

Introducción a la VoIP y Asterisk

Irontec VoIP <vozip@irontec.com>



<http://www.irontec.com>



irontec

irontec

Internet y Sistemas sobre GNU/Linux

<http://www.irontec.com>

Antes de empezar

- Conocimientos previos recomendados:
 - Manejo básico de la consola de GNU/Linux.
 - Interés
 - Ganas de aprender
 - $C_8H_{10}N_4O_8$
- ¿Qué sabes de Asterisk?
 - ¿Lo has utilizado?
 - ¿Qué intenciones de uso tienes?



Introducción

ironotec
GNU/Linux

Conceptos básicos sobre telefonía tradicional

Telefonía Tradicional

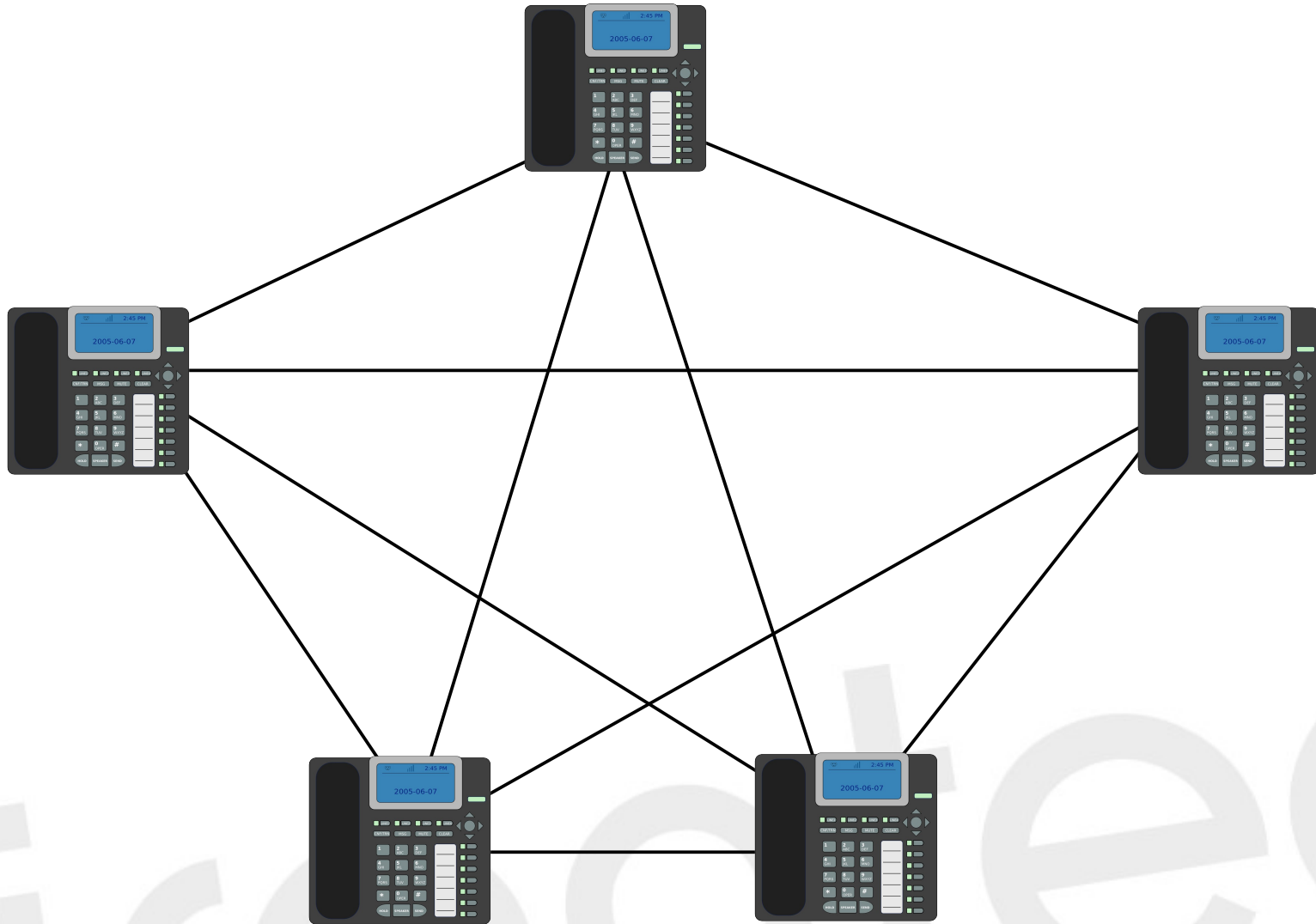
- Teléfono Inventado en 1876 por Antonio Meucci (atribuido a Alexander Graham Bell hasta el 2002).
- Idea principal:
 - Hacer audible la palabra hablada a largas distancias
- Originalmente: Transmisión sobre un hilo de hierro, comunicación punto a punto.
- Hoy en día: 1000 millones de teléfonos repartidos por todo el mundo.

Conmutación de circuitos

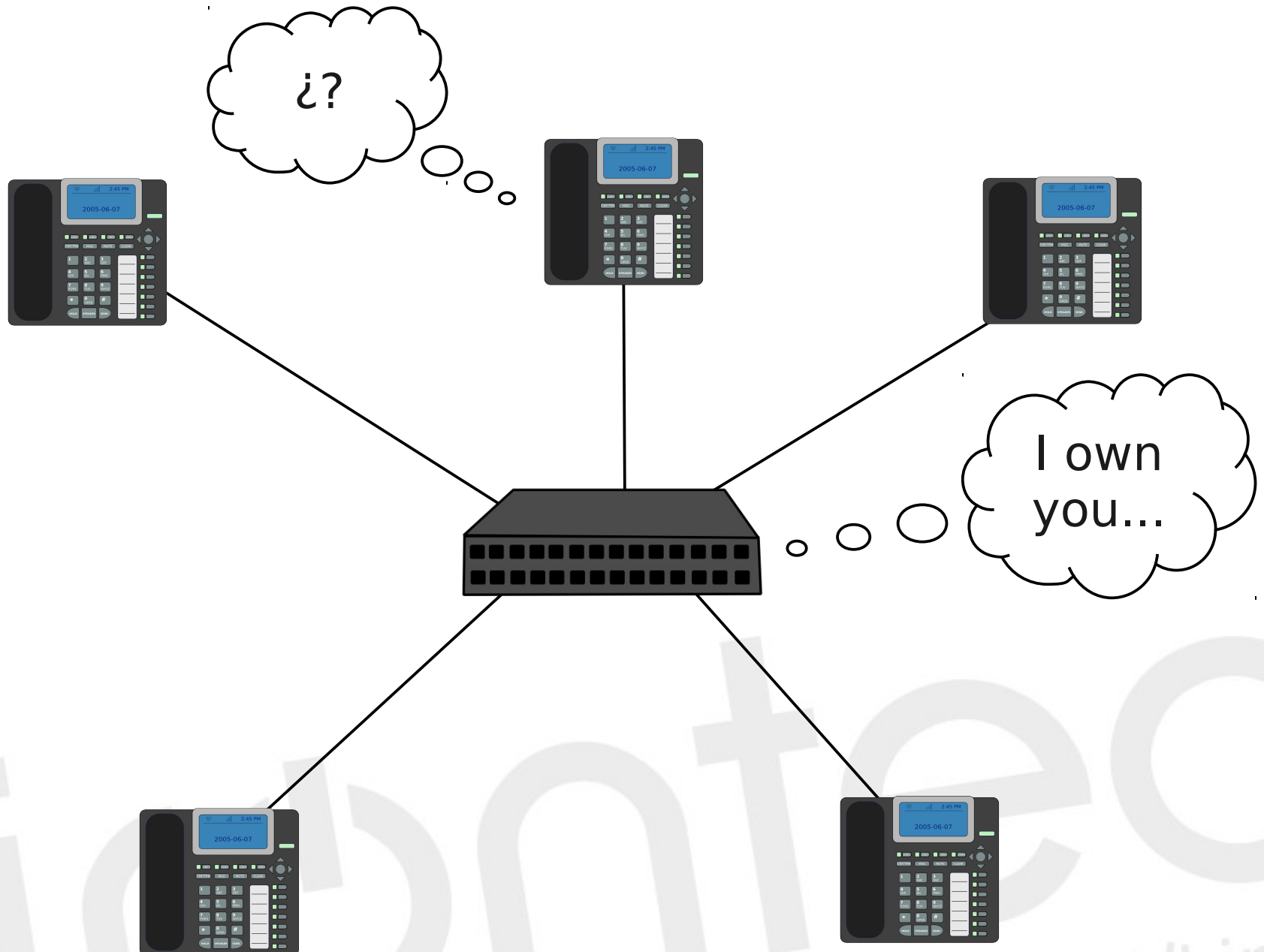
- La telefonía tradicional se basaba en conmutación de circuitos.
- Desde el comienzo hasta el final de una llamada se establecía un camino físico.
 - Consumo de recursos.
- Inicialmente -> redes totalmente malladas
 - 0% escalable.
 - Cambio a estructura en estrella.



