Elastix
Certified Engineer

Protocolos SIP/IAX/RTP

Laboratorios

Lab 6

Protocolos SIP / IAX/ RTP

Elastix ECT Training

Protocolos SIP / IAX/ RTP

Laboratorio 6.1

Descripción: Realizar la captura de una conversación utilizando la herramienta tcpdump.

Objetivo: Capturar una conversación realizada en nuestro servidor Elastix.

Tiempo Máximo: 15 minutos.

Instrucciones:

Tcpdump es una herramienta que permite capturar paquetes recibidos y transmitidos, en tiempo real, en la red en que se encuentra nuestro servidor. Más información sobre esta herramienta en: http://www.tcpdump.org/

Tcpdumb se ejecuta en la línea de comandos y se encuentra incluido en Elastix.

- Ingresamos a la consola como usuario root, y crearemos un directorio para guardar las capturas que realicemos en el servidor.

```
[root@elastix ~]# mkdir pruebaws
```

- Realizaremos la captura desde el directorio `pruebaws`

```
[root@elastix ~]# cd pruebaws
```

- Para realizar la captura utilizaremos el comando:

```
tcpdump –i any –s0 -w capturaXX.pcap
```

- Dónde:
- -i any - significa que vamos a tomar capturas en todas las interfaces de red.
- -s0 - significa que no voy a poner un límite en el tamaño de cada paquete a capturar.
- -w capturaXX.pcap - por un lado la opción “-w” indica que los paquetes capturados van a ser almacenados en un archivo “capturaXX.pcap”. El archivo puede tener un nombre arbitrario, solo que hay que tener cuidado porque la aplicación puede sobreescribir el archivo, por eso podemos usar XX como el número de captura: 01, 02, 03 …. etc. La extensión “.pcap” en el nombre del archivo.
- Una vez que ejecutemos el comando anterior, el equipo capturará paquetes de todas las interfaces de red. Inmediatamente podemos realizar una llamada desde una de las extensiones creadas (ej: 201 a 202).
- Una vez que terminemos de hablar y cerremos la extensión, presionamos “CTRL + C”.
- Ahora copiaremos el archivo capturaXX.pcap para cargarlo posteriormente en wireshark.

Laboratorio 6.2

Descripción: En esta práctica utilizaremos la herramienta wireshark para analizar la conversación capturada en el laboratorio anterior.

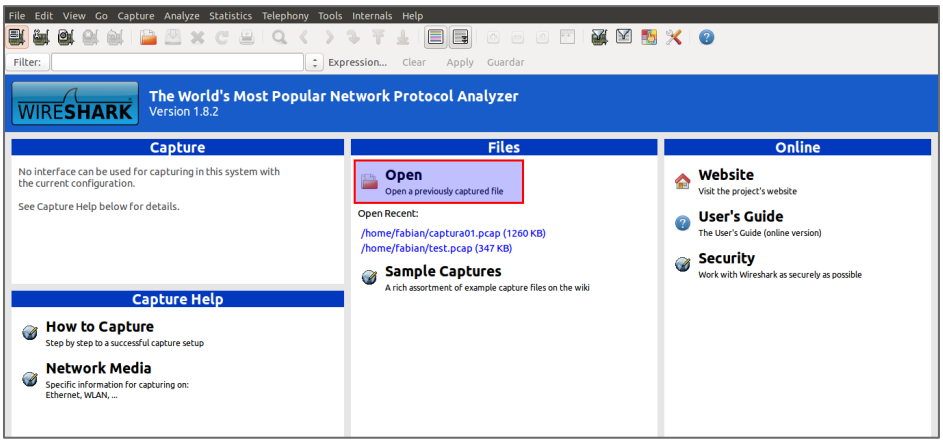
Objetivo: Adiestrar al estudiante en el uso de sniffers para analizar tráfico RTP y generar y reproducir el payload de una captura de paquetes.

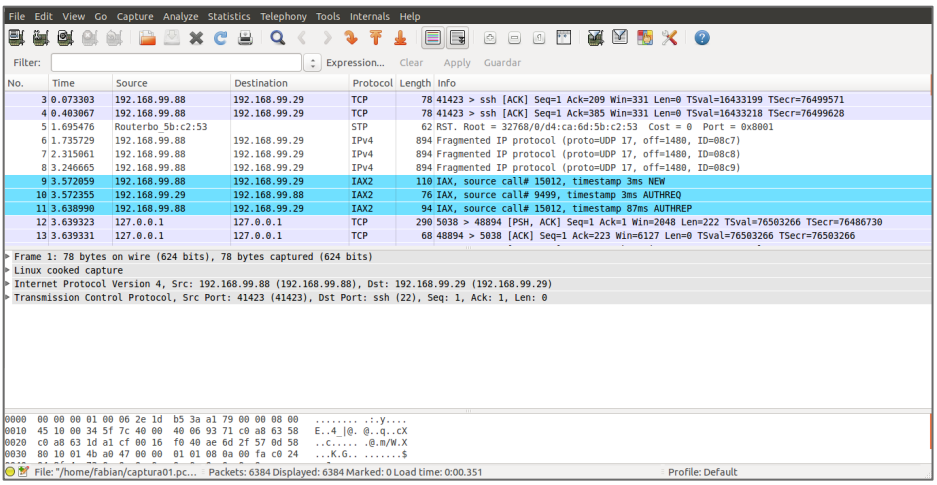
Tiempo Máximo: 10 minutos.

Instrucciones:

Wireshark es un programa open source desarrollado para el análisis de protocolos. Es similar en funcionalidad a tcpdump, con la ventaja de que cuenta con una interfaz gráfica y otras características adicionales que permiten un mejor análisis. Más información en: http://www.wireshark.org/

- Primero iniciamos Wireshark y hacemos clic en “Open a previously captured file” e importamos el archivo de captura de la práctica anterior.

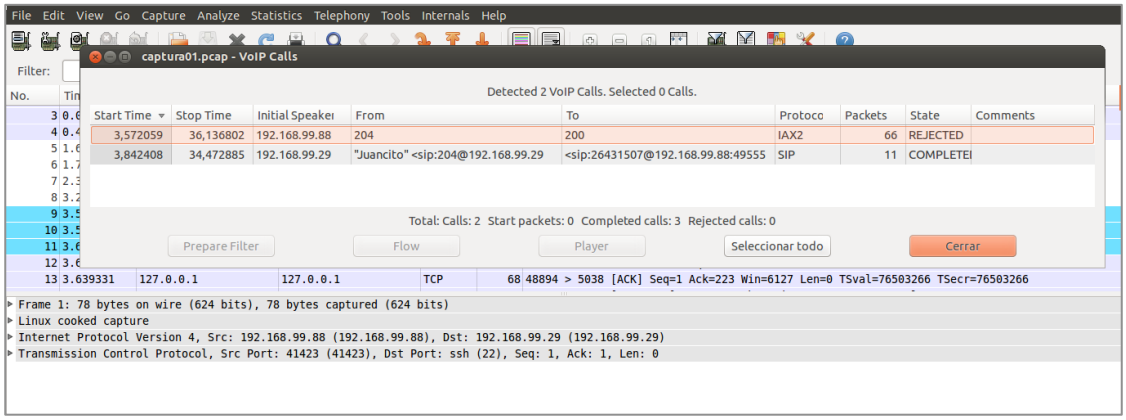




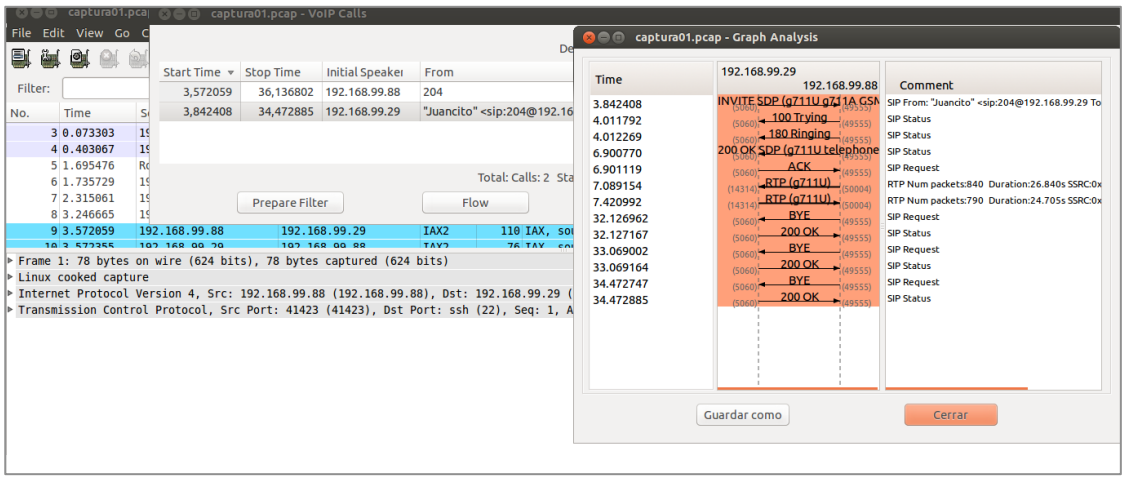
Nota: Para obtener el archivo facilmente podemos usar un cliente SCP, como: WINSCP, Filezilla, Fugu (Mac)

- En la captura podemos observas diferentes tipos de paquetes, sin embargo nos interesa analizar SIP y RTP.

