido por ese TCP hasta el momento, futuros segmentos legítimos serán descartados (es decir, la conexión se “congelará”).
  - Si el origen de la secuencia de timestamps se incializa a un valor fijo cada vez que se reinicia el sistema, el mismo revelará el uptime del sistema en cuestión.
- Por tal motivo, es deseable la introducción de aleatorización en la generación de los mismos.

Aleatorización de timestamps

- Algunas implementaciones de TCP (por ej., OpenBSD) decidieron aleatorizar los timestamps, con el fin de mitigar los vectores anteriormente descriptos.
- Sin embargo, esta decisión tiene un impacto negativo en la interoperabilidad de los protocolos ya que se rompe la “optimización” de Linux (y otros).
- Para mitigar los vectores mencionados, sin afectar la interoperabilidad de los protocolos, proponemos generar los timestamps de acuerdo a una expresión del tipo (RFC1948):

  TS = M + F(localhost, localport, remotehost, remoteport, secret)

TCP timestamps & TCP ISNs durante el three-way handshake

Procesamiento de SYNs entrantes

- Si existe una encarnación previa de la misma conexión en el estado TIME_WAIT, entonces:
  - Si la encarnación previa utilizaba timestamps, permitir la creación de una nueva encarnación de la conexión si el timestamp del SYN es mayor que el ultimo timestamp visto en ese sentido de la conexión. Si el timestamp del SYN fuera igual al último recibido por la encarnación previa, realizar el chequeo de ISN del SYN (fix de BSD).
  - Si la encarnación previa no utilizaba timestamps, pero el SYN entrante incluye uno, entonces permitir el establecimiento de la conexión.
  - Si ni la encarnación previa ni la nueva utilizan timestamps, entonces simplemente aplicar el chequeo de ISNs al SYN (fix de BSD).

Mejoras en interoperabilidad

- Incluso si existieran colisiones de connection-id’s, la optimización realizada mediante TCP timestamps permitirá una alta tasa de establecimiento de conexiones.
- Si dicha tasa fuera muy alta, y el último timestamp de la conexión coincidiera con el del SYN entrante, el ISN de dicho segmento nos brindaría otra oportunidad para aceptar la conexión.

TCP ephemeral ports

TCP ephemeral ports (I)

- En los últimos años se han divulgado dos familias de ataques “ciegos” contra TCP:
  - “**Slipping in the Window**”: Ataques basados en segmentos TCP falsificados (NISCC vulnerability advisory #236929)
  - “**ICMP attacks against TCP**” : Ataques basados en paquetes ICMP falsificados (NISCC vulnerability advisory #532967)
- Estos ataques pueden ser realizados sin la necesidad de acceder a los paquetes pertenecientes a la conexión atacada, y requieren (como mínimo) que el atacante conozca o pueda adivinar el connection-id (client IP, client port, server IP, server port).
- Mas allá de las posibles soluciones específicas para estas vulnerabilidades, resulta lógico mitigar las mismas dificultando la tarea del atacante para adivinar el connection-id.
- De los cuatro valores que componen al connection-id, el único que en principio puede elegirse arbitrariamente es el puerto TCP del cliente, o el puerto “efímero” de la conexión.

TCP ephemeral ports (II)

- Desde el año 2004 que se publicó un Internet-Draft en materia de aleatorización de puertos efímeros.
- El documento en cuestión (draft-ietf-tsvwg-port-randomization-02.txt) ha sido incluso adoptado por la IETF para su futura implementación con BCP RFC.
- Sin embargo, meses atrás hubo muchísimo trabajo sobre unas vulnerabilidades (Kaminsky sobre DNS) que podrían haber sido en gran parte mitigadas por…. aleatorización de puertos efímeros.

Algoritmos de selección de puertos efímeros

- Al seleccionar un puerto efímero, debe asegurarse que el connection-id resultante (client address, client port, server address, server port) no esté actualmente en uso.
- Si existe en el sistema local una conexión activa con dicho connection-id, se procederá a seleccionar otro puerto efímero, con el fin de salvar el conflicto.
- Sin embargo, es imposible para el sistema local poder detectar si existe alguna instancia de comunicación activa en el **sistema remoto** utilizando dicho connection-id (por ejemplo, una conexión TCP en el estado TIME-WAIT).
- En caso que el puerto efímero seleccionado resultara en un connection-id en uso en el sistema remoto, se dice que se ha producido una “colisión de connection-id’s”, y el intento de conexión usualmente fallará.
- En consecuencia, la frecuencia de reuso de puertos debe ser minimizada.

Rango de puertos efímeros

IANA asigna a los puertos efímeros el rango 49152-65535. Sin embargo, la mayoría de los sistemas eligen sus “puertos efímeros” de un subespacio de todo el espacio de puertos disponible que, en la mayoría de los casos, difiere de aquél elegido por la IANA.

