Evasión de Sistemas de Detección de Intrusos

Cómo atacar sin ser detectado

Roberto Gutiérrez Gil

Índice

1. - Qué es un IDS?

Un sistema de detección de intrusos, tal y como indica su nombre permitirá detectar ataques o actividad no autorizada que vulnera las políticas de seguridad en una red o sistema. Los sistemas de detección de intrusiones están compuestos básicamente por tres elementos:

- Una fuente de información que proporciona eventos de sistema. Estos eventos puede ser desde el propio tráfico de la red hasta los logs de cierta aplicación.
- Un motor de análisis que busca evidencias de intrusiones. El motor de análisis se encargará de encontrar los patrones de ataques o un comportamiento anómalo que evidencia el posible ataque de un intruso que intenta penetrar en al red o e un sistema.
- Un mecanismo de respuesta que actúa según los resultados del motor de análisis. Dependiendo de si el IDS ha detectado un posible ataque, generará en muchos casos una alerta que será visible por el administrador o incluso, en algunas ocasiones, podrá tomar medidas él mismo para paliar el ataque. Esto último es el caso de los IPS (Intrusion Prevention System).

2. - Categorías de IDS

Podemos clasificar los sistemas de detección de intrusiones dependiendo del tipo de fuente de información que utiliza, del motor de análisis y del mecanismo de respuesta que presenta.

- **NIDS vs HIDS:** Los sistemas de detección de intrusiones de red, o NIDS, analizarán todo los paquetes que viajen través de la red que vigila. Los NIDS podrán detectar entre otras cosas paquetes maliciosos diseñados especialmente para realizar un ataque, conexiones no autorizadas, paquetes malformados intencionadamente, etc. Los sistemas de detección de intrusiones a nivel de host, (HIDS) podrán examinar cualquier elemento del sistema que delate una posible intrusión. Los HIDS podrán analizar logs de las aplicaciones, detectarán cambios significativos en el sistema de ficheros, realizarán registros de la CPU, del disco, de la memoria, etc. Podemos encontrar dentro de esta clasificación un tercer tipo IDS, que más bien hace referencia a la infraestructura. Son los **DIDS** o sistemas de detección de intrusos distribuidos. Una configuración muy común hoy en día es tener diversos sensores (IDS) de distinto tipo distribuidos a través de la red. Todos los sensores emitirán las alertas a una base de datos centralizada en un servidor de seguridad, donde se podrán realizar tareas más complicadas de análisis de ataques como puede ser la correlación de eventos.

- **Detección de patrones vs Detección de anomalías:** En la detección de patrones el IDS analizará la información entrante y la comparará con una gran base de datos con firmas de ataques conocidos. Es importante puntualizar que un sistema de detección de patrones no podrá detectar nuevos ataques ni variantes de ataques que no coincidan exactamente con alguna firma de su base de datos. Al trabajar de una forma similar a la de los antivirus comunes, es evidente que la precisión del IDS dependerá totalmente de las firmas de su base de datos. En los sistemas basados en detección de anomalías, se define cual va a ser el comportamiento normal de la red para después detectar comportamientos anómalos que evidencien ataques o anomalías. Estos IDS, en casi todos los casos precisan de una fase de aprendizaje para que puedan conocer cual es el estado normal de la red.

- **Pasivos vs Reactivos:** En un IDS pasivo cuando se detecte un problema de seguridad logeará la información necesaria y emitirá una alerta que deberá ser vista por el administrador de seguridad. Por otro lado tenemos los IDS reactivos que responderán ellos mismos contra los ataques o actividades sospechosa, por ejemplo, cortando la conexión, bloqueando tráfico en el firewall, etc.

3. - Problemática de los IDS

Un sistema de detección de intrusos a pesar de ser una herramienta esencial dentro de los sistemas de seguridad presentan importantes limitaciones y problemas.

Uno de los inconvenientes más importantes es el de las falsas alarmas: **falsos positivos** y **falsos negativos**. Los **falsos positivos** consisten en aquellas alarmas que se generan cuando en realidad no se está produciendo ningún ataque, intrusión o abuso. Dentro de los sistemas de detección por firmas o patrones, es frecuente que una firma (que en muchos casos no es más que una sucesión de bytes) aparezca dentro del flujo de tráfico lícito y levante una alarma cuando no existe el ataque. Por otro lado los IDS basados en anomalías presentan también un problema similar. Pueden detectar como tráfico hostil la aparición de un nuevo tipo de tráfico generado, por ejemplo, por la instalación de un nuevo servicio.

El otro tipo de falsa alarma son los **falsos negativos** y tienen que afectan precisión del propio IDS, ya que se trata de ataques reales que no han podido ser detectados. Otro problema importante que se deriva, es la excesiva cantidad de eventos o alarmas (en un porcentaje alto de falsos positivos) que se pueden generar en una red de mediano tamaño. Lo más negativo de esta cuestión es que la continua aparición de falsos positivos puede hacer que un administrador acabe ignorando las alarmas.

La solución al problema de los falsos positivos, pasa por una minuciosa configuración del sistema de detección de intrusos adaptándolo al entorno que va a monitorizar. Y por el uso metodologías y técnicas de correlación de eventos que disminuye de manera considerable la cantidad de falsos positivos y de alarmas en general. Por supuesto la eficacia del IDS estará estrechamente relacionada con el mantenimiento periódico que reciba, totalmente necesario.

Centrándonos en la problemática a nivel de red de los sistemas de detección de intrusos, podemos encontrar varios tipos de problemas:

El primero, es la insuficiente información con la que cuenta el NIDS para conocer exactamente que está ocurriendo en la red. En muchos casos, los paquetes extraídos, son insuficientes para conocer datos sobre la topología, estado de la red, congestión, tipos de sistemas que intervienen en la comunicación, cómo se comportan esos sistemas a cierto tipo de tráfico, etc. Como veremos más adelante éste desconocimiento por parte del NIDS será uno de sus puntos débiles.

El segundo problema, es que los IDS son inherentemente vulnerables a los ataques de denegación de servicio (DoS) por el simple hecho de ser un sistema que hace uso de determinados recursos limitados, CPU, memoria, ancho de banda, etc. La disponibilidad del NIDS será otro de los principales objetivos de un intruso en la red.

Los NIDS capturan y analizan los paquetes de red en tiempo real para intentar encontrar patrones sospechosos que evidencien un ataque sobre la máquina o máquinas que vigila. El problema es que un NIDS será típicamente una máquina totalmente independiente y diferente a los sistemas que monitoriza y en ocasiones estará situada hasta en un punto diferente dentro de la red. Estas inconsistencias se ven agravadas por ambigüedades en los protocolos, diferencias en las implementaciones, etc.

Por ejemplo, consideremos un NIDS y una máquina localizada en diferentes lugares de la red. Los dos sistemas recibirán cualquier paquete en un tiempo diferente. Éste detalle es muy importante ya que durante ese lapso de tiempo puede ocurrir que, por alguna razón, el paquete no sea procesado en la máquina destino y sin embargo, el NIDS ya lo halla analizado. Otro ejemplo podría ser que un paquete IP con una determinada combinación de banderas incorrecta será rechazado por muchos sistemas operativos, pero por algunos otros, debido a una implementación diferente, el paquete podría ser procesado correctamente ignorando el campo erróneo de la cabecera. Si un IDS no se percata de esto, descartará un paquete que la máquina destino aceptará, reduciendo la efectividad y precisión del sistema.

El IDS no tiene una manera fácil de determinar la situación de la máquina destino, y por eso asumirá que la máquina monitorizada ha aceptado el paquete. Una saturación en la red y un uso exhaustivo de CPU en la máquina final pueden provocar el mismo problema.