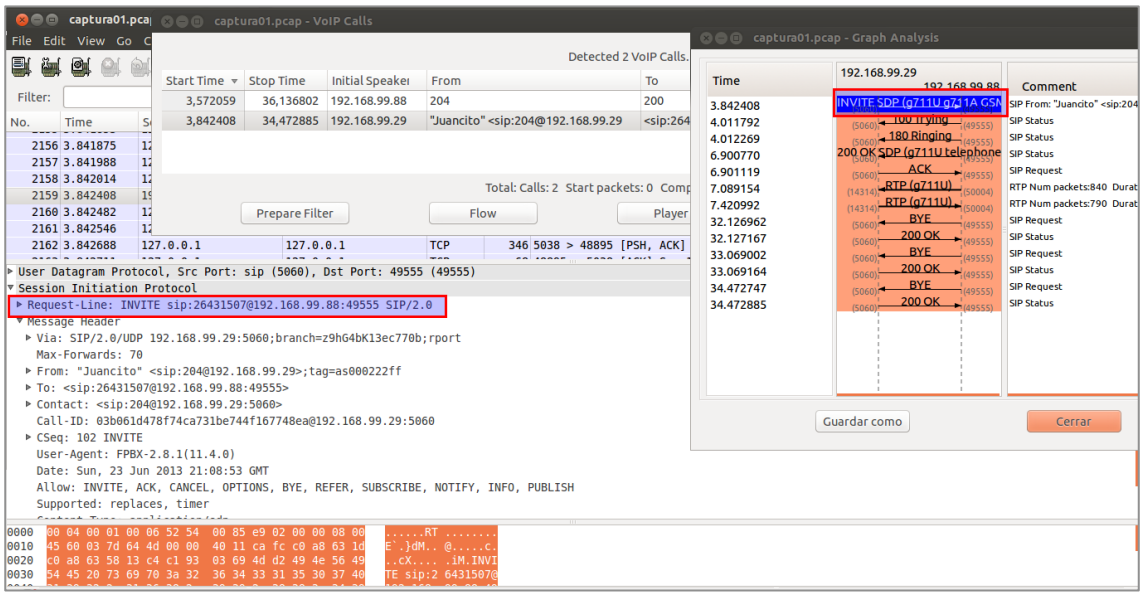
- Vamos a la pestaña “Telephony” → “VoIP Calls”. Allí obtendremos un listado de las llamadas VoIP capturadas con tcpdump, al igual que sus resultados (COMPLETE, REJECTED, CANCEL). Es muy útil si queremos una vista rápida para buscar alguna anomalía en el funcionamiento del sistema, como un proveedor SIP que nos esté rechazando llamadas o problemas comunes como el one-way-audio.

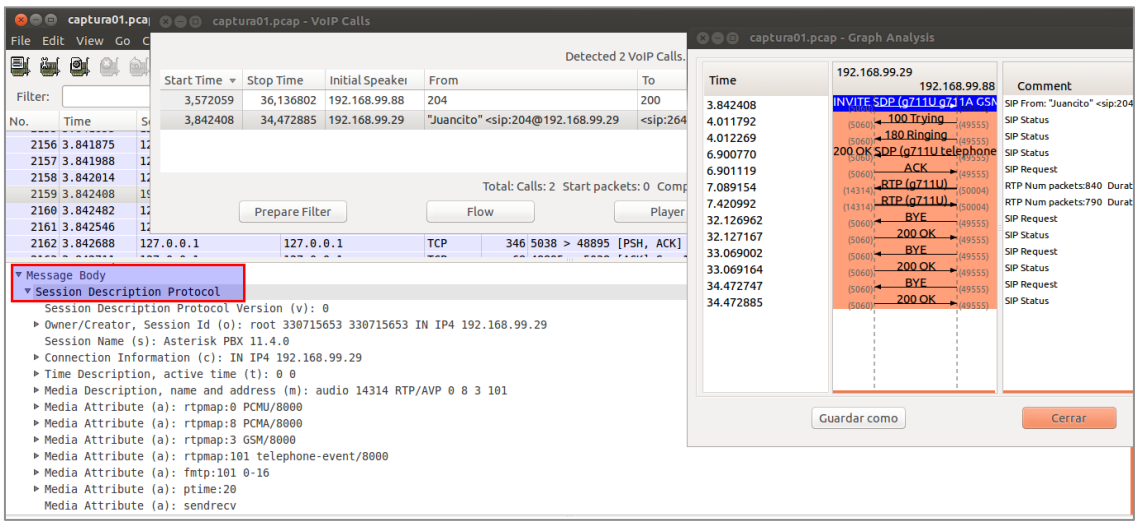


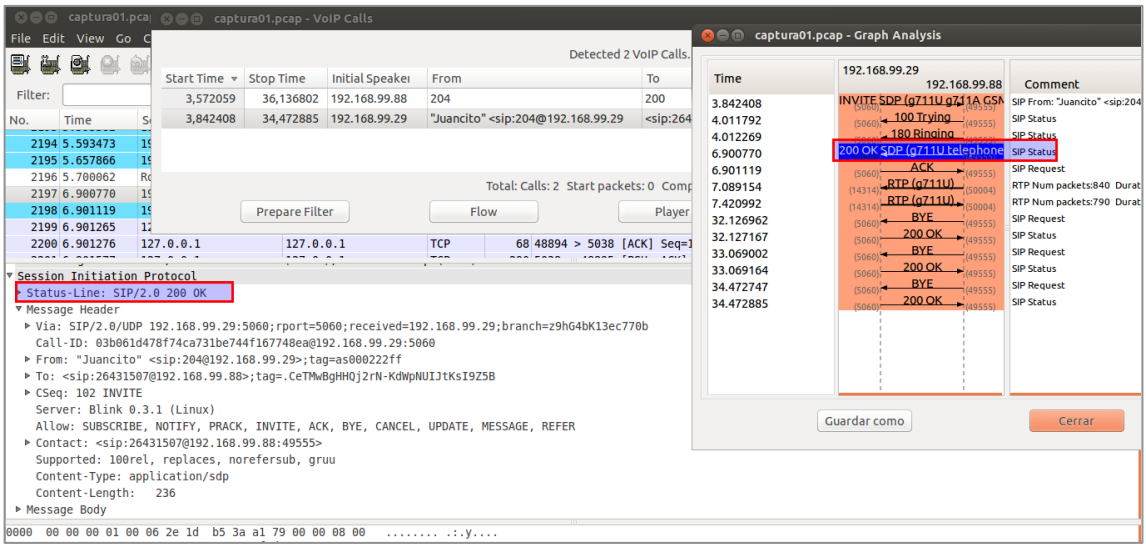
- Podemos seleccionar cualquiera de las llamadas (inclusive más de una) e indicarle a Wireshark que realice un diagrama (Flow) de cada paquete intercambiado entre las partes.



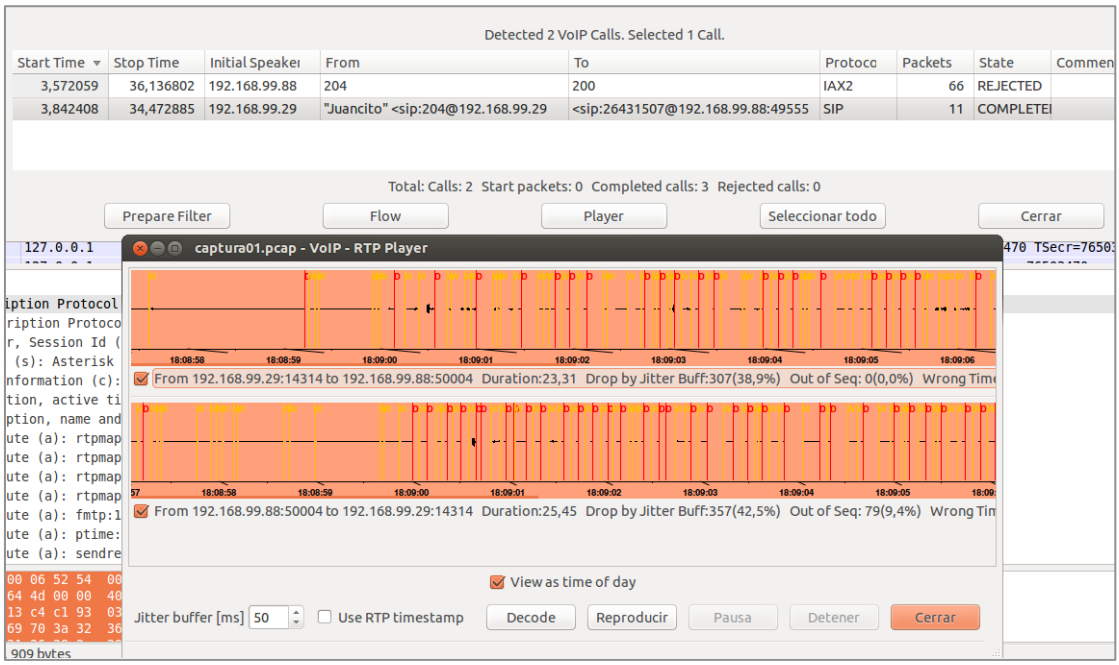
- Como podemos observar, tenemos el diálogo completo entre Elastix y un SIP peer. Si seleccionamos cada paquete podemos analizarlo en detalle, lo que permite hacer un depuramiento intensivo fácilmente. Podemos visualizar los valores de los campos del header SIP e inclusive el SDP como payload del SIP en mensajes como INVITE y 200 OK.


captura01.pcap - VoIP Calls
Detected 2 VoIP Calls.
Start Time | Stop Time | Initial Speaker | From | To
3,572059 | 36,136802 | 192.168.99.88 | 204 | 200
3,842408 | 34,472885 | 192.168.99.29 | "Juancito" <sip:204@192.168.99.29 | <sip:264
Total: Calls: 2 Start packets: 0
Prepare Filter
Flow
Player
2162 3.842688 127.0.0.1 127.0.0.1 TCP 346 5038 > 48895 [PSH, ACK]
User Datagram Protocol, Src Port: sip (5060), Dst Port: 49555 (49555)
Session Initiation Protocol
Request-Line: INVITE sip:26431507@192.168.99.88:49555 SIP/2.0
Message Header
Via: SIP/2.0/UDP 192.168.99.29:5060;branch=z9hG4bK13ec770b;rport
Max-Forwards: 70
From: "Juancito" <sip:204@192.168.99.29>;tag=as000222ff
To: <sip:26431507@192.168.99.88:49555>
Contact: <sip:204@192.168.99.29:5060>
Call-ID: 03b061d478f74ca731be744f167748ea@192.168.99.29:5060
CSeq: 102 INVITE
User-Agent: FPBX-2.8.1(11.4.0)
Date: Sun, 23 Jun 2013 21:08:53 GMT
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH
Supported: replaces, timer
captura01.pcap - Graph Analysis
Time
192.168.99.29
192.168.99.88
Comment
INVITE SDP (g711U g711A GSM
100 Trying
180 Ringing
200 OK SDP (g711U telephone
ACK
RTP (g711U)
BYE
200 OK
Guardar como
Cerrar


captura01.pcap - VoIP Calls
Detected 2 VoIP Calls.
Prepare Filter
Flow
Player
Message Body
Session Description Protocol
Session Description Protocol Version (v): 0
Owner/Creator, Session Id (o): root 330715653 330715653 IN IP4 192.168.99.29
Session Name (s): Asterisk PBX 11.4.0
Connection Information (c): IN IP4 192.168.99.29
Time Description, active time (t): 0 0
Media Description, name and address (m): audio 14314 RTP/AVP 0 8 3 101
Media Attribute (a): rtpmap:0 PCMU/8000
Media Attribute (a): rtpmap:8 PCMA/8000
Media Attribute (a): rtpmap:3 GSM/8000
Media Attribute (a): rtpmap:101 telephone-event/8000
Media Attribute (a): fmtp:101 0-16
Media Attribute (a): ptime:20
Media Attribute (a): sendrecv
captura01.pcap - Graph Analysis
Guardar como
Cerrar