Conmutación de circuitos (2)



Conmutación de circuitos (3)



Conmutación de circuitos (4)

- Al principio, telefonía 100% analógica.
- Gestión del crecimiento
 - Analógico: FDM
 - Digital: TDM
- Posteriormente los switches se sustituyeron por switches digitales.
- Digital vs. Analógico
 - Digital es más barato.
 - Digital tiene mejor calidad.
 - Analógico más rápido (switching).
 - Complejidad de los terminales digitales.
- Solución: terminales analógicos y red troncal digital.

Señalización

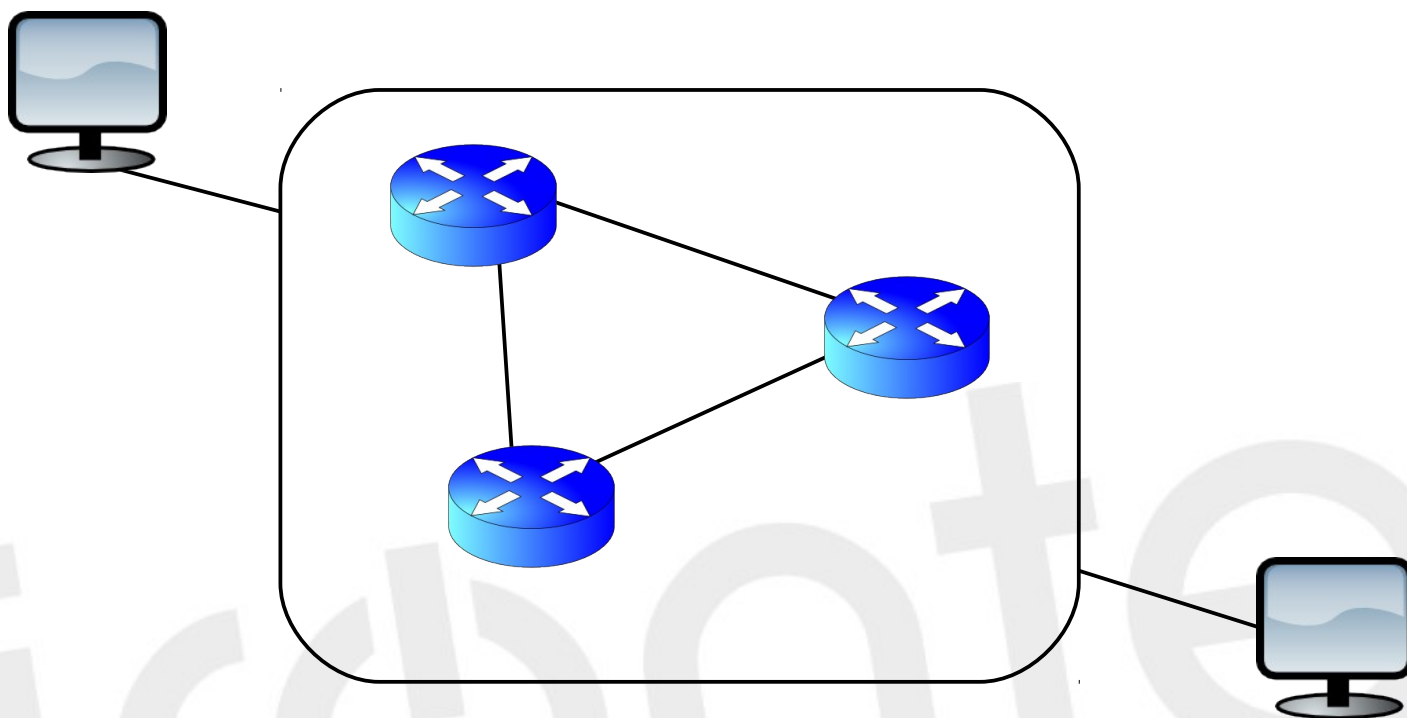
- Necesidad de comunicación entre distintos sistemas.
- Ligado a la evolución de centrales/terminales de usuario
- Señalización analógica
 - Inband
- Señalización digital
 - Access signalling (del terminal a la central)
 - Ex. DTMF
 - Trunk signalling (entre centrales)
 - CAS (señalización asociada al canal)
 - CCS (señalización por canal común)

Señalización (2)

- Actualmente se usa SS7 (CCS)
 - Señalización asociada al circuito
 - Relativa a la llamada
 - Señalización no asociada al circuito
 - Consulta de tablas de encaminamiento
 - Servicios suplementarios
 - Desvíos de llamada...
- Paradigma de SS7
 - La inteligencia reside en la red (terminales 'tontos')
 - El acceso a la red determina los servicios disponibles

Conmutación de paquetes

- En conmutación de circuitos raramente se utilizaba todo el ancho de banda disponible.
- TDM ineficiente gestionando el uso de la red.
- El contenido del paquete determina la ruta.



Conmutación de paquetes VS. Conmutación de circuitos

- C. de circuitos
 - Más rápida
 - No se examina el contenido de los paquetes
- C. de paquetes
 - Mejor gestión de recursos
 - Precio

El paradigma IP

- Su ÚNICO propósito es proporcionar conectividad.
- La red es independiente de la tecnología subyacente.
- Las aplicaciones pueden utilizar una infraestructura común IP.

Aplicaciones

Conectividad IP

Ethernet | ATM | ...

El paradigma IP (2)

- Protocolos de extremo a extremo
 - IP solo 'lleva' cosas
 - La INTELIGENCIA esta en los extremos
 - Internet es idiota :)
- Justo lo contrario que en la telefonía tradicional...



¿Qué es la VoIP?

ironotec

CNU/Lin

Conceptos básicos sobre VozIP

Voz sobre IP: ¿ Qué es ?

Utilizar redes de datos IP para realizar llamadas de Voz.

- En particular: Realizar llamadas por Internet (IP = Internet Protocol).
- Internet: La mayor red de datos del mundo.
- La tecnología Voz sobre IP se encuentra ahora mismo en su madurez, pero comenzó por los años 90.
- Tecnología conocida como 'VoIP'.