La solución se presenta compleja ya que, el monitor pasivo aun conociendo el SO de todas las máquinas de la red, solo con analizar el tráfico, no va a tener forma de saber si cierta máquina va a procesar cierto paquete o no. Y en consecuencia, si el NIDS no procesa todos y cada uno de los paquetes de la misma forma que las máquinas que vigila puede dar lugar a una incorrecta reconstrucción de lo que sucede en la red. Un intruso puede explotar este problema de ambigüedad de la red para realizar diversos tipos ataques sin ser detectado y no es nada fácil defenderse contra este tipo de problema.

Como vemos, un NIDS obtiene toda la información de los paquetes que captura, información totalmente insuficiente para llevar a cabo la detección de intrusiones de forma precisa. El NIDS necesita conocer bien la situación de las máquinas que monitoriza en la red, del mismo modo que las condiciones de tráfico de sus segmentos de red, el tipo de sistema operativo, y lo que es más importante un NIDS debe conocer como se va a comportar cada máquina respecto al tráfico que le llega.

4. - Tipos de ataques

Una vez discutidos los posibles tipos de problemas que presentan los sistemas de detección de intrusiones de red. Vamos a pasar a ver las diferentes formas que existen de aprovechar esas vulnerabilidades y conseguir anular la funcionalidad básica y primordial de reconocimiento de patrones en un NIDS.

Los dos tipos básico de ataques consisten en engañar al analizador de red para que vea un tráfico diferente al que ve realmente la máquina objetivo. Para conseguirlo el intruso alterará su propio flujo de tráfico y creará situaciones anómalas o erróneas que el NIDS y la máquina destino manejarán de diferente modo.

El primer tipo de ataque se llama "**inserción**" y consistirá en insertar paquetes inválidos que el analizador tomará como paquetes lícitos al contrario que la máquina destino que los descartará. El segundo tipo de ataque se llama "**evasión**", el atacante intentará explotar las inconsistencias que hablamos al principio entre el analizador y el sistema final para que ciertos paquetes pasen sin más por el analizador.

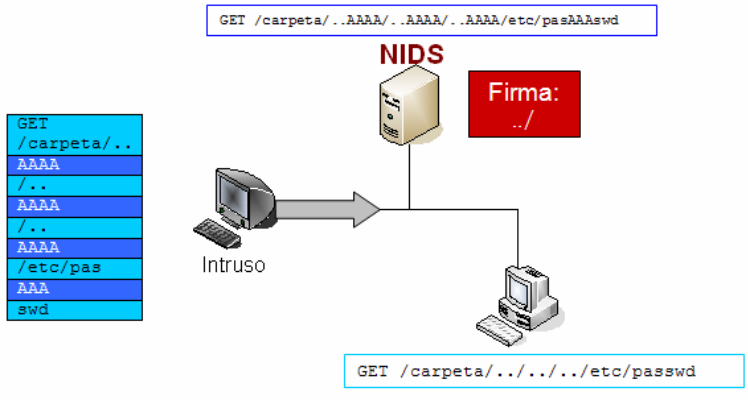
4.1. - Ataques de Inserción

Debido a la carencia de información de la que dispone, un NIDS puede aceptar un paquete pensando erróneamente que el paquete ha sido aceptado y procesado en el sistema final cuando realmente no lo ha hecho. Un atacante puede explotar esta condición enviando paquetes al sistema final que rechazará, pero que el IDS pensará que son válidos. Haciendo esto, el atacante estará "insertando" tráfico en el IDS.

Para entender porque la inserción hace fallar el análisis de firmas, es importante entender la técnica empleada por los IDS. El análisis de firmas usan algoritmos de patrones para detectar cierta cadena de bytes en un flujo de datos. Por ejemplo, para evitar un ataque de "transversal directory" un IDS intentará detectar una cadena como "`../`" en la URL de una petición http. De esta forma, es normal que el NIDS genere una alerta cuando encuentre una petición Web como "`GET /carpeta/../../../etc/passwd`". ¿Pero que ocurriría si el intruso consigue inyectar paquetes que no serán aceptados en la máquina atacada pero que el IDS si procesará?. Pues conseguiría que el NIDS ve una petición diferente y no detectara el ataque

```
GET /carpeta/ ..AAAA/..AAAA/..AAAA/etc/pasAAAswd
```

El intruso ha usado un ataque de inserción para añadir los caracteres 'A' al flujo original. El IDS ya no detectaría el ataque porque no encuentra literalmente la cadena "../" que era su patrón para encontrar este tipo de ataques.



Como vemos en el diagrama anterior, la petición del intruso se fragmente en trozos más pequeños e intercala paquetes especialmente formados (azul oscuro) que serán procesados por el IDS pero no por la máquina objetivo. Con éste ataque de inserción el NIDS es incapaz de encontrar la firma en la petición http y por tanto no detecta el ataque.

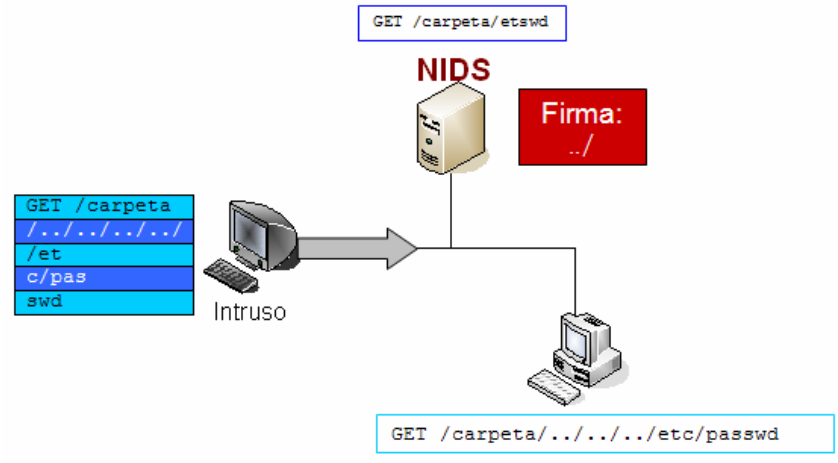
En general, los ataques de inserción ocurren cuando el IDS es menos estricto en el procesamiento de paquetes que la máquina atacada. Por lo tanto una primera solución para evitar este tipo de ataques podría ser hacer el IDS lo más estricto posible en el procesamiento de paquetes, pero como veremos más adelante, esta opción genera otro tipo de problemas.

4.2. - Ataques de Evasión

Son muy similares a los anteriores ataques de inserción, pero sucede la situación contraria: Una máquina de la red puede aceptar un paquete que el NIDS desecha erróneamente perdiendo todo su contenido. Información que será vital para la correcta detección de la alerta. Esto es debido a que el procesamiento de paquetes del NIDS es más restrictivo que el qué realiza la máquina final.

Estos ataques de evasión pueden tener un impacto de seguridad importante ya que sesiones enteras pueden ser forzadas a evadir un NIDS y por consiguiente que no se detecte ni la más mínima alerta a nivel de red del ataque.

Como vemos en la figura siguiente, los ataques de evasión burlan la búsqueda de patrones de una manera similar a los ataques de inserción. De nuevo, el atacante que consigue que el IDS vea un flujo de datos diferente al del sistema final.



La petición HTTP, de nuevo, la fragmentamos en distintas partes. Los paquetes azul oscuro evaden el NIDS y no son procesados por el monitor, en cambio le máquina objetivo procesa todos los paquetes normalmente.

Al contrario de lo que puede parecer los ataques de inserción y evasión no son fáciles de explotar. Por suerte para el administrador de red, un intruso, en la mayoría de ocasiones no tiene ni la facilidad ni la flexibilidad para inyectar paquetes en un flujo de datos y que solo ciertos sistemas procesen ciertos paquetes.