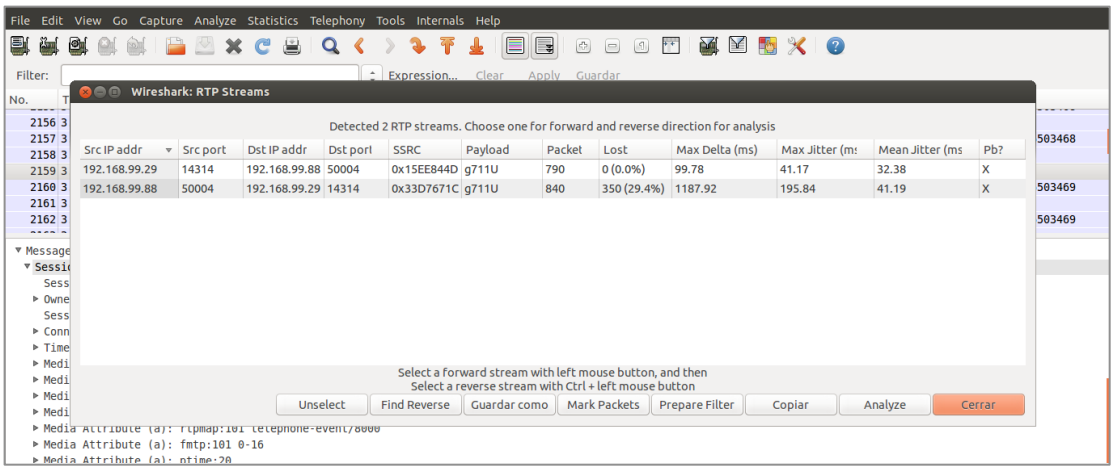

captura01.pcap - VoIP Calls
Detected 2 VoIP Calls.
Prepare Filter
Flow
Player
2194 5.593473
2195 5.657866
2196 5.700062
2197 6.900770
2198 6.901119
2199 6.901265
2200 6.901276 127.0.0.1 127.0.0.1 TCP 68 48894 > 5038 [ACK] Seq=1
Session Initiation Protocol
Status-Line: SIP/2.0 200 OK
Message Header
Via: SIP/2.0/UDP 192.168.99.29:5060;rport=5060;received=192.168.99.29;branch=z9hG4bK13ec770b
Call-ID: 03b061d478f74ca731be744f167748ea@192.168.99.29:5060
From: "Juancito" <sip:204@192.168.99.29>;tag=as000222ff
To: <sip:26431507@192.168.99.88>;tag=.CeTMwBgHHQj2rN-KdWpNUIJtKsI9Z5B
CSeq: 102 INVITE
Server: Blink 0.3.1 (Linux)
Allow: SUBSCRIBE, NOTIFY, PRACK, INVITE, ACK, BYE, CANCEL, UPDATE, MESSAGE, REFER
Contact: <sip:26431507@192.168.99.88:49555>
Supported: 100rel, replaces, norefersub, gruu
Content-Type: application/sdp
Content-Length: 236
Message Body
captura01.pcap - Graph Analysis
Guardar como
Cerrar

- Wireshark también puede reproducir un audio con los paquetes RTP de la llamada seleccionada. Cerramos el diagrama, y hacemos clic "Player" en la llamada seleccionada y luego en “Decode”.

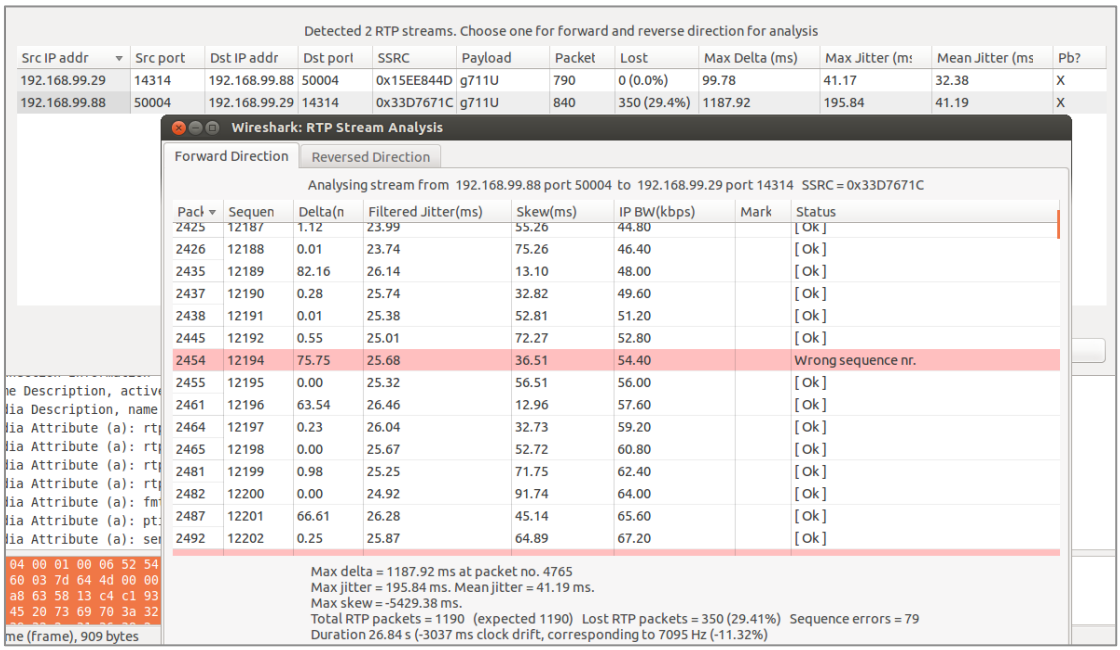


- Como podemos observar, tenemos el audio completo, tanto el upstream y downstream que pasó por la interfaz de red de nuestro Elastix. Podemos analizar el flujo y hasta reproducir el mismo. Esto nos permite, por ejemplo, determinar que un problema de corte de llamadas no es por causa de Elastix.

- También podemos hacer un análisis sobre el flujo de RTP de cada llamada capturada. Vamos a “Telephony → RTP → Show All Streams” y allí tenemos todos los flujos de RTP capturados por tcpdump. Los mismos podemos diferenciarlos por su IP y SSRC.



- Como podemos observar en la imagen, tenemos datos fundamentales acerca de la calidad de cada stream, como el porcentaje de paquetes perdidos, el diferencial de jitter, el máximo valor de jitter presente, etc. Además podemos analizar el stream paquete por paquete, para ello debemos seleccionar un stream y elegir “Analyze”.



Como podemos observar, en este punto también podemos escuchar el Stream o descargarlo como un archivo de audio.

Laboratorio 6.3

Descripción: En este laboratorio configuraremos el uso de TLS (Seguridad en la Capa de Transporte) y SRTP (Encripción de la Voz) utilizando el softphone blink.

Objetivo: Configurar encriptación de audio en Elastix.

Tiempo Máximo: 40 minutos.

Instrucciones:

- Ingresamos a nuestra consola y nos colocamos en el directorio /etc/asterisk, una vez ahí debemos crear una carpeta llamadas keys. Para esto ejecutamos el siguiente comando:

  ```
  mkdir keys
  ```

- Luego nos dirigimos al siguiente directorio:

  ```
  /usr/share/doc/asterisk-x.x.x/contrib/scripts/
  ```

> **Nota:** el directorio asterisk-x.x.x corresponde a la versión de Asterisk actual. Para determinar con exactitud la versión podemos ejecutar el siguiente comando desde consola: `asterisk -vvvr`, ejemplo:
>
> ```
> [root@ecte ~]# asterisk -vvvr
> Asterisk 1.8.20.0, Copyright (C) 1999 - 2012 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 1.8.20.0 currently running on ecte (pid = 2903)
> Verbosity is at least 3
> ecte*CLI>
> ```
>
> Directorio: `/usr/share/doc/asterisk-1.8.20.0/contrib/scripts/`

- Una vez que nos encontramos en el directorio, ejecutamos el siguiente comando para generar los certificados que serán utilizados por asterisk:

  ```
  ./ast_tls_cert -C pbx.mycompany.com -O "mycompany" -d /etc/asterisk/keys
  ```

> **Nota:** En este caso, pbx.mycompany.com es el hostname de nuestro servidor ("mycompany" es un comentario). Podemos verificar el hostname ingresando al archivo **/etc/sysconfig/network**. Es necesario además adicionar la siguente línea al final del archivo **/etc/hosts**: `IP_del_servidor server server.example.com`, donde server.example.com es el hostname, Ej: IP = 192.168.5.93, hostname = test.elastix.com -> `192.168.5.93 test test.elastix.com`.

- Durante el proceso nos pedirá ingresar una clave y su validación un par de ocasiones, la escribimos y presionamos enter para continuar. Esta clave es la contraseña que utilizaran nuestros certificados de asterisk.

- Ahora, desde el mismo directorio, ejecutamos el siguiente comando para generar los certificados que serán utilizados por nuestros teléfonos:

```
./ast_tls_cert -m client -c /etc/asterisk/keys/ca.crt -k
/etc/asterisk/keys/ca.key -C phone.mycompany.com -O "mycompany" -d
/etc/asterisk/keys -o phone
```

Nota: Este comando crea certificados para los teléfonos que lo requieran, sin embargo hay muchos teléfonos que traen un certificado self signing. Tomando el ejemplo de hostname anterior (test.elastix.com), podemos reemplazar phone.mycompany.com por phone.elastix.com.

- Nos solicitará una clave, es la misma que usamos para Asterisk.