Conceptos básicos sobre VozIP

Voz sobre IP: Características Principales

- Se utiliza una única red. Si dos empresas están unidas a través de Internet, ¿Porqué no aprovecharlo?
- Se administra una única red.
- **Finalmente** se puede hablar de: **Estándares abiertos e internacionales**. Interoperabilidad, Bajada de precios en proveedores y fabricantes de hardware VoIP.
- **Calidad:** Es posible conseguir la misma calidad e incluso mayor gracias a nuevos codecs.
- **Fiabilidad:** En LAN, se puede lograr una gran fiabilidad. En Internet también, pero existen quizás demasiados factores.

Razones del éxito de VoIP

- Gran expansión actual de las redes de datos
 - LAN, WAN...
 - Internet: ADSL, ADSL2+, VDSL
 - WIFI, WiMax...
- Posibilidad de desarrollar **nuevos servicios** rápidamente.
- Menor inversión inicial para los proveedores..
- Costes más bajos para los clientes.

VoIP: Problemas

- **NAT:** El cáncer de la VoIP. Distintos tipos, no es fácilmente manejable.
- **QoS:** Necesidad de ofrecer calidad de servicio, al ser la voz crítica en tiempo real.
- **Latencia:** Tiempo que tarda la voz en llegar al destino.
- **Jitter:** Variación de la latencia.
- **Ancho de banda:** En España conexiones asimétricas. El ancho de banda es muy caro.

Conceptos básicos sobre VozIP

Voz sobre IP: Elementos Implicados

- **Teléfonos IP:** Físicamente, son teléfonos normales, con apariencia tradicional. Incorporan un conector RJ45 para conectarlo directamente a una red IP en Ethernet. No pueden ser conectados a líneas telefónicas tradicionales.



SNOM 360



SIPURA 841

Conceptos básicos sobre VoIP

Voz sobre IP: Elementos Implicados (II)

- **Adaptadores analógicos IP:** Permiten aprovechar los teléfonos analógicos actuales, transformando su señal analógica en los protocolos de VoIP.

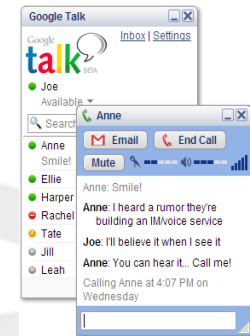
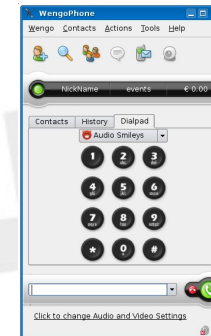
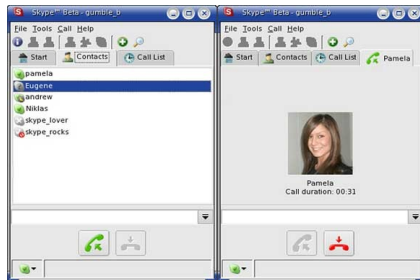


LINKSYS PAP2

Conceptos básicos sobre VoIP

Voz sobre IP: Elementos Implicados (III)

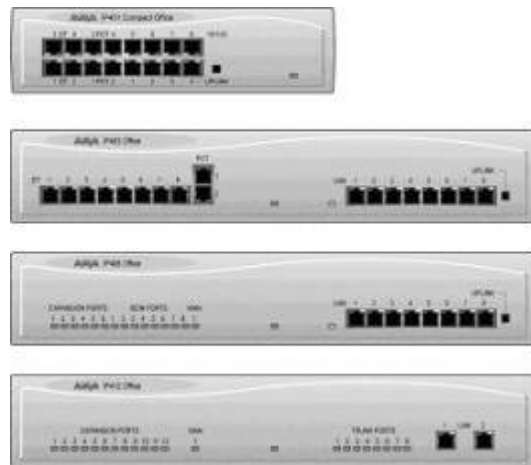
- **Softphones:** Programas que permiten llamar desde el ordenador utilizando tecnologías VoIP.



Conceptos básicos sobre VozIP

Voz sobre IP: Elementos Implicados (IV)

- **Centralitas IP:** Centralitas de telefonía que permiten utilizar de forma combinada la tecnología VozIP (mixtas) o exclusivamente IP (puras).