A pesar de ello, en las siguientes dos secciones ser verán ejemplos de ataques de evasión e inserción en el mundo real, basándose por supuesto en los protocolos de los niveles de red y transporte.

4.3. - Denegación de Servicio

Otro de los ataques más populares, son los ataques de denegación de servicio (DoS) que intenta comprometer la disponibilidad del un recurso. Entre los más conocidos se encuentran los SynFloods, MailBomb, PingFlod y el famosísimo "Ping de la Muerte". Todos estos ataques intentan consumir desproporcionadamente montones de recursos para que los procesos legítimos no los puedan utilizar. Otros ataques de DoS están orientados a bugs en el software que al explotarlos pueden llegar a bloquear el sistema.

Cuando un dispositivo de red recibe un ataque de denegación de servicio, pueden ocurrir dos cosas: que el dispositivo deje de dar servicio bloqueando la red, que sería el caso más común de un router o un buen firewall. O bien, la otra posibilidad es que el dispositivo de red deje de dar servicio pero permita completamente que cualquier tipo de trafico pase a través de él, este es el caso de los NIDS. Son sistemas pasivos que no pueden actuar sobre la red y por lo tanto en caso de recibir un ataque de DOS actuarían en modo "fail-open". Si un atacante consigue bloquear el IDS o sus recursos, tiene total libertad para realizar ataques a la red sin peligro de ser detectado por los NIDS.

Algunas denegaciones de servicio surgen de un error en el software: Un IDS falla al recibir cierto paquete o una serie de mensajes. Afortunadamente, este tipo de errores son rápida y fácilmente corregidos por los fabricantes. Es también interesante hablar de los IPS, que puede ser inundados con falsos positivos (paquetes maliciosamente malformados) para que reaccionen contra ataques que nunca han ocurrido y bloqueen el tráfico de ciertas máquinas consiguiendo un ataque de DoS.

Hay muchos tipos diferentes de DoS que afectan a los sistemas de detección. Por lo general estos ataques involucran el agotamiento de algún recurso. El atacante identifica algún punto de la red que requiere la asignación de algún recurso y causa una condición que consume todo el recurso:

Agotamiento de CPU: Consiste en el agotamiento de las capacidades de procesamiento de un IDS, haciéndole "perder el tiempo" en trabajo inútil y así evitar que realice otras operaciones. Un ejemplo concreto es la fragmentación: un atacante puede forzar un IDS a procesas gran cantidad de fragmentos incompletos y lo más pequeños posibles para consumir ciclos de CPU.
Agotamiento de la memoria: Un atacante puede determinar que operaciones requieren una mayor utilización de la memoria y forzar al IDS para asignar toda la memoria a información inútil para facilitar algún tipo de ataque. Un IDS necesita memoria para una gran cantidad de cosas: almacenar el estado de conexiones TCP, reensamblado de fragmentos, búferes necesarios para la detección de firmas, etc.
Agotamiento de Disco: En algún punto, ya sea en fichero, BD o en algún otro formato, el IDS necesitará almacenar los logs y las alertas de l tráfico de red. Un atacante puede crear un flujo de eventos sin importancia que continuamente sean almacenados en el disco y poco a poco ir completando el espacio del disco necesario para almacenar eventos reales
Agotamiento de Ancho de Banda: Los sistemas de IDS deben llevar un seguimiento de toda la actividad de las redes que monitoriza. Pocos IS pueden hacerlo cuando las redes están extremadamente llenas de carga. Un IDS, a diferencia que una máquina final, deben leer todos los paquetes de la red, no solo los destinados a ella. Un atacante puede sobrecargar la red y evitar que el IDS detecte correctamente todo lo que ocurre en la red.

Desafortunadamente este tipo de ataques son extremadamente difíciles de defender. La sobrecarga de recursos no es fácil de solventar y hay muchos puntos diferentes en los que los recursos de un IDS pueden ser consumidos. Un punto importante por supuesto será la correcta configuración, securización y bastionado del NIDS.

4.4. - Inundación de alertas

Otro tipo de ataque contra la precisión y eficacia de los NIDS, es la inundación de alertas con el objetivo de enmascarar y ocultar el verdadero ataque. Un intruso puede conseguir fácilmente que un NIDS genere gran cantidad de alertas de falsos ataques mientras se realiza el verdadero ataque que solo genera unas pocas alertas. El objetivo es que la verdadera alerta pase inadvertida para el administrador o la tome como un falso positivo al igual que el resto de alertas falsas. Otra implicación que puede tener la inundación de alertas es la del agotamiento del ancho de banda, antes comentado, que puede mermar la capacidad del NIDS para detectar el verdadero ataque.

4.5. - Ataques de spoofing

El término spoofing hace referencia al uso de técnicas de suplantación de identidad generalmente con objetivos maliciosos Existen diferentes tipos de spoofing que dependerán del entorno donde se produzca. Algunos ejemplos pueden ser el IP spoofing (quizás el más conocido), ARP spoofing, DNS spoofing, Web spoofing o e-mail spoofing,etc.

Como hemos dicho el IP-spoofing es una de las técnicas más conocidas, debido al desafortunado hecho de que el protocolo Ipv4 no tiene forma de autenticación: cualquier paquete puede provenir de cualquier host. Esto supone algunos problemas para el IDS que puede ser engañado al analizar solamente la información basada en las cabeceras IP y creer algo que realmente no está pasando en la red. Este tipo de tipo de ataques tiene un especial impacto en los protocolos no orientados a conexión donde no existe ninguna clase de código de control de secuencia.

Por suerte, hoy en día hay que tener en cuenta que los routers actuales no admiten el envío de paquetes con IP origen no perteneciente a una de las redes que administra (los paquetes spoofeados no sobrepasarán el router) ni tampoco aceptan paquetes entrantes con IP origen alguna de alguna de las redes internas. Estas dos características mitigan notablemente la posibilidad de realizar ataques de IP-spoofing.

5. - Ataques en la capa de red

Durante este capítulo se van a explicar algunas de las formas que tendrá el intruso para realizar ataques de inserción o evasión de NIDS. Se hacen especialmente críticos los ataques realizados en esta capa del modelo red, ya que se verán afectados también, los protocolos de las capas superiores.

5.1. - Cabeceras Malformadas

Para realizar ataques de inserción simples el intruso debe generar paquetes que se rechacen en la máquina objetivo pero que sean procesados por el NIDS. La forma más sencilla de que un datagrama sea descartado es que tenga una cabecera no válida. Como a priori el objetivo del NIDS no es el validar los campos de un paquete, puede ser relativamente sencillo que procese paquetes malformados sin percatarse de que no son del todo correctos.

La definición de la cabecera IP y de todo el protocolo lo podemos encontrar en el RFC791 (http://www.ietf.org/rfc/rfc0791.txt).

Consultando el código fuente del driver IP de Linux podemos ver cuando un paquete IP será descartado :

- La versión no es '4'.
- Checksum incorrecto
- El tamaño total del datagrama es menor que el de la cabecera IP.
- El campo IPHL (IP header length) es demasiado pequeño ó es más grande que todo el paquete.
- Opciones incorrectas.

El fichero consultado ha sido "linux\net\ipv4\ip_input.c" de la versión del kernel 2.1.0. (He elegido esta versión de kernel por simplicidad).