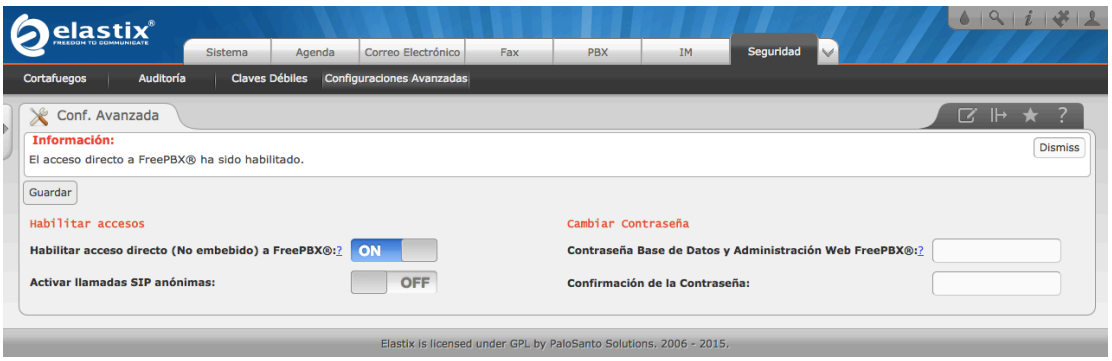
- Nos dirigimos al directorio /etc/asterisk/keys y validamos que los archivos estén generados:

```
[root@ecte keys]# ll
total 48
-rw------- 1 root root 1212 Jun  2 13:09 asterisk.crt
-rw------- 1 root root  574 Jun  2 13:09 asterisk.csr
-rw------- 1 root root  887 Jun  2 13:09 asterisk.key
-rw------- 1 root root 2099 Jun  2 13:09 asterisk.pem
-rw------- 1 root root  155 Jun  2 13:08 ca.cfg
-rw------- 1 root root 1753 Jun  2 13:09 ca.crt
-rw------- 1 root root 3311 Jun  2 13:09 ca.key
-rw------- 1 root root 1216 Jun  2 13:10 phone.crt
-rw------- 1 root root  578 Jun  2 13:10 phone.csr
-rw------- 1 root root  887 Jun  2 13:10 phone.key
-rw------- 1 root root 2103 Jun  2 13:10 phone.pem
-rw------- 1 root root  124 Jun  2 13:10 tmp.cfg
```

- Es necesario cambiar el propietario y el grupo de archivos a Asterisk, y habilitar todos los permisos

```
[root@test keys]# ll
total 48
-rwxrwxrwx 1 asterisk asterisk 1196 Feb  9 15:34 asterisk.crt
-rwxrwxrwx 1 asterisk asterisk  566 Feb  9 15:33 asterisk.csr
-rwxrwxrwx 1 asterisk asterisk  891 Feb  9 15:33 asterisk.key
-rwxrwxrwx 1 asterisk asterisk 2087 Feb  9 15:34 asterisk.pem
-rwxrwxrwx 1 asterisk asterisk  150 Feb  9 15:32 ca.cfg
-rwxrwxrwx 1 asterisk asterisk 1736 Feb  9 15:33 ca.crt
-rwxrwxrwx 1 asterisk asterisk 3311 Feb  9 15:33 ca.key
-rwxrwxrwx 1 asterisk asterisk 1204 Feb  9 15:40 phone.crt
-rwxrwxrwx 1 asterisk asterisk  570 Feb  9 15:40 phone.csr
-rwxrwxrwx 1 asterisk asterisk  887 Feb  9 15:40 phone.key
-rwxrwxrwx 1 asterisk asterisk 2091 Feb  9 15:40 phone.pem
-rwxrwxrwx 1 asterisk asterisk  120 Feb  9 15:40 tmp.cfg
```

- Ahora activamos TLS en Elastix. Para realizar esta tarea es necesario entrar a FreePBX, para lo cual activamos su acceso en la interfaz de administración de Elastix en "Seguridad → Configuraciones Avanzadas":



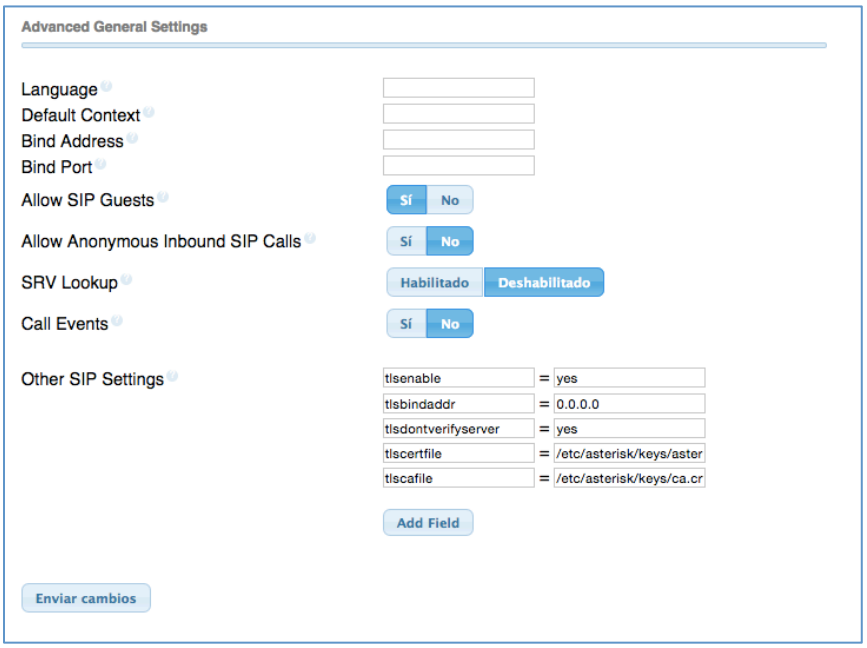
- Una vez que activamos FreePBX ingresamos a la interfaz en el siguiente enlace: https://ip-de-su-servidor/admin, el usuario es admin y la contraseña es la que configuramos al instalar Elastix.

- Al ingresar vamos a la siguiente ruta "Settings → Asterisk SIP Settings".

- Primero configuramos la sección "NAT Settings" y seleccionamos la opción "Public IP".

- Luego en la sección "Advanced General Settings" ingresamos los siguientes parámetros:

tlsenable=yes
tlsbindaddr=0.0.0.0
tlsdontverifyserver=yes
tlscertfile=/etc/asterisk/keys/asterisk.pem
tlscafile=/etc/asterisk/keys/ca.crt



Una vez que hemos ingresado los datos hacemos clic en “Enviar cambios” y luego aplicaciomos los cambios.

Nota: Si no aplicamos los cambios, los campos que hemos agregado no se incluirán en el archivo /etc/asterisk/sip_general_additional.conf

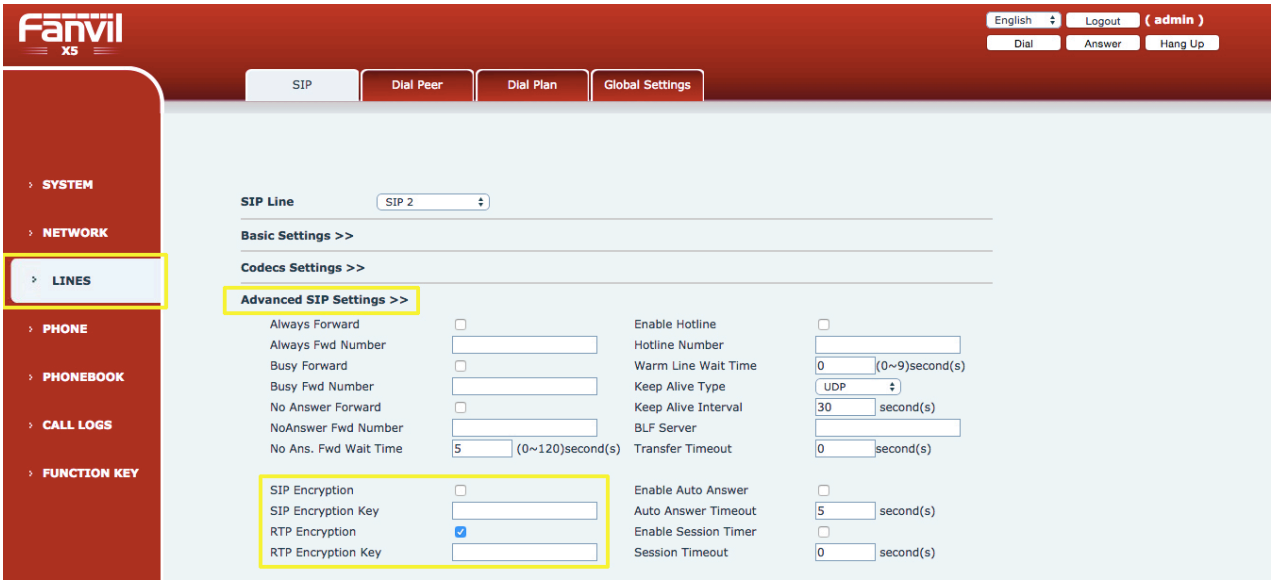
Encriptación con SRTP

- Ahora, en la interfaz de administración, vamos a PBX → Configuración PBX → Extensiones. Ingresamos a cualquiera de las extensiones y en la sección Device Options configuramos el campo “encryption” seleccionando “SRTP only”:

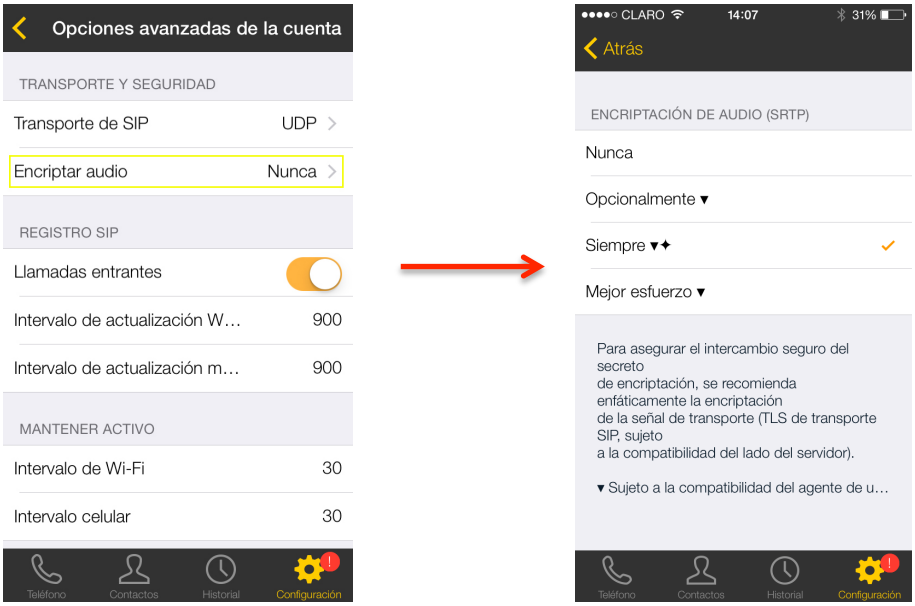
- Para esta práctica usaremos cualquiera de las extensiones que hemos configurado anteriormente.