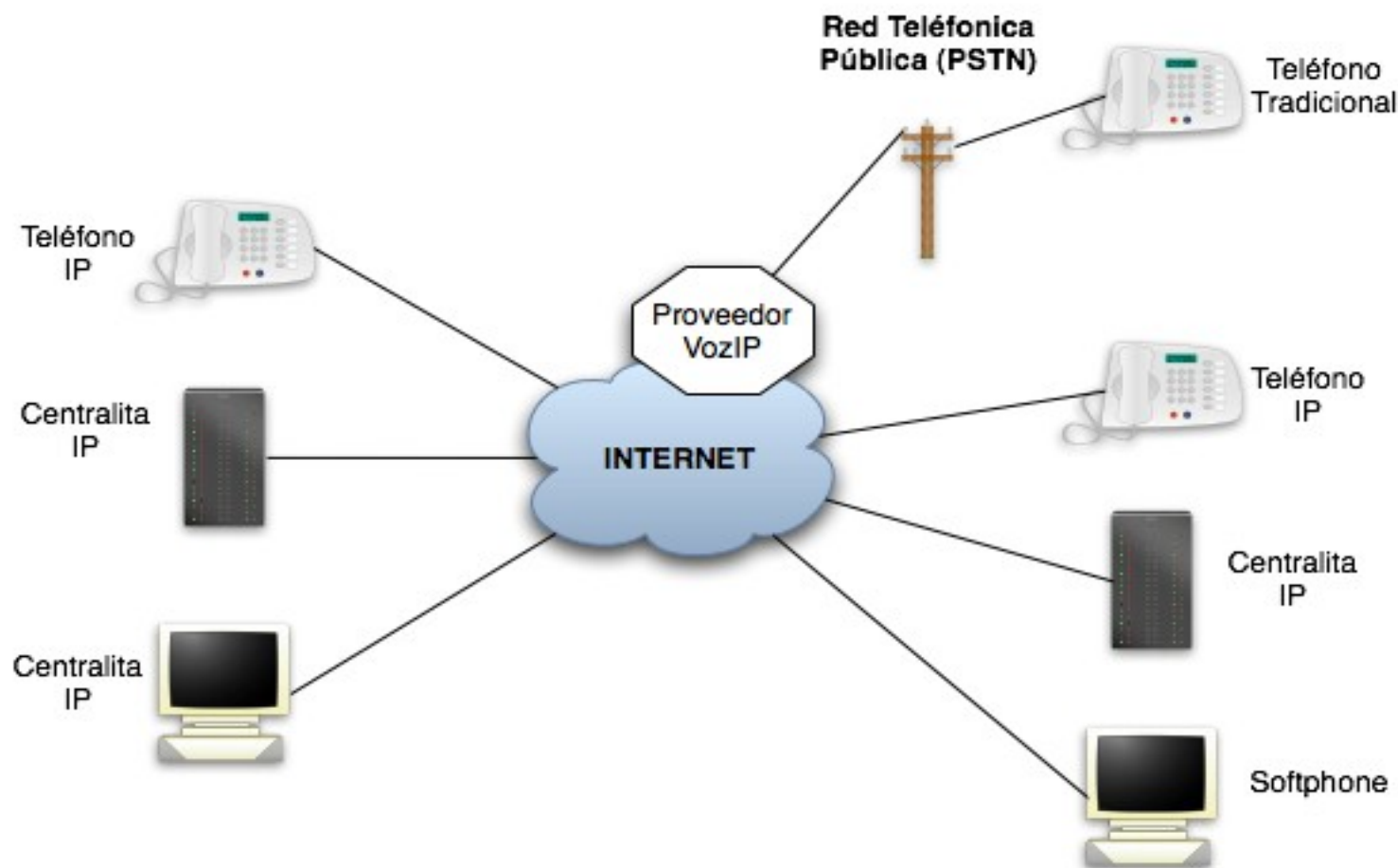


Avaya IP Office



Conceptos básicos sobre VozIP

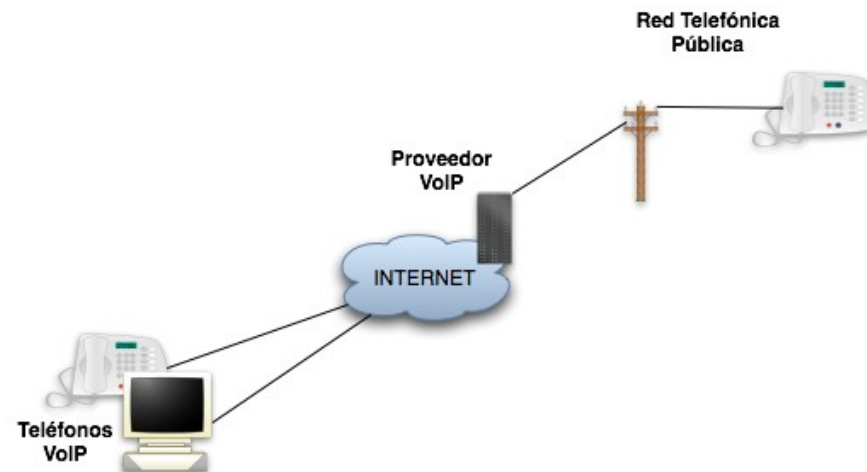
Voz sobre IP: Visión General



Proveedores de Servicios VozIP

Función Principal

- El principal servicio de los diferentes proveedores de Voz sobre IP es el de hacer de pasarela hacia la red telefónica pública (conocida como PSTN/POTS) a costes muy reducidos.



Proveedores de Servicios VozIP

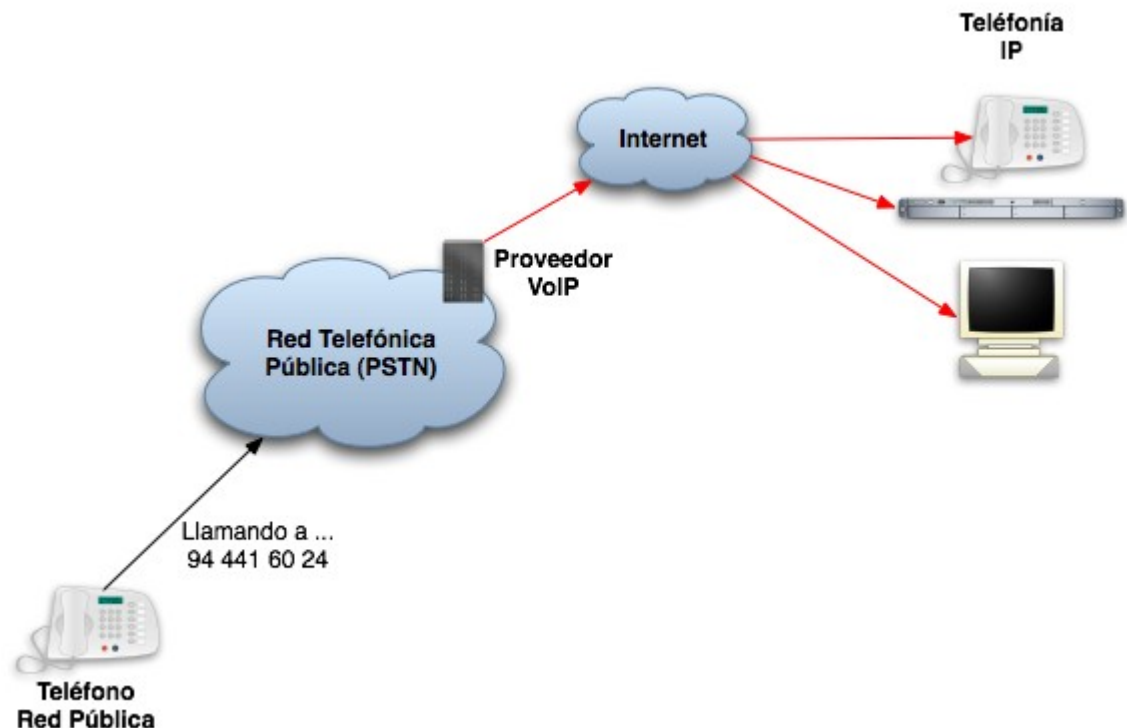
Características Principales

- Soportan determinados protocolos estándar (SIP, H323 normalmente).
- Algunos tienen protocolos propietarios: Skype, ...
- Soportan determinados codecs (GSM, G.729 normalmente).
- Casi siempre permiten realizar más de una llamada a la vez.
- Las llamadas entre usuarios de un mismo proveedor son gratuitas, en algunos casos existen 'prefijos' para saltar entre redes de proveedores conocidos.

Proveedores de Servicios VozIP

Características Avanzadas

- Enlace PSTN -> VoIP
 - Numeración geográfica: 944, 91...
 - Numeración 902, 700 ...



Proveedores de Servicios VozIP

Algunos Proveedores



- iMercado parcialmente sin regular!
- Dependencia de **Internet**.

Tecnologías Voz sobre IP

- **Protocolo:** Es el 'lenguaje' que se utiliza para negociar y establecer las comunicaciones de voz sobre IP. Los más importantes: SIP, H323 e IAX2.
- **Codec:** Es la forma de digitalizar la voz humana para ser enviada por las redes de datos. Algunos ejemplos: G.711, G729, GSM, iLBC, Speex, G.723.

Codecs

- Los codecs se utilizan para transformar la señal de voz analógica en una versión digital.
- Los softphones, hardphones o centralitas IP soportan una serie de codecs cada uno. Cuando hablan entre sí negocian un codec común.
- Aspectos a tener en cuenta por codec:
 - Calidad de sonido
 - Ancho de banda requerido
 - Coste de computación