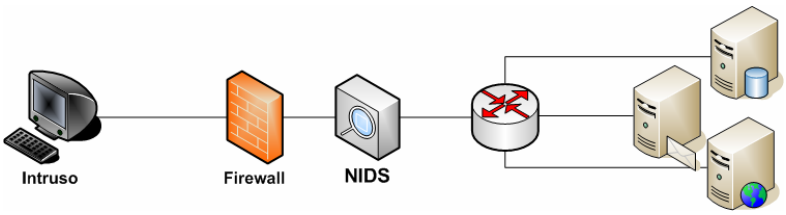
El problema de intentar usar paquetes con cabeceras malformadas en ataques de inserción, es que dichos paquetes no van a ser reenviados por los routers, que detectarán los paquetes como erróneos. Por esta razón se hace complicado utilizar éste tipo de técnica para engañar a los NIDS a menos que nos encontremos en la misma red local que el analizador de red.

Un ejemplo de ataque de inserción consistiría, por ejemplo, en intercalar paquetes con el checksum erróneo en el flujo de datos. Si el NIDS no comprueba la validez de este campo procesará más datagramas que la máquina destino viendo un flujo de datos distinto.

5.2. - Ataques basados en TTL

El campo TTL dentro de la cabecera IP indica cuantos saltos puede realizar el paquete para llegar a su destino. Cada vez que el paquete pasa a través de un router, éste decrementa en uno el valor del campo TTL. Si el valor llega a cero el paquete es descartado.

Existe un tipo de ataque basado en el TTL, que puede ser tanto de inserción como de evasión, e intenta aprovechar el problema de la diferente localización en la red del NIDS y la máquina atacada. Se verá mas claro con el siguiente ejemplo de red:



Por supuesto para realizar este ataque es necesario un conocimiento previo por parte del atacante, de la topología de red; aunque pueden ser especialmente útiles herramientas básicas como *traceroute* o algo más complejas como *paratrace* para extraer información acerca de la topología.

Supongamos un caso ficticio de la empresa HAL. De dicha empresa sabemos que tiene un firewall protegiendo la red interna del exterior y que solo presenta dos tipos de servicios a Internet: Servicio Web y servicio DNS (Domain Name Resolution) de la zona "*hal-enterprise.com*". Por tanto los únicos puertos abiertos al exterior serán el 53 UDP y el 80 TCP. Sabemos también que utiliza direccionamiento público para todas las máquinas de su red.

El objetivo del ataque será el servidor Web y se intenta explotar una vulnerabilidad en la aplicación Web www.hal-enterprise.com, por lo que dicho ataque podrá ser realizado a través del puerto 80 sin que el firewall sea un impedimento. El único problema que encuentra el intruso, es que desconoce si existe algún otro dispositivo de análisis de tráfico como un NIDS que pueda detectar el ataque. El intruso antes de arriesgarse a atacar y ser detectado intenta recabar más información acerca de la red.

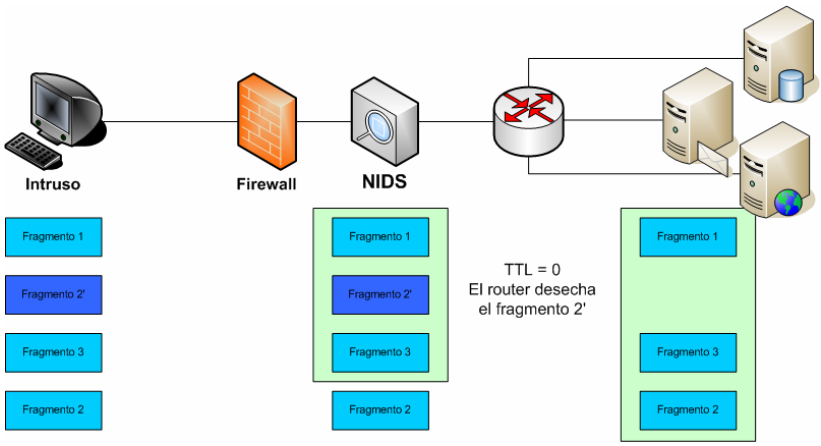
Esta información puede ser obtenida de muy diferentes fuentes, pero en el ejemplo existe un problema grave en el servicio DNS (ya sea por configuración, software no actualizado o vulnerabilidades) que le permite dicha información. El ejemplo más catastrófico sería que el servidor DNS permitiera al atacante realizar una transferencia de zona y obtener todos los nombres de dominio de la zona "*hall-enterprise.com*".

Una vez obtenida esta información el intruso observa un nombre que llama la atención "ids.hall-enterprise.com" ;) Para encontrar la situación exacta de estas dos máquinas en la red, una posibilidad que tiene es realizar un trazado de ruta utilizando paquetes UDP con puerto destino 53 (como el servicio DNS) ya que en el firewall esta permitido el tráfico de estas características. Se podría maquillar aun más el trazado, utilizando paquetes con peticiones DNS o HTTP reales pero para el ejemplo no será necesario.

Para realizar el trazado el intruso utiliza una potente herramienta de generación de paquetes: HPING (http://www.hping.org/). Los comandos ejecutados serían:

```
hacker# hping ids.hall-enterprise.com  --traceroute --udp --destport 53
hacker# hping www.hall-enterprise.com  --traceroute --udp --destport 53
```

Una vez realizadas distintas pruebas, el intruso llega a la conclusión que el NIDS y la máquina objetivo no se encuentran en la misma red y que hay uno o varios routers situados en medio. Es posible entonces generar paquetes que solo verá el NIDS usando el valor TTL justo para que sea procesado por el analizador y que llegue a cero antes de alcanzar la máquina objetivo.



Por supuesto, esta ha sido una situación ficticia donde han coincidido varios problemas graves de seguridad que han permitido el ataque. Con una correcta política de cortafuegos, un buen "*tunning*" del NIDS, un diseño de red seguro y un software actualizado y bien configurado serían prácticamente imposibles ataques de este tipo.

5.3. - Opciones IP

Otro tipo de problema es el relacionado con las opciones de la cabecera IP. El intruso utilizando una combinación adecuada de estas opciones, es capaz de nuevo, de realizar ataques de inserción y evasión. Para evitar estos ataques, el NIDS debería analizar las opciones de cada paquete. Éste proceso adicional no es tan sencillo como verificar un checksum y supone un coste adicional de procesamiento por paquete, además de que no soluciona el problema en todos los casos. Aquí vemos algunas de las posibilidades que tiene un paquete para ser descartado antes de ser procesado por la máquina atacada:

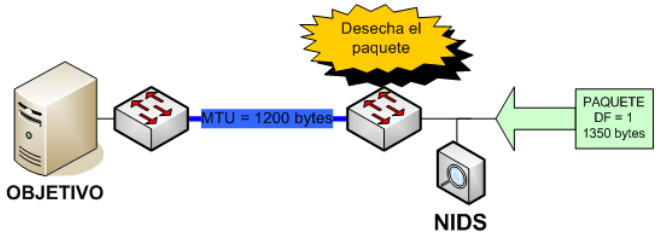
Any Bad option length
Source Route Option offset is less than `4'
Strict Source Route This host is not one of the listed hops
Source Route This host is configured to drop source routed
Source Route No route to next hop in route
Record Route Option offset is less than `4'
Record Route No route to next hop
Timestamp Option length is too short
Timestamp Not enough record space to hold timestamp and IP address
Timestamp Timestamp recording space is full and theoverow counter has wrapped back to zero
Timestamp Bad timestamp type given

Tabla 1 - FreeBSD4.4

Un ejemplo es el uso de la opción *strict source routing* (encaminamiento estricto desde el origen) que especifica la ruta exacta salto a salto. Si la ruta no es posible se produce un error y el router desecha el paquete. Por lo tanto se pueden llegar a realizar ataques muy similares a los basados en TTL. Examinar las opciones de *source route* en paquetes IP parece ser la solución obvia, pero hay más opciones que hay que tener en cuenta y que son tan obvias de procesar.