- Es importante mencionar que la habilitación de SRTP no es un standard en todos los endpoints. Cada endpoint puede tener una manera diferente de habilitar esta funcionalidad y es importante contactar al fabricante para verificar primero que el endpoint soporta SRTP y TLS y además como se configura.

- En el caso de un teléfono fanvil X5 la configuración se realiza en la sección advanced de la cuenta SIP en el campo “RTP Encryption”, el cual se debe habilitar.

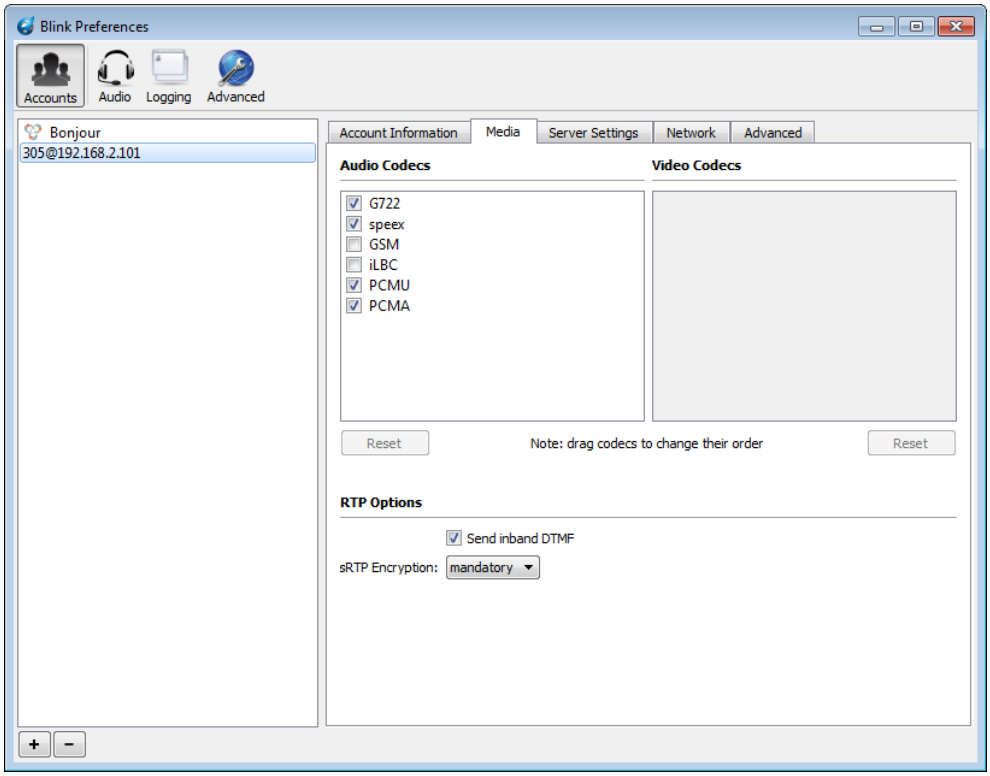


- En el caso de un cliente Bria (Counterpath) para iOS, la configuración se realiza en la cuenta en Opciones avanzadas de la cuenta → Transporte y Seguridad → Encriptar audio

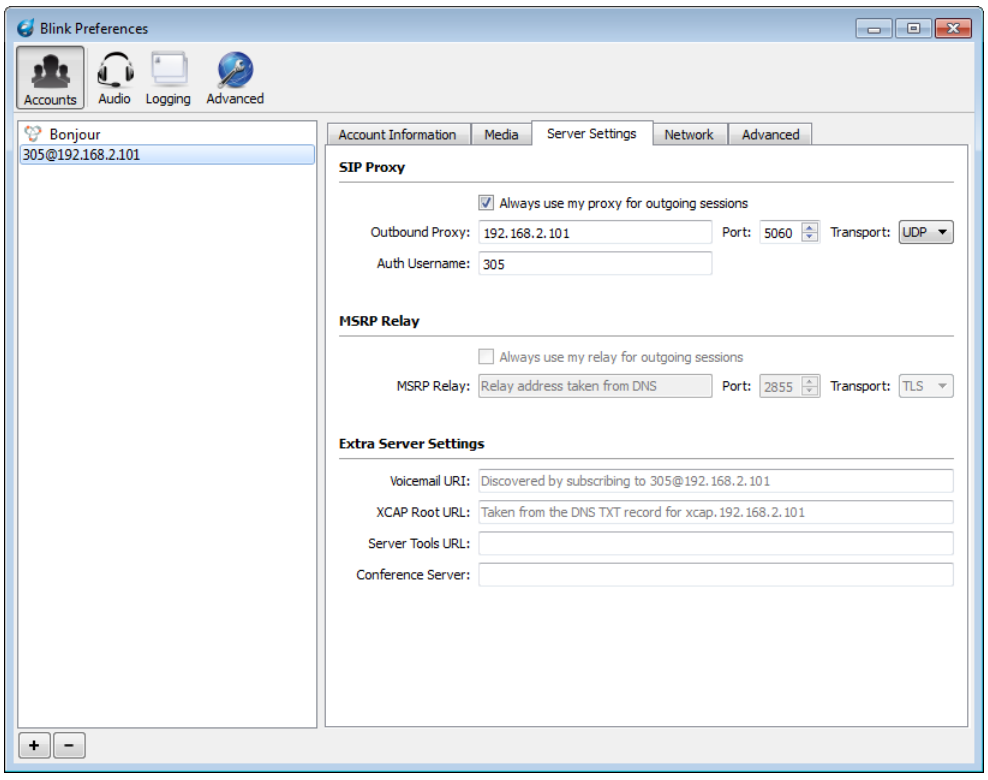


- Configuraremos el softphone Blink para Windows, el cual incluye soporte para SRTP y TLS en sus versiones gratuitas.

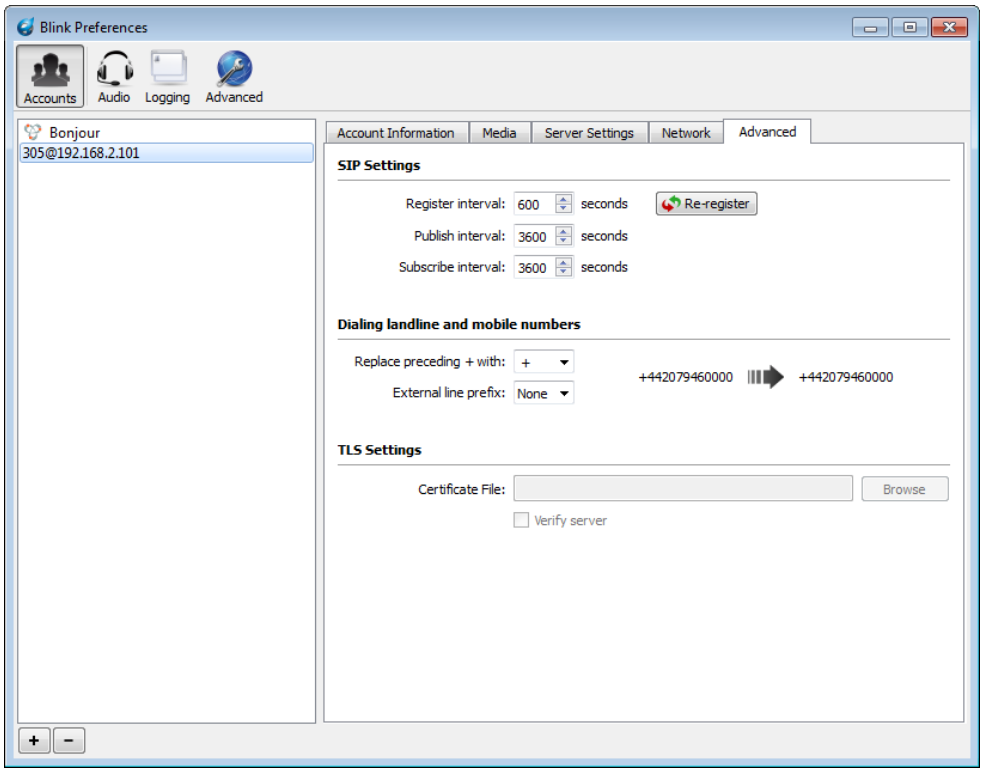
- En la sección “Media”, en el campo “sRTP Encryption” seleccionamos **mandatory**



- Procedemos a configurar los siguientes parámetros en server settings



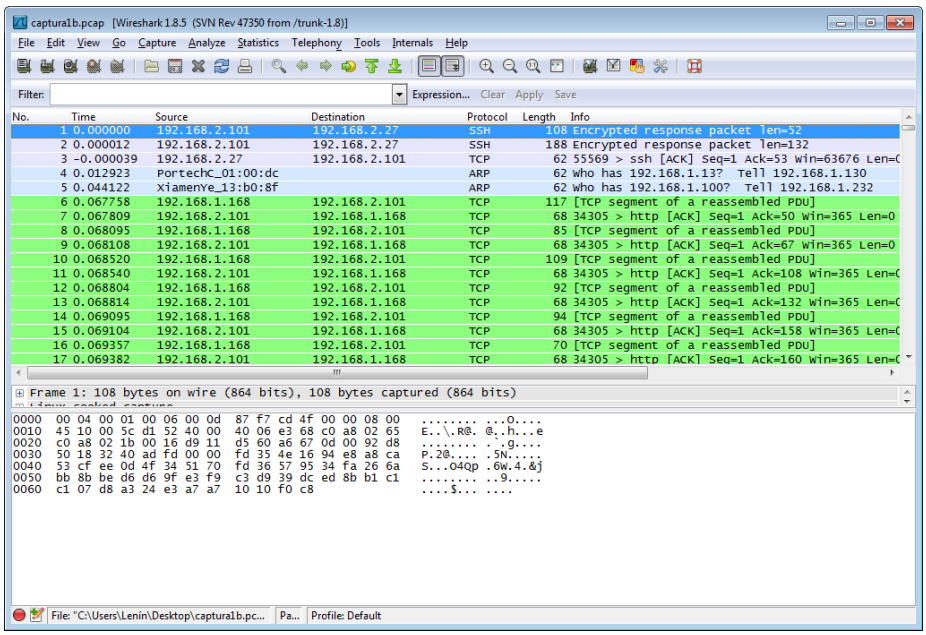
- En la sección “Advanced” , del menú “Accounts”, editamos las opciones Register interval, Publish interval y Subscribe interval, tal como se muestra en la figura.