Tecnologías Voz sobre IP

Comparativa de Codecs

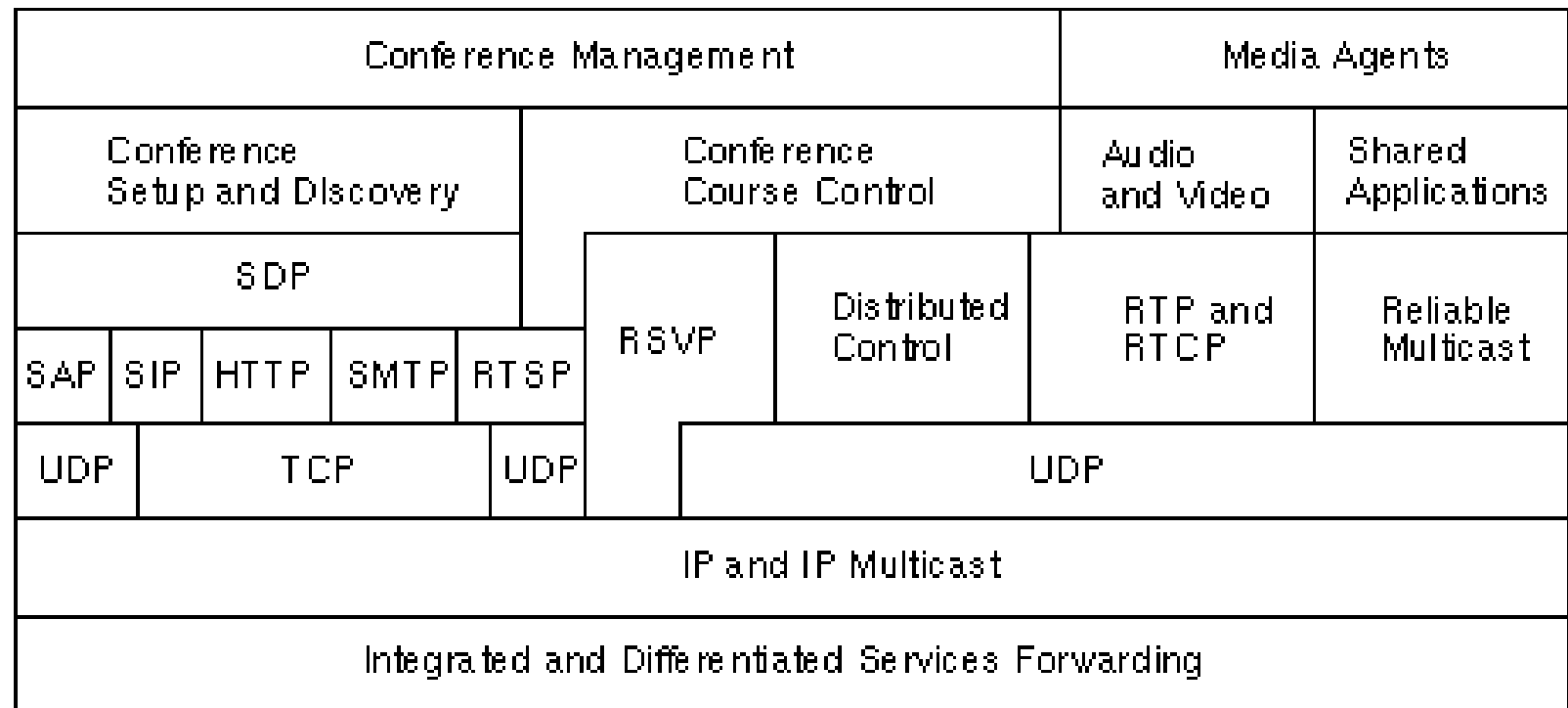
CODEC	Codec Bitrate	Intervalo	A Banda(Ethernet)
G.711	64 Kbps	10ms	87 Kbps
G.729	8 kbps	10ms	31,2 Kbps
Speex	4-44,2 Kbps	30	17,63 – 59,63 Kbps
ILBC	13,3 Kbps	30	30,83 Kbps
G.723.1	6,3 Kbps	37	21,9 Kbps
GSM	13,2 Kbps	20	28,63Kbps

- **Fuentes:** cisco.com (ID:7934), terracal.com (FAQ), asteriskguru.com Bandwith calculator)

El protocolo SIP

ironotec
GNU/Linux

Arquitectura de Conferencias Multimedia en Internet



- Protocolo de transporte en Tiempo Real.
- Requerimiento de aplicaciones con retardo ~ 0 .
- Internet es un medio hostil
 - Latencias
 - Jitter
- Para solucionarlo:
 - Timestamps
 - Números de secuencia
- Si tenemos varios streams de audio/vídeo, es necesaria la sincronización
 - RTCP
 - Asocia los timestamps con un RealTime Clock

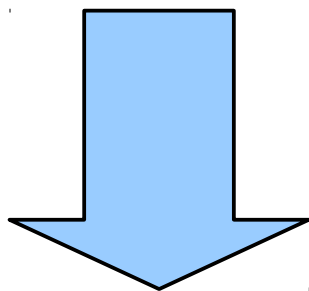
- Session Announcement Protocol
- Sirve para 'anunciar' una sesión multimedia
 - “Hoy a las 8, película de noseke...”
 - Como la revista de la TV
- No se encarga de describir la sesión, para eso tenemos SDP

- Session Description Protocol
- Contiene toda la información que un usuario puede necesitar para unirse a una sesión multimedia.
- Ofrece la siguiente información
 - IP para conectarse a la sesión
 - Codecs soportados
 - Información descriptiva
 - ...



Recapitulando...

- Hasta ahora sabemos hacer 2 cosas
 - Anunciar una sesión multimedia
 - Describirla
- Pero... ¿cómo indicamos a alguien que se una?
 - Hay que INVITARLE a inicial una sesión



SIP: Session Initiation Protocol

- Para cubrir la carencia de no poder iniciar una sesión multimedia con alguien, surgió SIP.
- Estándar de la IETF, recogido en el RFC3261 (SIPv2)
- 'Merge' entre
 - SIPv1 (Session Invitation Protocol)
 - SCIP (Simple Conference Invitation Protocol)



Funcionalidades

- SIP proporciona un mecanismo para iniciar, modificar y finalizar una sesión.
- Independiente del tipo de sesión multimedia y de su descripción.
 - Podemos invitar a alguien a una partida online de mus mediante SIP, utilizando MGDP (Mus Game Description Protocol) para describir la sesión. XD
- Movilidad del usuario
 - Necesidad de conocer su localización.
 - SIP URIs: identificar a usuario SIP.
sip:saghul@irontec.com
 - Los usuarios registran su ubicación en el servidor.

Entidades SIP

- **User-Agent:** entidad con la que interactúa el usuario.
 - Teléfono SIP
 - Softphone
- **Servidor Proxy:** servidor que gestiona las invitaciones a las sesiones
 - Sabe donde esta el usuario destino, así que le enruta el mensaje.
- **Registrar:** servidor que acepta peticiones de registro, y guarda la ubicación del usuario.
- **Location Server:** no es una entidad SIP, pero es necesario para localizar al usuario.
- Normalmente los 3 anteriores son el mismo software.

Porqué SIP es el camino a seguir

- Diferencia entre el establecimiento y la descripción de la sesión
 - Extensible
- Protocolo de extremo a extremo
 - Un usuario ES DUEÑO DE SU SESIÓN
 - Paradigma IP vs. Paradigma SS7
- Favorece la interoperabilidad
 - El 'core' es “”relativamente”” sencillo: 6 métodos
 - Funcionalidades adicionales mediante extensiones
- Es escalable
 - La inteligencia esta en los extremos
 - La red guarda muy pocos datos del estado

- **INVITE**

- Invita a un usuario a una sesión multimedia
- Modifica una sesión multimedia existente

- **ACK**

- Proporciona un 3-way-handshake en el INVITE, sirve para confirmar la recepción de una respuesta final a un INVITE

- **CANCEL**

- Cancela una transacción en curso

- **BYE**

- Se utilizan para abandonar una sesión

- **REGISTER**

- Sirven para informar al servidor de la ubicación del usuario

- **OPTIONS**

- Nos permite consultar qué métodos soporta un usuario.42

Transacciones Cliente-Servidor

- Un cliente GENERA peticiones.
- Un servidor RECIBE peticiones.
- El UA que genera peticiones se conoce como UAC: User Agent Client.
- El UA que responde a las peticiones se conoce como UAS: User Agent Server.
- Una petición, junto con las respuestas que genera, es una TRANSACCIÓN.