Otro ejemplo de ataque puede ser la opción *timestamp* que indica que cada router guarde en el paquete el instante en el que el paquete pasa por él. Si no hay sufriente espacio para añadir ese dato de 32 bits, el paquete será descartado por el router.

Un problema similar ocurre con el bit DF de la cabecera IP (Don't fragment). La bandera DF dice a los dispositivos de enrutado que no deben fragmentar dicho paquete, cuando el paquete es demasiado grande para ser reenviado por la red, simplemente es desechado. Si el MTU en la red del NIDS es mayor que en el sistema destino, un atacante puede insertar paquetes haciéndolos demasiados grandes para llegar a la máquina objetivo pero lo suficientemente pequeños para viajar por la red del NIDS y que los procese.



Como contramedida muchos de estos ataques se basan en la emisión de mensajes ICMP de error, informando al emisor de lo que acaba de ocurrir. Un IDS puede potencialmente analizar de dichos mensajes para intentar detectar estos ataques. Esto no siempre es posible e implica mantener el "estado" de cada paquete IP, lo cual supone un consumo de recursos muy importante y un consiguiente ataque de denegación de servicio contra el IDS. Además de que el NIDS en muchos casos no sabrá diferenciar entre un ataque y un error normal en la red.

5.4. - Ataques de fragmentación

El tamaño máximo de un paquete viene definido por diseño de la red. Cuando un paquete supera la MTU de la red (maximium transfer unit) la única posibilidad para que el paquete viaje por la red es fragmentarlo en trozos más pequeños que serán reensamblados en el destino. A pesar de ser un mecanismo esencial del protocolo IP son bien conocidas las implicaciones de seguridad que tiene la fragmentación para dispositivos de análisis de tráfico como firewalls o IDS.

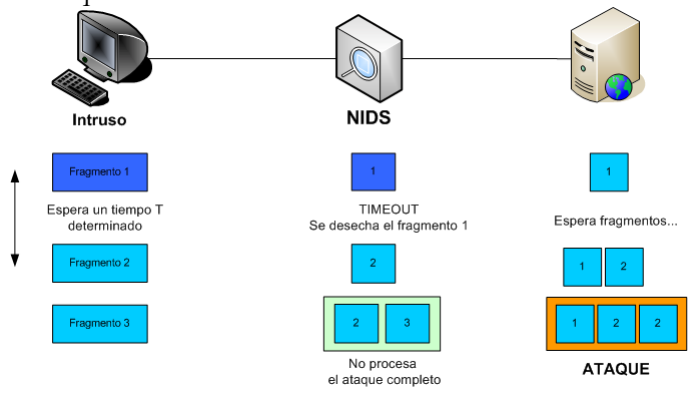
A continuación se dan algunos ejemplos de ataques posibles contra NIDS basándonos en la fragmentación.

El ataque más básico que existe para evadir la detección de firmas por parte de un IDS consiste en fragmentar todo el tráfico en trozos muy pequeños asegurando que en cada datagrama individualmente no exista la cadena de bytes que identifica el ataque. Por supuesto, la utilidad de los NIDS se pondrían en duda si fueran vulnerables a un ataque tan básico. El analizador de red, debe ser capaz de reensamblar **correctamente** todos los fragmentos tal y como haría la máquina destino para poder analizar correctamente el flujo de datos.

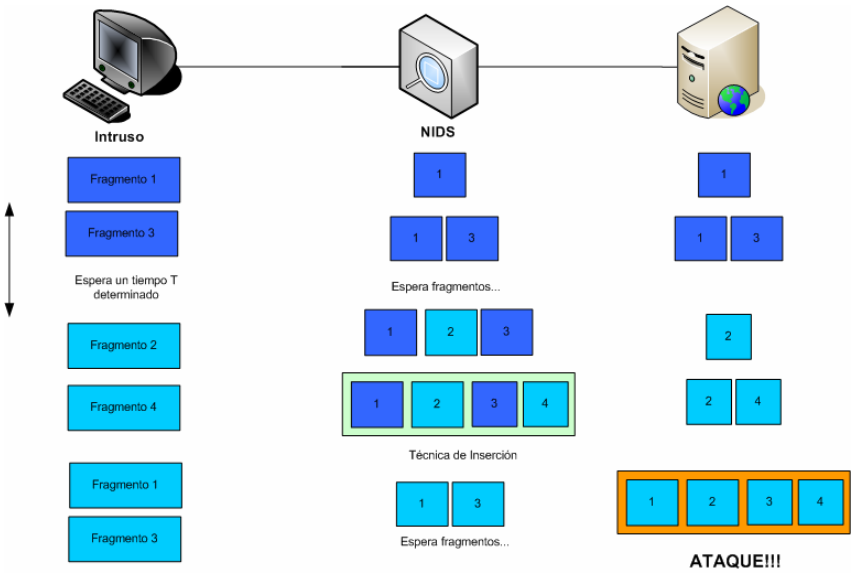
Para poder realizar el reensamblado el IDS (o cualquier máquina) debe esperar para recibir todos los fragmentos de un paquete. Un NIDS puede ser atacado, enviándole infinidad de fragmentos parciales, que nunca llegarán a completar un paquete. El IDS esperando recibir todas las partes, almacenará en memoria cada fragmento que reciba, lo cual se convertirá en un potencial ataque de denegación de servicio (DoS) por consumo de memoria.

5.4.1. - Timeout de fragmentación

Para evitar la situación anterior y que no se produzca un consumo de memoria, muchos sistemas desechan fragmentos basándose en sus tiempos de llegada. Un IDS eventualmente puede desechar fragmentos viejos e incompletos pero lo debe hacer de una forma consistente con las máquinas que vigila, o será vulnerable a ataques de inserción o evasión como veremos a continuación.



Imaginemos una red, donde el NIDS tiene un timeout de fragmentación más pequeño que el servidor que vigila. El intruso puede enviar una serie de fragmentos y esperar que el NIDS los deseche por tiempo límite de reensamblado, consiguiendo un ataque de evasión. Podemos encontrar también la situación contraria:

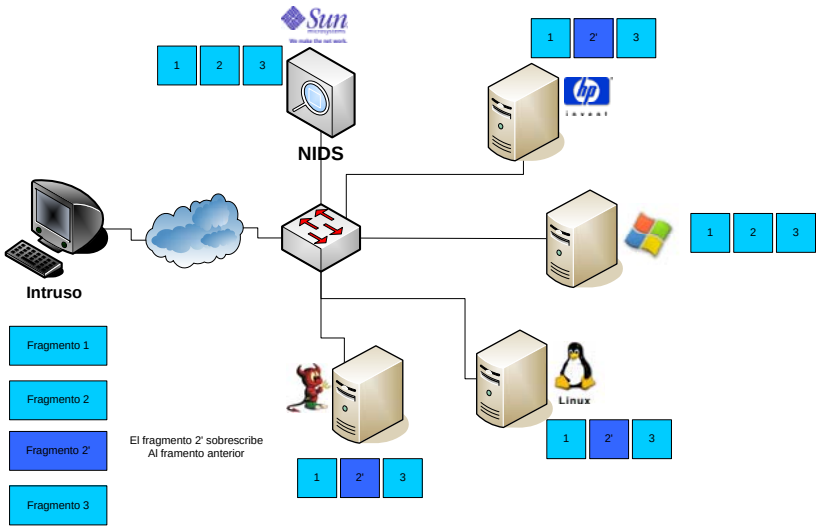


El tiempo de reensamblado en el NIDS es mayor que en la máquina objetivo, por lo que se pueden conseguir ataques de inserción esperando el tiempo exacto entre fragmentos para que el servidor deseche algunos fragmentos.