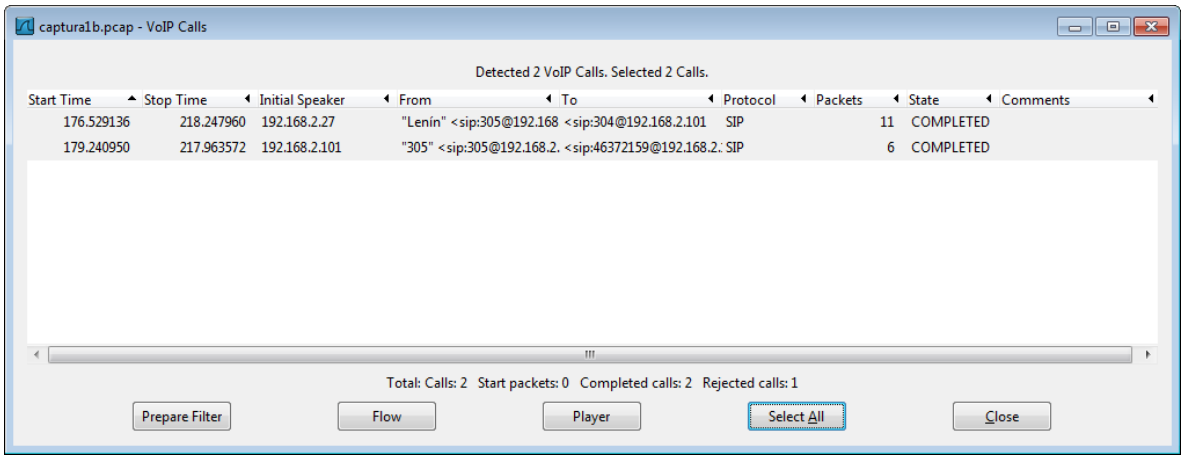
- Una vez que hemos realizado las configuraciones correspondientes, realizaremos una captura entre las extensiones de prueba usando tcpdumpba.

- Obtenemos el archivo de la captura desde nuestro servidor Elastix e iniciamos Wireshark, luego hacemos clic en “Open a previously captured file” y seleccionamos el archivo de la captura.

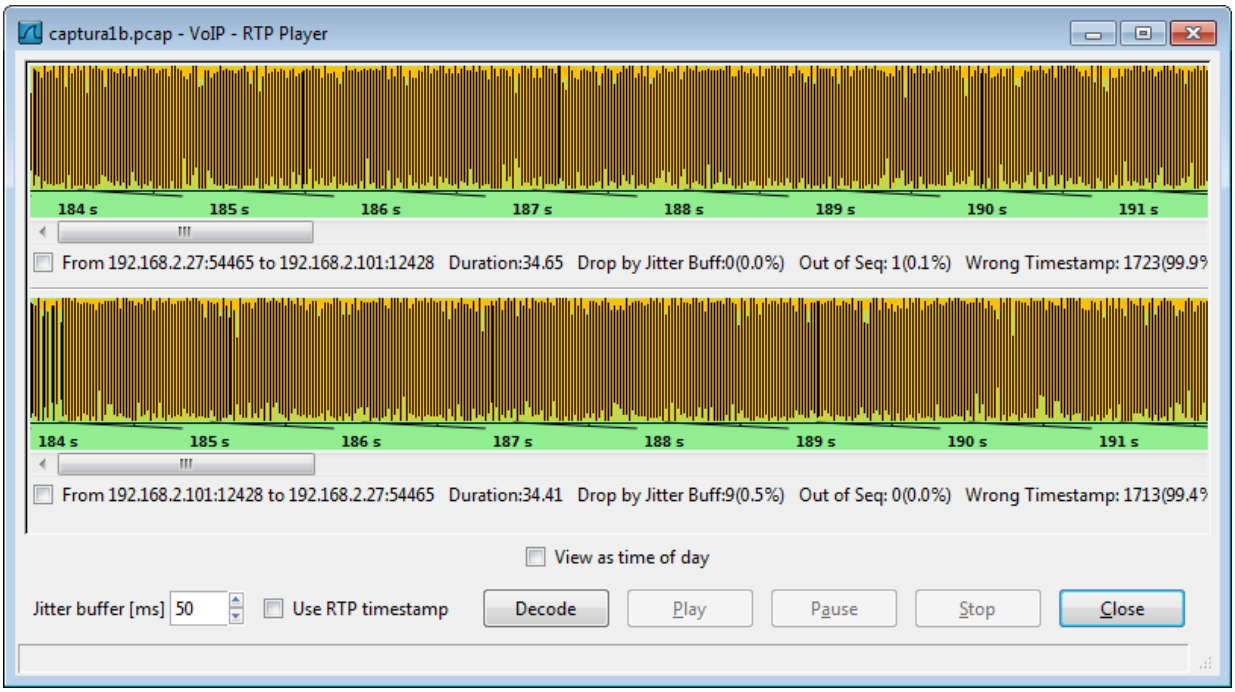
  Nota: Es necesario obtener el archivo de nuestro servidor Elastix, cuando sea necesario. Al igual que en el laboratorio 6.2, podemos usar usar un cliente SCP.



- En Wireshark vamos a “Telephony → VoIP Calls”, allí vamos a obtener un listado de las llamadas VoIP junto a su resultado (COMPLETE, REJECTED, CANCEL).



- Seleccionamos cualquiera de las dos llamadas y hacemos clic en “Player” y luego en “Decode”. Al hacerlo nos damos cuenta que no es posible entender lo que los participantes de la llamada estaban conversando en el momento.

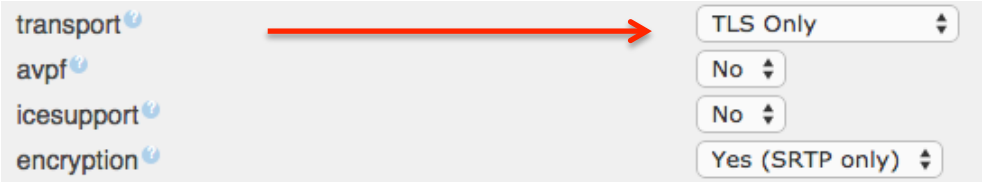


- La imagen anterior muestra la incorporación de ruido al canal de audio.

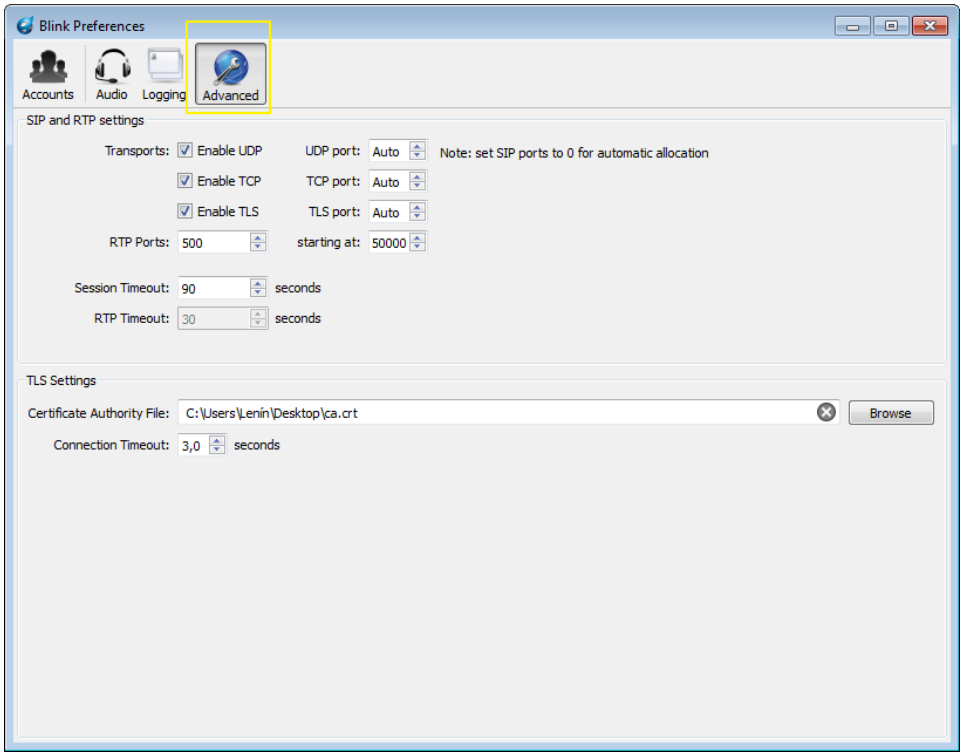
Usando TLS

- Ahora, usaremos TLS junto con SRTP y realizaremos una captura.

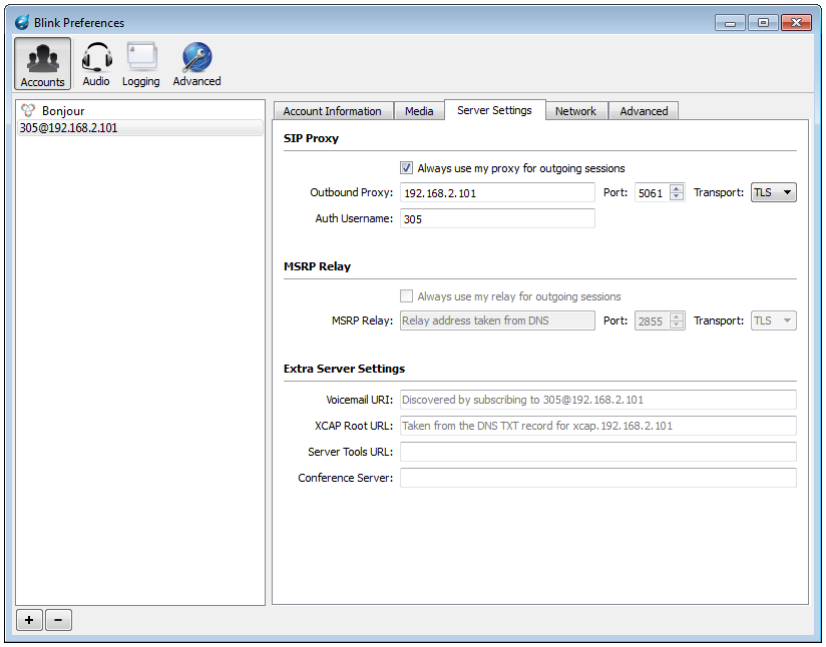
- Habilitamos transport=tls en la extension (PBX → PBX Configuration → Extensions) tal como se muestra en la figura.