Respuestas SIP

- 100 – 199: provisional e informativa
 - 200 – 299: afirmativa
 - 300 – 399: redirección
 - 400 – 499: error del cliente
 - 500 – 599: error del servidor
 - 600 – 699: fallo global
-
- Las respuestas incluyen un mensaje descriptivo, pero lo importante es el código numérico.

Alice

Bob

INVITE



INVITE

180 Ringing

200 OK

Alice

Bob

ACK



INVITE

180 Ringing

200 OK

ACK

Conversación

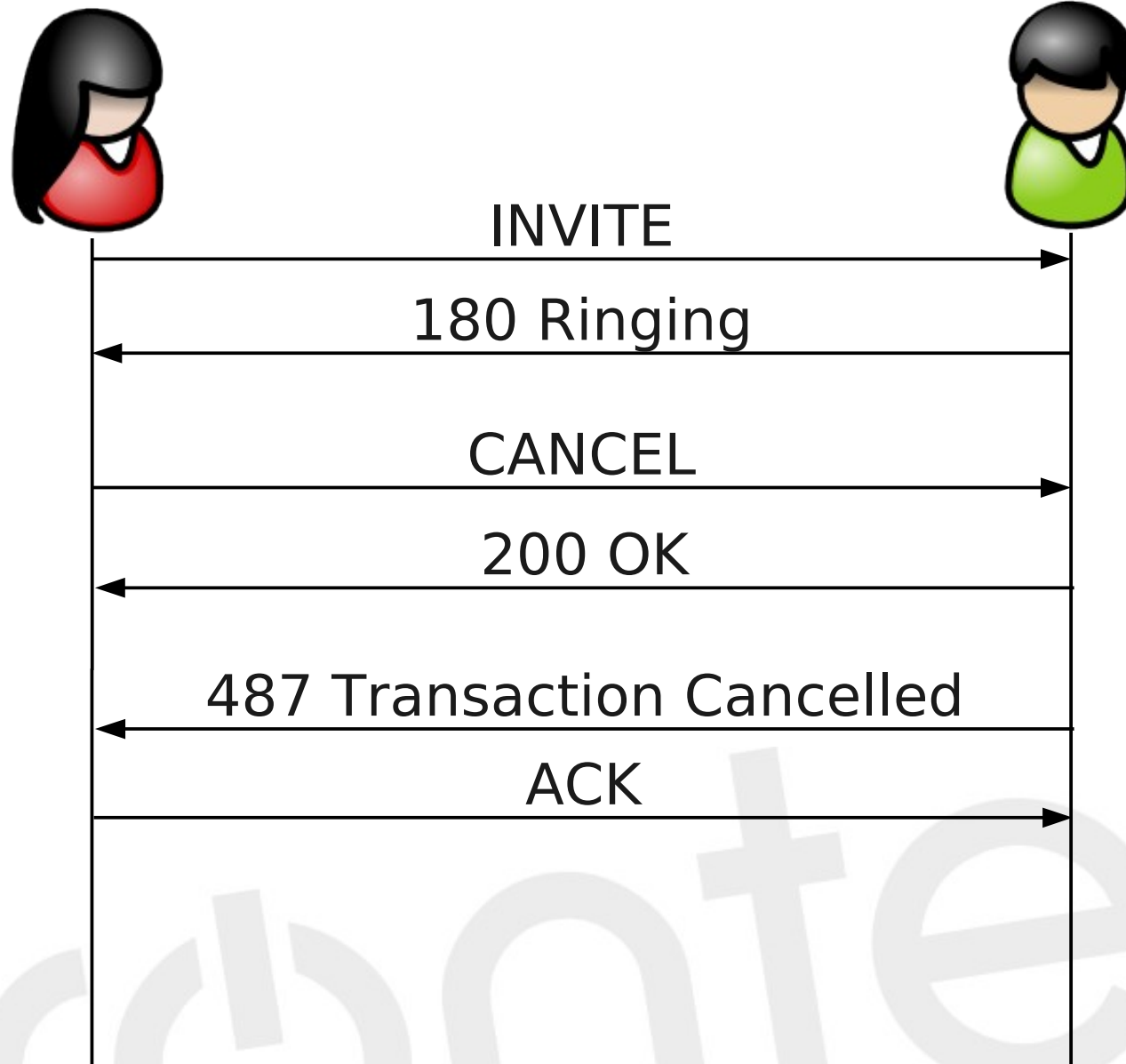
- INVITE es el único método que utiliza 3 way handshake.
- El resto de mensajes esperan una respuesta veloz, pero en el caso del INVITE, esta puede tardar.
- El UAC manda al UAS un ACK, indicando que ha recibido su respuesta.
- Aseguramos el correcto establecimiento de la sesión sobre un medio no fiable: UDP