5.4.2. - Superposición de fragmentos

Es segundo problema y que puede presentar un mayor impacto en los IDS de red, es la superposición de fragmentos. Cuando a un sistema le llega un fragmento IP con datos que ya tenía, pueden ocurrir dos cosas según la implementación del algoritmo de reensamblado que tenga. Una posibilidad es que la máquina sobrescriba con los nuevos datos que acaba de recibir, la otra es que, deseche esos nuevos datos y conserve los antiguos. Como decíamos esto presenta un problema importante ya que si el NIDS no maneja la superposición de fragmentos de la misma forma que las máquinas que vigila podrá ser vulnerable a ataques de evasión.

La solución a este problema es bastante complicada, cada sistema manejará la superposición de paquetes de una forma, a veces a favor de los nuevos datos y otras prevaleciendo los datos antiguos y desechando los nuevos. Por ejemplo, un Windows NT resuelve la superposición de fragmentos manteniendo las datos antiguos (no sobrescribe en caso de superposición). Esto difiere de los BSD y UNIX que tal y como sugieren los estándares sobrescribe con los datos nuevos que recibe.



En este ejemplo de red heterogénea vemos que el NIDS (corriendo bajo un sistema SUN) después de reensamblar no ve el mismo flujo de datos que otras máquinas (Linux, FreeBSD y HP). El resultado final es que el reensamblado de fragmentos es diferente en el sistema final dependiendo del sistema operativo. A menos que el IDS conozca que OS esta corriendo en cada máquina y además conozca cómo maneja ese SO la superposición, técnicamente es vulnerable a ataques de superposición de fragmentos.

6. - Problemas de la capa de transporte

En el nivel de transporte encontramos quizás uno de los protocolos más importantes de Internet: TCP. Prácticamente todas las comunicaciones se realizan sobre conexiones TCP, por lo que es evidente que la gran mayoría de ataques se realizarán sobre este protocolo. Esto supone una serie de requerimientos para los sistemas de detección de intrusos a nivel de red. Los NIDS deben ser capaces de reconstruir toda la sesión TCP y analizar la información de envía a través de ella, tal y como la ven los sistemas que protege. Pero sino es capaz de ello, tal y como pasaba con el protocolo IP, el NIDS será vulnerable a cierto tipo de ataques.

6.1. - Cabeceras malformadas

Normalmente en los sistemas de detección de intrusos, los segmentos TCP son reensamblaos y extraídos sus datos sin verificar la mayoría de los campos de la cabecera. Esto hace peligrosamente sencillo realizar ataques de inserción contra el NIDS. El atacante podría insertar correctamente en un flujo de datos, segmentos con cabeceras malformadas, de forma que fuesen procesados por el NIDS pero descartados en la máquina destino. Por lo tanto es importante que un NIDS valide las cabeceras TCP antes de procesar sus datos.

Un buen lugar para empezar, es examinar el código fuente de la pila TCP/IP en algún sistema operativo, con el objetivo de encontrar todos los puntos donde un segmento TCP puede ser descartado o cuándo puede provocar la pérdida de conexión.

Acuse de recibo	Condición
No	Actual received packet too short
No	Bad checksum
No	Offset too Short (into TCP header) or too long
No	Actual received packet too short
RST	No listening process
RST	No listening process
No	Connection is in CLOSED state
No	Packet other than SYN received in LISTEN state
RST	ACK packet received in LISTEN state
No	Can't track new connections
No	Received RST packet in LISTEN state
RST	ACK packet received in LISTEN state
No	Any packet without SYN received in LISTEN state
No	Broadcast or Multicast SYN received
No	Out of resources
No	in pcbconnect() failure
No	Out of resources
No	ACK packet, bad sequence numbers
No	In SYN SENT state, received packet other than SYN
No	In SYN SENT state, received packet has bad CC.ECHO
No	In TIME WAIT state, packet has bad CC option
No	Any other packet received in TIME WAIT state
ACK	Bad timestamp (too old)
No	In T/TCP, no CC or bad CC on non-RST packet
RST	Listening user process has terminated
ACK	Packet is out of receive window
No	bit not set on non-SYN data packet
RST	ACK packet, bad sequence numbers
No	Duplicate ACK
ACK	ACK packet sent out of window
ACK	In TIME WAIT state, received ACK

Por supuesto estas condiciones variarán de un sistema operativo a otro.

Como se puede ver, un campo fácilmente modificable para que un segmento TCP sea descartado es el de las opciones binarias o *flags*. Cierta combinación de opciones en ciertas situaciones no serán validas y el paquete simplemente no se procesará.

Otro ejemplo de ataque típico es que, muchas implementaciones TCP no aceptarán datos en un paquete que no tiene bit ACK activo. De acuerdo con la especificación TCP, los sistemas deberían ser capaces de aceptar datos contenidos en un paquete SYN. La realidad es que al no ser una situación nada común, muchas implementaciones de la pila TCP/IP no tienen en cuenta este caso. Desde el punto de vista del NIDS, si descarta los datos de los paquetes SYN será vulnerable a un trivial ataque de evasión y si por el contrario, los tiene en cuenta pero las máquinas objetivo no lo implementan correctamente, el NIDS es vulnerable a ataque de inserción. Como vemos el gran problema suponen mantener la consistencia entre el analizador y los sistemas atacados.

Otro frecuente campo clave es el de checksum. Todas las implementaciones de TCP requieren comprobar la validez de los paquetes entrantes con el checksum de la cabecera. Pero sorprendentemente, muchos IDS no realizan esta comprobación, por lo que el sistema es vulnerable a la inserción de segmentos TCP con checksum erróneo, que por supuesto serán descartados en la máquina destino.

6.2. - Opciones TCP

Tal y como pasaba con el protocolo IP, es importante que un IDS procese y analice las opciones de la cabecera TCP correctamente. Desafortunadamente, el procesamiento de las opciones TCP es significativamente mas complicado que el procesamiento de las opciones IP, por lo que pueden haber más puntos por donde atacar a un NIDS. Una razón para esto es el hecho que muchas de las opciones TCP han sido creadas recientemente como puede ser el *TCP timestamps* and *Protection Against Wrapped Sequence Numbers (PAWS)* que permiten mejorar el control de flujo descartando paquetes duplicados o segmentos antiguos con un timestamp (que llevará cada paquete) superior a un cierto umbral. Consultar RFC1323.

Otra razón de la complejidad de analizar las opciones TCP es que existen restricciones concretas que hacen que algunas opciones sean inválidas en ciertos estados de conexión. Por ejemplo, algunas implementaciones TCP puede que rechacen paquetes no SYN con opciones que no se han especificado antes. Es importante que un IDS no acepta este tipo de paquetes de forma ciega, ya que seguramente serán rechazados en la máquina final y se producirá un ataque de inserción.

Por otro lado algunos sistemas puede que simplemente ignoren las opciones erróneas, pero continúen procesando el segmento. Si el IDS no conoce bien como se comportarán las máquinas que vigila será técnicamente posible realizar ataques de inserción y evasión.