- Es hora de subir el certificado creado en nuestro servidor Elastix en blink. Nos dirigimos a la parte superior al menú “Advanced”, hacemos clic en “Browse” y subimos el archivo ca.crt.



- Luego vamos a la cuenta y en la sección “Server Settings” editamos el puerto al 5061 y la opción TLS en “Transport”.



- Repetimos el procedimiento en ambas extensiones.

  Nota: En el caso de la versión de Blink para Mac, la configuración se debe realizar en Avanzado → Señalización SIP. En el campo proxy primario es necesario colocar IP_Servidor:5061;transport=tls. Esta versión de Blink no requiere que subamos un certificado.

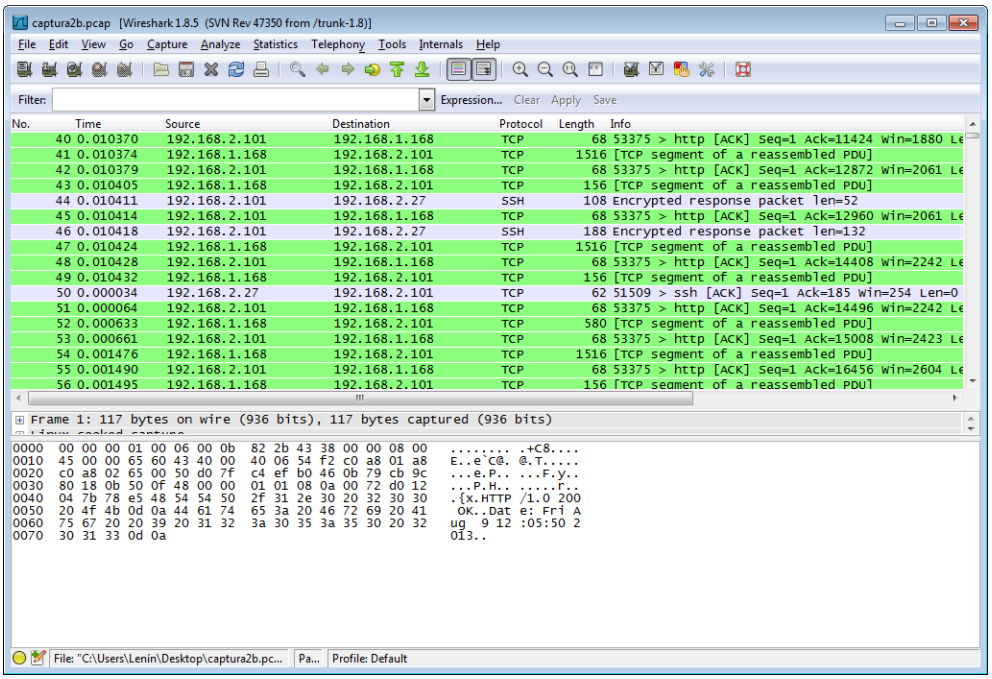
- En el caso de Bria (Counterpath) para iOS, no es necesario subir un certificado. Este cliente acepta el certificado sin intentar verificarlo, cuando la opción "Verficar certificado TLS", está en off. Tampoco es necesario especificar el puerto de transporte.

- Activamos el transporte por TLS en en la cuenta en Opciones avanzadas de la cuenta → Transporte y Seguridad → Transporte de SIP

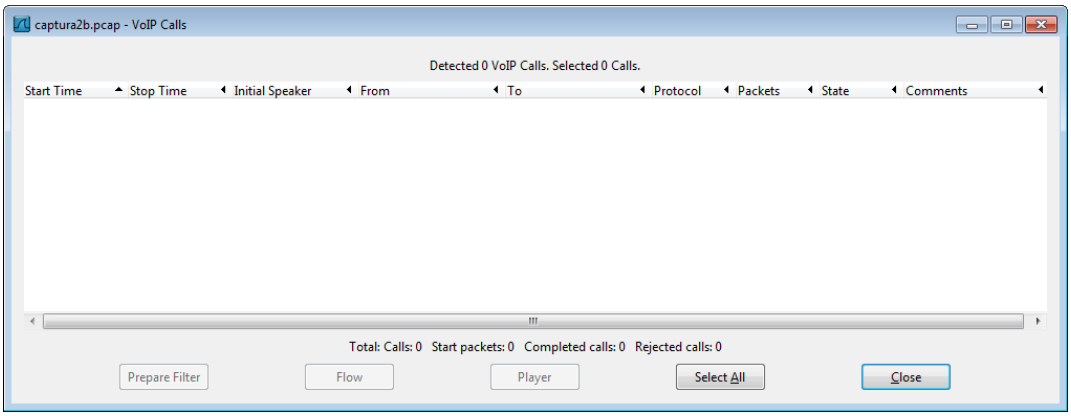
- Si no tenemos dos clientes endpoint que soporten TLS y SRTP, podemos hacer la práctica con una sola extensión.

- Iniciamos la capruta de paquetes, realizamos la llamada y obtenemos el archivo.

- Abrimos Wireshark y nos dirigimos a “Telephony → VoIP Calls”.



- Este es el resultado que obtenemos al usar TLS



- TLS cifra completamente la señalización y como conclusión no se puede ver el diagrama de respuestas SIP como en la captura anterior.

Laboratorio 6.4 (Práctica recomendada)

Descripción: Algunos filtros útiles para tcpdump.

Objetivo: Adiestrar al estudiante en el uso de sniffers para analizar señalización y/o los paquetes RTP con mayor detalle.

Tiempo Máximo: 10 minutos.

Instrucciones:

- Ingresamos a la consola como usuario root
- Realizamos una captura de audio desde el directorio `pruebaws`

```
cd pruebaws
```

- Vamos a realizar una captura de un host en concreto y de una interfaz de red en particular. Ejecutamos el siguiente comando:

```
tcpdump -i eth0 -s0 -w capturaXX.pcap udp port 5060 and host XXX.XXX.XXX.XXX
```

- Donde XXX.XXX.XXX.XXX debe ser la IP del teléfono IP. Esto nos permite capturar solamente los paquetes SIP cuyo origen/destino sea el host XXX.XXX.XXX.XXX.
- Con esto reducimos significativamente el tamaño del archivo capturado y podemos realizar una captura más extensa. Ideal para servidores Elastix con un alto flujo de llamadas.
- Para realizar una captura en IAX, de un host en concreto y de una interfaz de red en particular, ejecutamos el siguiente comando:

```
tcpdump -i eth0 -s0 -w capturaXX.pcap udp port 4569 and host XXX.XXX.XXX.XXX
```

- Para realizar la captura de paquetes UDP, de un host en concreto y de una interfaz de red en particular, ejecutamos el siguiente comando:

```
tcpdump -i eth0 -s0 -w capturaXX.pcap udp and host XXX.XXX.XXX.XXX
```

Analizar los archivos generados, usando lo aprendido en los laboratorios anteriores.

Laboratorio 6.5 (Práctica recomendada)

Descripción: En este laboratorio instalaremos el códec G.729 versión gratuita desde los repositorios de Asterisk.

Objetivo: Instalación del códec G.729

Tiempo Máximo: 20 minutos.

Instrucciones:

Digium distribuye una versión comercial del codec g729 y puede ser adquirida en el siguiente enlace:

http://www.digium.com/en/products/software/g729-codec

- En este laboratorio usaremos una versión no comercial para propósitos académicos. Nos dirigimos a la siguiente dirección:

  http://asterisk.hosting.lv/

- Bajo "Linux Binaries" encontramos tres categorías:

  Asterisk 1.8
  Asterisk 11
  Asterisk 12

- Procedemos a revisar la versión de Asterisk instalada en el sistema, que es muy importante a la hora de elegir el códec requerido.

- En la consola de linux de nuestro Elastix ejecutamos

```
asterisk –vvvr
```

```
[root@ecte ~]# asterisk -vvvr
Asterisk 1.8.20.0, Copyright (C) 1999 - 2012 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 1.8.20.0 currently running on ecte (pid = 2903)
Verbosity is at least 3
ecte*CLI>
```

- Una vez que encontramos la versión hacemos clic en una de las categorías encontradas en la página web y obtendremos un listado de codecs disponibles. Hay varias alternativas.

- Para la selección del binario adecuado debemos tomar en consideración la arquitectura del sistema (32 bits, 64 bits) y el tipo de procesador que tenemos.

- Para obtener información lo más precisa posible ejecutamos el siguiente comando desde consola:

```
cat /proc/cpuinfo
```

```
[root@ecte ~]# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 23
model name      : Intel(R) Core(TM)2 Duo CPU     P8600  @ 2.40GHz
stepping: 10
cpu MHz         : 1714.448
cache size      : 6144 KB
fdiv_bug: no
hlt_bug         : no
f00f_bug: no
coma_bug: no
fpu             : yes
fpu_exception   : yes
cpuid level     : 5
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic mtrr pge mca cmov pat
pse36 clflush mmx fxsr sse sse2 nx constant_tsc up pni monitor ssse3
bogomips: 3428.89
```

- Basaremos nuestra selección en la información obtenida en el campo “model name” y el campo “flags”. Además es importante revisar primero las notas que se incluyen en la página:

Notas

- Después de descargar revise MD5SUM.
- Use estructuras x86_64 si está operando en modo de 64-bits. Los Binarios que no contienen x86_64 en el nombre son de 32-bit.
- Use estructura Pentium 4 para Pentium D. Use Pentium 3 para VIA C3 y Pentium 4 para VIA C7.
- XEON es una rama de CPU para servidores de la familia Pentium3/Pentium4/Core. Celeron es Pentium3/Pentium4/Core con menor cache. Básicamente, hay 5 núcleos de software: MMX, SSE, SSE2, SSE3, SSE4. Escoger el más cercano CPU Intel y probarlo.
- GCC4, ICC - todos los binarios son compatibles con estructuras de Asterisk creadas con otros compiladores, se debe usar el más rápido o mejor para usted.
- Las estructuras sse3 son para iteraciones Prescott de chips Pentium 4 con soporte SSE3, revisar por PNI en flag en /proc/cpuinfo.
- Núcleos ICC no siempre son más rápidos, revisar con '(core) show translation recalc 10' en la consola de Asterisk, asegúrese que el servidor esté desocupado o el timing puede ser incorrecto.