Alice

Bob

CANCEL



Alice

Bob

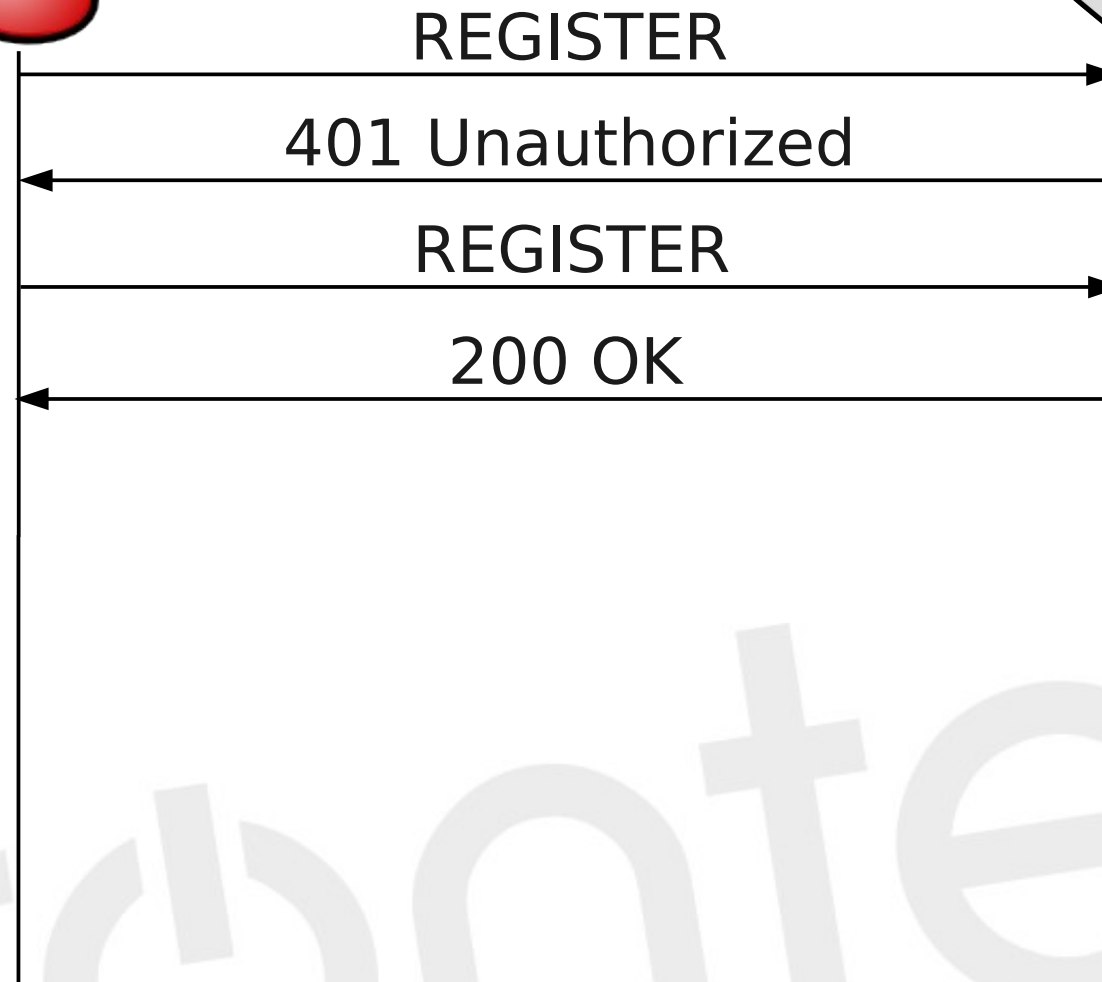
BYE



Alice

Servidor SIP

REGISTER



Alice

Bob **OPTIONS**



OPTIONS

200 OK

Cabeceras SIP

- From
 - Identifica al que origina una petición.
- Call-ID
 - Representa una relación entre 2 dispositivos SIP, relacionando un INVITE y todas las transacciones asociadas.
- Contact
 - Incluye una SIP URL, indicando donde se puede contactar con el usuario.
- To
 - Identifica al receptor de una petición.
- Vía
 - Contiene todos los proxys que han gestionado una petición.
 - Hace que las respuestas sigan el mismo camino que las peticiones

NB

Conceptos importantes: Transacción y Diálogo

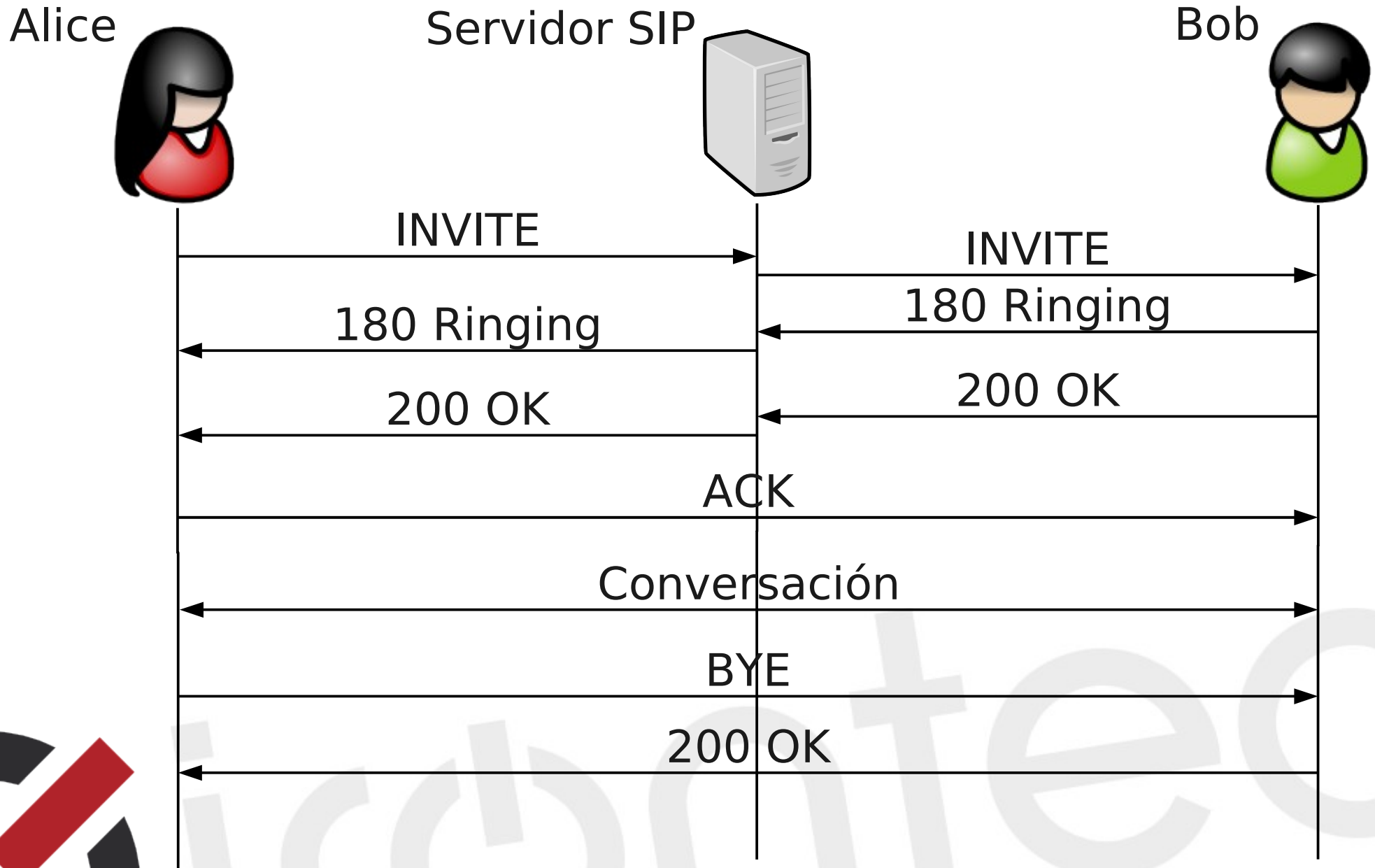
- Transacción
 - Una petición + respuesta, SI la respuesta es afirmativa (INVITE + 200 OK)
 - Una petición + respuesta negativa + ACK (INVITE + 404 Not Found + ACK)
 - Identificado unívocamente por el 'branch' de la cabecera Vía.
- Diálogo
 - Concepto de 'llamada'
 - Identificado unívocamente por el From tag, To tag y Call-ID.

Tipos de proxys SIP

- Stateful Proxy
 - Su ámbito es la transacción.
 - No entiende de diálogos, pero sí de transacciones.
- Stateless Proxy
 - No guardan ningún tipo de estado.

IMPORTANTE: Asterisk **NO** es un proxy SIP!!