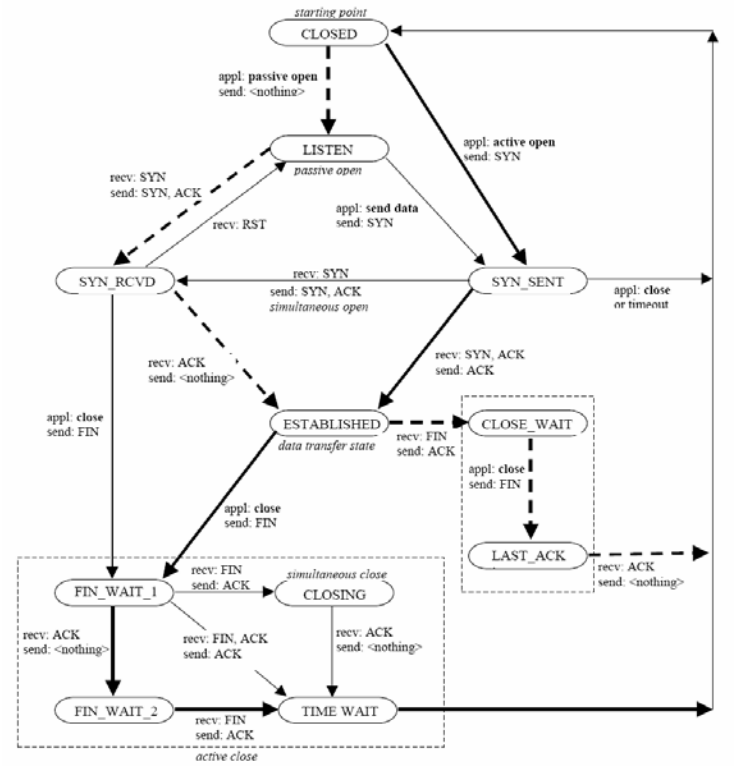
Vemos como aprovecharlos de las nuevas opciones implementadas en el RFC 1323: el PAWS o *Protection Against Wrapped Sequence Numbers* permite descartar segmentos duplicados, perdidos o muy antiguos que presentan un timestamp superior a cierto umbral. Un atacante puede trivialmente crear un segmento TCP, con un timestamp muy bajo, que causar que el módulo PAWS de la pila deseche el paquete sin procesarlo. El NIDS cuando analice el tráfico deberá estar seguro que los segmentos que procesa no serán descartados en la máquina final por culpa del PAWS. El IDS no solo necesita saber que el sistema final suporta PAWS necesita conocer además el umbral para los timestamp. Sin esta información, un IDS puede erróneamente procesar segmentos TCP inválidos, siendo vulnerable a ataques de inserción.

Otra forma de aprovecharse de esta nueva funcionalidad es que el umbral de timestamp normalmente depende del timestamp del último paquete procesado correctamente que llega. Un intruso puede maliciosamente generar un segmento con un tiemstamp adecuado para modificar el umbral y que los próximos fragmentos que lleguen sean descartados a pesar de ser completamente correctos. Esto es posible ya que en muchos casos e n la actualización de este umbral no se comprueba la validez del segmento (uso de números de secuencia no válidos).

6.3. - Sincronización de NIDS

Cada conexión TCP la podemos identificar por cuatro variables que la distinguen de cualquier otra conexión de la red existente en el mismo instante. Estas variables son la dirección IP origen del cliente, el puerto origen, la dirección IP destino del servidor y el puerto destino. Son los llamados parámetros de conexión y no podrán existir a la vez en la misma red dos conexiones con estos mismos parámetros.

Una sesión TCP va a poder estar en diferentes estados, en el siguiente grafo vemos como se realiza esta transición:



Para que un IDS pueda reconstruir la información de una sesión TCP, primero debe conocer de forma fiable el estado en el que se encuentra cada conexión. Y después debe llevar un seguimiento de los números de secuencia utilizados. Es decir, el IDS debe estar **sincronizado**. Cuando un IDS se encuentra desincronizado de una conexión, no puede reconstruir los datos que pasan a lo largo de ella y será trivialmente vulnerable a ataques de inserción y evasión. Por eso, uno de los principales objetivos de un intruso es desincronizar el NIDS de las conexiones.

Para mantener esta sincronización, el NIDS deberá almacenar cierta información de la conexión como sus parámetros de conexión, números de secuencia, la ventana, estado actual, etc. El bloque de memoria asignado a cada conexión se llamaba BCT que son las siglas de *"Bloque de control TCP"*. El NIDS mantendrá un BCT por cada conexión que vigila.

Las discusiones sobre los problemas de los NIDS en la capa de transporte se centran en 3 puntos dentro del procesamiento de una conexión: **La creación del BCT, El reensamblado y La destrucción del BCT.**

6.4. - Creación del BCT

Es un punto crítico. La forma que tiene un NIDS de detectar una nueva conexión condicionará en gran medida su precisión. En el momento de inicializar BCT se almacenan datos como los números de secuencia iniciales que el NIDS utilizará para mantenerse "sincronizado" y poder reensamblar la sesión. Si un intruso consigue forzar la creación de un falso BCT con falsos números de secuencia, conseguirá que las siguientes conexiones con los mismos parámetros (direcciones y puertos) estén desincronizadas con el NIDS, ya que detectará que el SEQ de los paquetes es erróneo.

Existen diversas aproximaciones para detectar una nueva conexión, pero prácticamente todas presentan inconvenientes que un atacante puede vulnerar. La primera opción y más obvia consiste en crear el BCT cuando el NIDS detecte el saludo a tres vías TCP. El inconveniente claro que sin saludo, analizará cada paquete

individualmente y no como un flujo TCP. Cosa conseguirá, si el atacante se asegura que el NIDS no vea el saludo utilizando alguna de las técnicas de evasión antes comentadas. Otro problema es la creación de falsos BCT forzados por el atacante, de desincronizará cualquier otra conexión con los mismos parámetros (ejemplo del párrafo anterior). A priori en una red donde se controle el spoofing de IP será una tarea realmente complicada falsear las respuestas del servidor para engañar al NIDS.

Existen otras posibilidades que consisten, por ejemplo, en crear el BCT cuando detecten paquetes SYN o obtener el estado de la conexión y los números de secuencia analizando el tráfico existente.

6.5. - Reensamblado

Como ya se ha comentado en varias ocasiones, para conseguir que el reensamblado del flujo TCP sea correcto es necesario un seguimiento fiable de los números de secuencia. Veamos algunos puntos débiles donde romper esa sincronización y evadir el análisis de patrones.

Un problema inherente de los monitores pasivos, es que si pierden paquetes no pueden solicitar una retransmisión como cualquiera de las máquinas implicadas en la comunicación. Por tanto un intruso puede provocar de algún modo intencionado la pérdida de paquetes, por ejemplo gracias a ataques de DoS o paquetes que llegan fueran de orden y el NIDS no procesa adecuadamente. Las pérdidas de paquetes provocan una desincronización del NIDS y por consiguiente evita que el reensamblado del flujo TCP sea posible.

Otra clase de problemas hacen referencia al tamaño de la ventana de la comunicación TCP, un NIDS normalmente no analizará si los datos segmentos que procesa se encuentran dentro de la ventana indicada por la máquina destino. Un NIDS que no realiza esta comprobación será vulnerable a ataques de inserción ya que estos paquetes que superan el tamaño de la ventana serán descartados en la máquina destino.

Al igual que ocurría con el reensamblado de fragmentos IP, en TCP también es posible que ocurra una superposición de segmentos. Un intruso intencionadamente puede generar dos segmentos TCP exactamente iguales salvo en el campo de datos (y por supuesto en el checksum).A la máquina objetivo le llegarán los dos, pero solo procesará uno dependiendo de su implementación (coger el más antiguo o el más nuevo), el NIDS sin conocer el SO de la máquina atacada, solo con el tráfico de red no puede saber cual de los dos segmentos ha procesado. Y si no los procesa exactamente igual, es susceptible de ataques de inserción o evasión.

La segunda posibilidad es el envío de segmentos TCP desordenados con un tamaño tal que al reensamblarlos se sobrescriban ciertos segmentos.