Sistema operativo	Puertos efímeros
Microsoft Windows	1024 - 4999
Linux kernel 2.6	1024 - 4999
Solaris	32768 - 65535
AIX	32768 - 65535
FreeBSD	10000 - 65535
NetBSD	49152 - 65535
OpenBSD	49152 - 65535

Aleatorización de puertos TCP efímeros

- Un buen algoritmo de aleatorización de puertos TCP efímeros debería:
  - Minimizar la predictibilidad de los números de puerto utilizados para futuras conexiones salientes.
  - Minimizar la frecuencia de reuso de puertos (es decir, evitar las “colisiones” de connection-id’s).
  - Evitar conflictos con aplicaciones que dependen de la utilización de números de puertos específicos (por ejemplo, evitar utilizar para los puertos efímeros números de puerto como el 80, el 100, el 6667, etc.)
- Asimismo, y por razones obvias, el rango de los puertos efímeros debería ser maximizado.

Algoritmo de aleatorización básico

```
next_ephemeral = min_ephemeral + random()
                    % (max_ephemeral - min_ephemeral + 1)

count = max_ephemeral - min_ephemeral + 1;

do {
    if(four-tuple is unique)
            return next_ephemeral;

    if (next_ephemeral == max_ephemeral)
        next_ephemeral = min_ephemeral;
    else
        next_ephemeral_port++;

    count--;
} while (count > 0);

return ERROR;
```

Características del Algoritmo

- Ha sido implementado en OpenBSD y FreeBSD
- Produce una secuencia de puertos efímeros muy difícil de predecir
- La frecuencia de reuso de puertos puede ser mucho mayor que la del algoritmo tradicionalmente implementado por los sistemas BSD
- Usuarios de los mencionados sistemas operativos han reportado problemas de interoperatividad. En consecuencia, FreeBSD incorporó un hack que deshabilita la aleatorización de puertos cuando el número de conexiones salientes por unidad de tiempo excede un determinado valor.

Un mejor algoritmo de aleatorización (I)

- El Internet-Draft “Port Randomization” [Larsen, M. y Gont, F., 2008] describe un algoritmo de selección de puertos efímeros basado en una función similar a la propuesta por Steven Bellovin para los ISN:

  Port = counter + F(local_IP, remote_IP, remote_port, secret_key)

- De este modo, se disminuye la predictibilidad de los puertos efímeros (separando el espacio de números de puerto), manteniendo baja la frecuencia de reutilización de puertos efímeros.

Un mejor algoritmo de aleatorización (II)

```
next_ephemeral = 1024;  /*init., could be random */

offset = F(local_IP, remote_IP, remote_port, secret_key);

do {
    port = min_ephemeral + (next_ephemeral + offset)
              % (max_ephemeral - min_ephemeral + 1);
    next_ephemeral++;

    if(four-tuple is unique)
        return port;

    count--;

} while(count > 0);

return ERROR;
```

Secuencia producida por este algoritmo

Nr.	IP:port	offset	min_ephemeral	max_ephemeral	next_ephemeral	port
#1	128.0.0.1:80	1000	1024	65535	1024	3048
#2	128.0.0.1:80	1000	1024	65535	1025	3049
#3	170.210.0.1:80	4500	1024	65535	1026	6550
#4	170.210.0.1:80	4500	1024	65535	1027	6551
#5	128.0.0.1:80	1000	1024	65535	1028	3052

Características del algoritmo

- Implementado en el Linux Kernel
- Produce una secuencia muy difícil de predecir por terceros
- Posee una frecuencia de reuso de puertos igual a la del algoritmo tradicional BSD
- Su implementación es algo más compleja que la de los algoritmos anteriores
- El uso de una función de hashing trae algunas penalidades en materia de performance

Conclusiones

Algunas conclusiones…

- Usualmente se asume que, debido a la antigüedad de los protocolos “core” de la suite TCP/IP, todas las implicancias negativas de seguridad del diseño de los mismos han sido resueltas, o solo pueden resolverse mediante uso de IPsec.
- Las vulnerabilidades publicadas incluso en los últimos cinco años parecen indicar lo contrario.
- Curiosamente, este es el primer proyecto que, en 25 años de utilización de los protocolos TCP e IP, intenta hacer un análisis completo de las implicancias de seguridad de los mismos.
- La respuesta de la comunidad a este proyecto ha sido muy positiva. Sin embargo, la colaboración por parte de fabricantes no fue la esperada.

Preguntas?

Agradecimientos

Mis agradecimientos a:

- UK CPNI y UTN/FRH por su apoyo en mis actividades
- los organizadores de este Congreso

… y a Uds, quienes presenciaron esta conferencia.

Información de contacto

Fernando Gont

fernando@gont.com.ar

Más información en:

http://www.gont.com.ar