- En el caso de este ejemplo, el procesador es Core2 para el cual existen las siguientes opciones:

codec_g729-ast18-gcc4-glibc-core2-sse4.so
codec_g729-ast18-gcc4-glibc-core2.so
codec_g729-ast18-icc-glibc-core2-sse4.so
codec_g729-ast18-icc-glibc-core2.so

- De acuerdo a la información obtenida en flags, nuestro servidor no tiene instrucciones sse4, por lo cual nuestras alternativas se reducen a dos:

codec_g729-ast18-icc-glibc-core2.so

codec_g729-ast18-gcc4-glibc-core2.so

- ICC y GCC4, son compiladores usados para compilar el binario, el primero es una versión comercial de Intel y el segundo es un compilador GNU de versión 4 que no tiene cargo. Aunque el primero debe producir mejores y optimizados binarios, podemos probar ambos en nuestro equipo y escoger el que tenga mejor timing.

  Nota: En este ejemplo solo instalaremos el primero, pero usted puede instalar varias opciones y revisar el timing, lo cual se detallará al final de esta práctica.

- Una vez que seleccionamos el binario a instalar copiamos su enlace para usarlo posteriormente.

- Nos dirigimos al directorio /usr/lib/asterisk/modules desde ahí obtendremos el binario utilizando el comando wget y el link del binario que copiamos anteriormente.

  ```
  wget http://asterisk.hosting.lv/bin/codec_g729-ast18-icc-glibc-core2.so
  ```

- Ahora debemos cambiar el nombre al codec (`mv nombre_actual.so nombre_nuevo.so`)

  ```
  mv codec_g729-ast18-icc-glibc-core2.so codec_g729.so
  ```

- Reiniciamos Asterisk con service asterisk restart

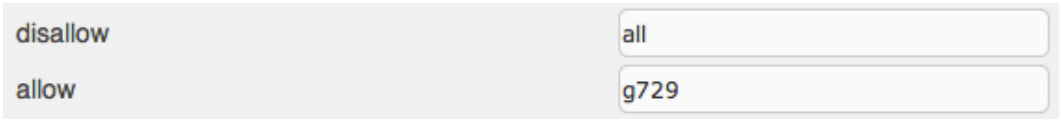
- Ahora ingresamos a la consola de Asterisk y ejecutamos “core show translation”

```
[root@ecte modules]# asterisk -r
Verbosity is at least 3
ecte*CLI> core show translation
         Translation times between formats (in microseconds) for one
          Source Format (Rows) Destination Format (Columns)

            g723   gsm  ulaw  alaw g726aal2 adpcm  slin lpc10  g729
       g723    -     -     -     -        -     -     -     -     -
        gsm    -     -     2     2     1001     2     1  1001  1001
       ulaw    -  1001     -     1     1001     2     1  1001  1001
       alaw    -  1001     1     -     1001     2     1  1001  1001
    g726aal2   -  1999  1000  1000        -  1000   999  1999  1999
      adpcm    -  1001     2     2     1001     -     1  1001  1001
       slin    -  1000     1     1     1000     1     -  1000  1000
      lpc10    -  2000  1001  1001     2000  1001  1000     -  2000
       g729    -  1001     2     2     1001     2     1  1001     -
      speex    -  1001     2     2     1001     2     1  1001  1001
       ilbc    -  1001     2     2     1001     2     1  1001  1001
       g726    -  2000  1001  1001     2000  1001  1000  2000  2000
       g722    -  1001     2     2     1001     2     1  1001  1001
     siren7    -     -     -     -        -     -     -     -     -
     siren14   -     -     -     -        -     -     -     -     -
      slin16   -  2001  1002  1002     2001  1002  1001  2001  2001
       g719    -     -     -     -        -     -     -     -     -
     speex16   -  3001  2002  2002     3001  2002  2001  3001  3001
     testlaw   -  1001     2     2     1001     2     1  1001  1001
```

- Si todo salió bien podremos ver los tiempos de transcoding entre g729 y otros codecs.

- Una comprobación adicional se puede realizar habilitando g729 en una de las extensiones que estamos usando. Para hacerlo vamos a la interfaz de administración web de Elastix a PBX → Configuración PBX → Extensiones. Entramos a la extensión y editamos los campos disallow y allow en la sección "Device Options" como se muestra la imagen.



- Luego hacemos clic en "Submit" y aplicamos los cambios. Ahora haremos una prueba marcando 1234, deberíamos escuchar la voz de Allison Smith dándonos la bienvenida a Asterisk.

- Para verificar el timing ejecutamos desde la consola de Asterisk el siguiente comando: `core show translation recalc 10`

- En una prueba, fuera de laboratorio, hicimos la comparación de los datos obtenidos con los dos binarios disponibles para nuestro servidor y obtuvimos los siguientes resultados:

	g729	
Codec	**A**	**B**
gsm	1899	798
ulaw	1600	500
alaw	1600	500
g726aa12	1899	798
adpcm	1600	500
slin	1599	499
lpc10	1899	4098
speex	1999	4498
ilbc	4398	1298
g726	1799	698
g722	1699	898
slin16	2298	2597
speex16	3097	3496
testlaw	1600	500

Alternativa A) codec_g729-ast18-icc-glibc-core2.so
Alternativa B) codec_g729-ast18-gcc4-glibc-core2.so

- Como podemos observar en el cuadro, el binario de la alternativa B, tiene un mejor timing para la traducción o transcoding hacia otros codecs, como por ejemplo g711 (ulaw, alaw), en cambio la alternativa A tiene un mejor tiempo de traducción con el codec speex y lpc10.

- Dependiendo el uso que tenga nuestra operación de telefonía debemos hacer la selección, si vamos a usar un porcentaje alto de extensiones con codec g711, entonces nos conviene usar la alternativa B.

Laboratorio 6.6 (Práctica Recomendada)

Descripción: Habilitar los codecs de video y probar video llamada con teléfonos grandstream.

Objetivo: Habilitar codecs de video

Tiempo Máximo: 10 minutos.

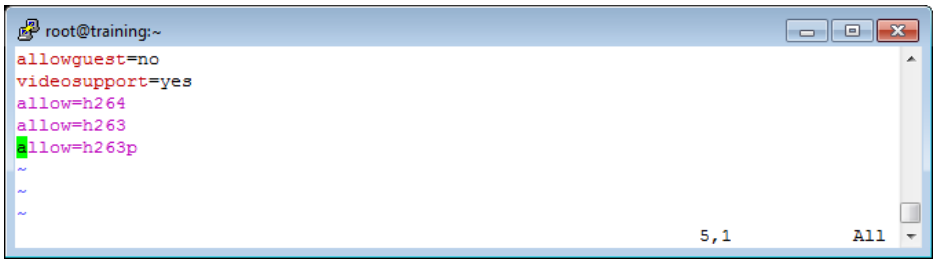
Instrucciones:

- Habilitamos llamada de video editando el archivo

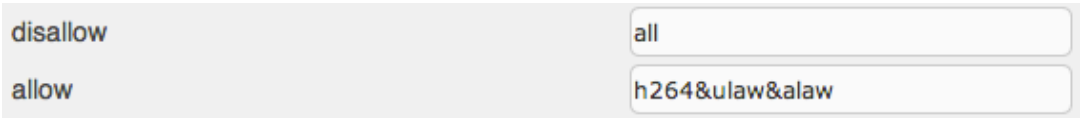
  ```
  /etc/asterisk/sip_general_custom.conf
  ```

- Una vez dentro del archivo agregamos las siguientes líneas:

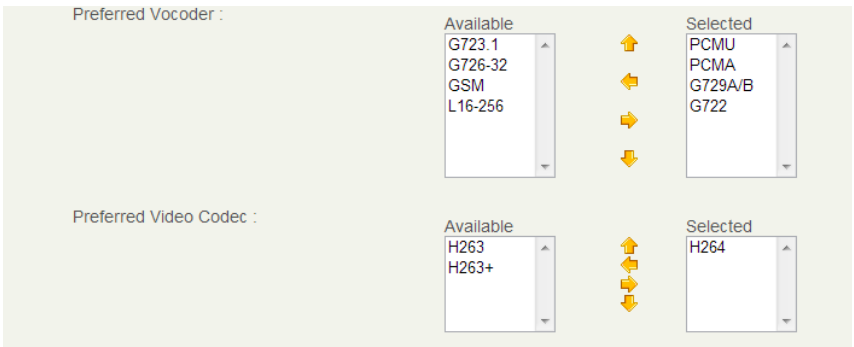
  ```
  videosupport=yes
  allow=h264
  allow=h263
  allow=h263p
  ```



- Ir a la interfaz Elastix y configurar las extensiones asociadas a los teléfonos y editar los campos disallow y allow en la sección “Device options”:



- Deben de estar habilitados los codecs en el teléfono Grandstream



- Una vez realizado las siguientes configuraciones podremos realizar la video llamada.

Laboratorio 6.7 (Práctica Recomendada)

Descripción: Configurar el archivo sip_nat.conf para solucionar problemas de NAT

Objetivo: Configurar el archivo sip_nat.conf

Tiempo Máximo: 10 minutos.

Instrucciones:

- En ocasiones suele suceder que no podemos realizar o recibir llamadas aun cuando todo esté configurado correctamente.
- Esto puede suceder debido a problemas de NAT para lo cual recomendamos la siguiente configuración:
- Editar el siguiente archivo:

  ```
  /etc/asterisk/sip_nat.conf
  ```

- Dentro de ese archivo incluir:

  nat=yes
  externip=IP_externa/mascara_de_red
  localnet=IP_Interna/mascara_de_red

- Ejemplo:

  ```
  nat=yes
  externip=200.4.5.23/255.255.255.255
  Localnet=192.168.1.0/255.255.255.0
  ```

- Grabar y Salir (Esc → `:wq`)
- Recargar asterisk: `[root@elastix asterisk]# asterisk -rx "reload"`

El contenido de este libro está sujeto a mejoramiento. Si usted encuentra errores, por favor envíe un email a training@elastix.com y recibirá una actualización gratis en la siguiente edición.