En la siguiente tabla podemos ver como manejan los diferentes sistemas operativos la superposición de segmentos TCP:

Irix 5.3	Procesa los datos nuevos
HP-UX 9.01	Procesa los datos nuevos
Linux	Procesa los datos nuevos
AIX 3.25	Procesa los datos nuevos
Solaris 2.6	Procesa los datos nuevos
FreeBSD 2.2	Procesa los datos nuevos
Windows NT	Procesa los datos antiguos

Una posibilidad que tiene el IDS para saber si un paquete es procesado en la maquina destino, es esperar hasta que escuche el ACK de la máquina destino, que indicará que el paquete se ha procesado correctamente. Pero no será posible en todos los casos ya que el número ACK es acumulativo y no se realiza el asentamiento de todos y cada uno de los segmentos.

6.6. - Destrucción del BCT

Evidentemente es imposible mantener eternamente los BTC en memoria, por lo que el NIDS tendrá algún mecanismo para desecharlos. Mecanismo que intentará vulnerar un atacante.

Un intruso puede hacer creer al NIDS que la conexión con la máquina objetivo ha terminado, entonces el NIDS al cabo de un tiempo o inmediatamente desechará el BTC de memoria, y el intruso podrá seguir atacando sin miedo a que el NIDS reensamble los paquetes. Por lo tanto el NIDS debe conocer exactamente cuando se cierra una conexión y evitar ser engañado.

En TCP una conexión se finaliza con dos formas diferentes: utilizando paquetes FIN o paquetes RST. Cuando un sistema envía un segmento FIN indica que ha terminado de enviar datos y esta preparado para cerrar la conexión. Los mensajes FIN se responden con un ACK y cada extremo de la conexión envía deberá enviar un segmento FIN para terminar la comunicación. A priori, en un entorno sin posibilidad de spoofing de IP, es suficientemente fiable que el NIDS confíe en la desconexión FIN para determinar que realmente una conexión ha terminado.

No ocurre lo mismo con los paquetes RST. Éste flag indica que ha habido algún error y la conexiones debe cerrarse inmediatamente. Como el segmento RST no tiene acuse de recibo, el NIDS no puede saber si el servidor que vigila le ha llegado el paquete y ha cerrado la conexión realmente. Un atacante puede generar un segmento RST válido (SEQ ok) pero que no será procesado por la máquina objetivo. De esta forma el NIDS cree que ha finalizado la conexión.

El único método que tiene el IDS para verificar que se ha cerrado la conexión es escuchando durante un periodo de tiempo si se transmiten datos, si no es así se desecha el BTC. Pero esto implica potencialmente un nuevo tipo de ataque contra el NIDS. Imaginemos la siguiente situación: El atacante inicia una conexión normalmente, inmediatamente después genera un segmento RST que no llega a la máquina objetivo pero que si procesa el NIDS. El atacante inicia otra conexión con los mismos parámetros pero con números de secuencia diferentes. ¿Como va a actuar el NIDS? Generando un nuevo BTC y eliminando el antiguo?. O Desecha la nueva conexión porque los segmentos tienen un número de secuencia diferente al que espera?. En un caso como este cada IDS actuará de un modo, cosa que el intruso puede intentar explotar.

7. - Ataques en otras capas

7.1. - Nivel de enlace

Existen las mismas implicaciones comentadas durante todo el documento para los ataques de inserción en la campa de enlace. Un atacante en la misma LAN que el monitor de red puede directamente enviar paquetes de la capa de enlace del IDS, sin permitir que el host especificado como IP destino vea el paquete (Ya que el direccionamiento se hace a nivel MAC). Si el atacante conoce la MAC del NIDS, el atacante utilizando la técnica del ARP-spoofing, simplemente debe falsificar el paquete con la dirección del NIDS en la MAC destino, y la dirección IP de la máquina atacada. Ningún otro sistema de la LAN procesará el paquete aunque . Y si el NIDS no comprueba la correspondencia MAC-IP es correcta, el monitor de red es vulnerable a ataques de inserción.

De nuevo, a menos que el IDS compruebe la dirección de destino en la cabecera contrastándola con la cabecera de la capa de enlace (lo cual es normal que haga),podrá ser vulnerable a este tipo de ataques.

7.2. - Nivel de aplicación

Todos los ataques que sea han comentado hasta ahora aprovechan inconsistencias del entorno y manipulan el tráfico de red, por supuesto no son las únicas técnicas existentes a la hora de evadir la detección de patrones de un IDS. Vamos a ver algunas técnicas sobre el protocolo HTTP (RFC2616) para evitar que un IDS pueda detectar ataques Web.

Evasión de IDS: Atacar sin ser detectado

Técnica	Ejemplo	Comentarios
Cambio de método	GET /cgi-bin/some.cgi → HEAD /cgi-bin/some.cgi	En muchos casos para explotar un CGI puede ser posible cambiar el método utilizado y evadir la detección del ataque.
Codificación de la URL	cgbin → %63%67%69%2d%62%69%6e	Técnica clásica que todos los IDS tienen en cuenta. El http permite la codificación de los caracteres de la URL en formato hexadecimal.
Transformación UNICODE	0x5C 0xC19C → U+005C / 0xE0819C	En el ejemplo todos esos valores en hexadecimal al ser traducidos a Unicode representan el mismo valor, la barra /. ¿Cuántas firmas tendría que tener el IDS?
Directorio 1	GET ////cgi-bin///vuln.cgi	
Directorio 2	GET /cgi-bin/blahblah/../some.cgi HTTP/1.0	
Directorio 3	GET /cgi-bin/./././.vuln.cgi	
Múltiples peticiones	GET / HTTP/1.0\r\n Header: /../../cgi-bin/some.cgi HTTP/1.0\r\n \r\n	Un IDS es posible que solo analice la primera línea de la petición, hasta el HTTP/1.x
Evitar la detección de parámetros.	GET /index.htm%3fparam=/../cgi-bin/some.cgi HTTP/1.0 GET /index.htm?param=/../cgi-bin/some.cgi HTTP/1.0	Muchos IDS buscan patrones en los parámetros de las aplicaciones de ataque. Se intenta evitar que el IDS situe los parámetros dentro de la petición http.
Directorios largos	GET /rfprfp**[caracteres]**rfprfp/../cgi-bin/some.cgi HTTP/1.0	
Caracteres NULL	GET%00 /cgi-bin/some.cgi HTTP/1.0	
Sensible a mayúsculas	/CGI-BIN/SOME.CGI	Al atacar máquinas Windows es posible utilizar mayúsculas para evadir detecciones de patrones básicas.

Todos esto ataques son bien conocidos y prácticamente todos los sistemas de detección deberían manejarlos correctamente. Por lo que, en muchas ocasiones, el intruso intentará combinar varias técnicas o modificar las ya existentes.

8. - Bibliografía

- "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection". Thomas Ptacek, Timothy Newsham,. Secure Networks, 1998.
- "Evading/Attacking NIDS" Priyank Porwal
- "Bypassing Intrusion Detection Systems" Ron Gula, Founder Network Security Wizards.
- "The Tao of Network Intrusion Detection", Gianni Tedesco.
- Evading NIDS, revisited. (http://www.securityfocus.com/infocus/1852)
- IDS Evasion Techniques and Tactics (http://www.securityfocus.com/infocus/1577).
- IDS Evasion with Unicode (http://www.securityfocus.com/infocus/1232).
- Resynchronizing NIDS Systems (http://www.securityfocus.com/infocus/1226).
- PAWS Attacks (http://www.kb.cert.org/vuls/id/637934)
- Whisker's anti-ids tactics (http://www.wiretrip.net/rfp/txt/whiskerids.htm).
- www.wikipedia.org
- www.google.es