Routing de mensajes SIP

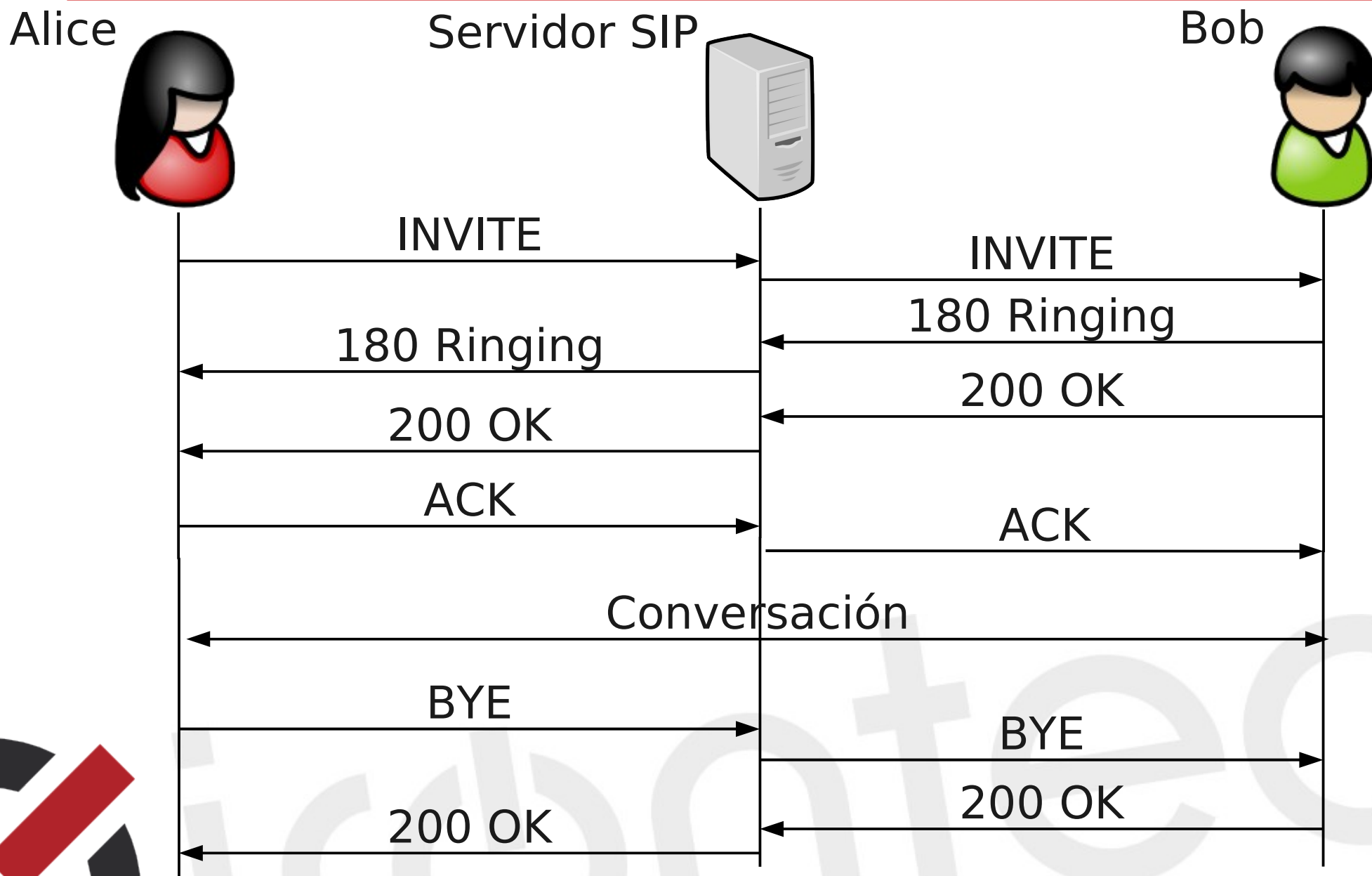


Routing de mensajes SIP (2)

- Después del 200 OK, Alice ya sabe donde esta Bob (Contact)
- Las transacciones siguientes (ACK y BYE-200 OK) van directamente de extremo a extremo.
- Podemos alterar este comportamiento con las cabeceras Record-Route y Route
 - Si queremos facturar, queremos estar al tanto de la señalización...



Routing de mensajes SIP (3)



Routing de mensajes SIP (4)

- Cada proxy que quiere quedarse 'en medio' añade una cabecera Record-Route al invite que pasa a través de él.
- Las cabeceras se mantienen y se envían de vuelta en la respuesta.
- Las siguientes transacciones se generan con la cabecera Route (en orden inverso que las Record-Route).
- En mensaje se envía al proxy que indica su primera cabecera Route y el proxy la elimina.

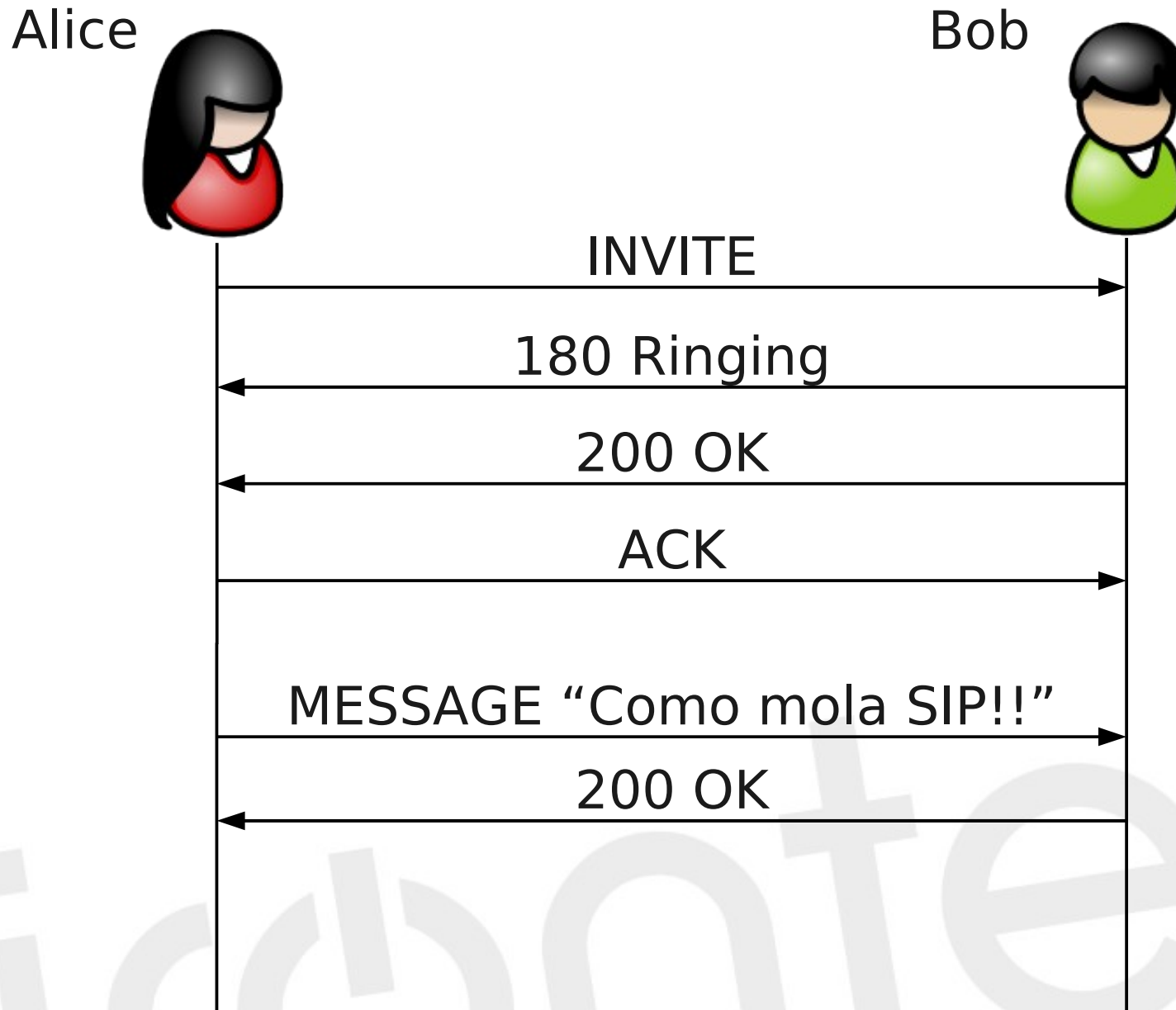


Extendiendo SIP

- Con lo visto hasta ahora, solo podemos hacer y recibir llamadas :-O
- PEEEEERO, SIP se diseñó para ser extensible, por lo que se le han añadido servicios mediante extensiones al protocolo.
 - Mensajería
 - Notificaciones Asíncronas de Eventos
 - Transferencia de sesiones
 - ...



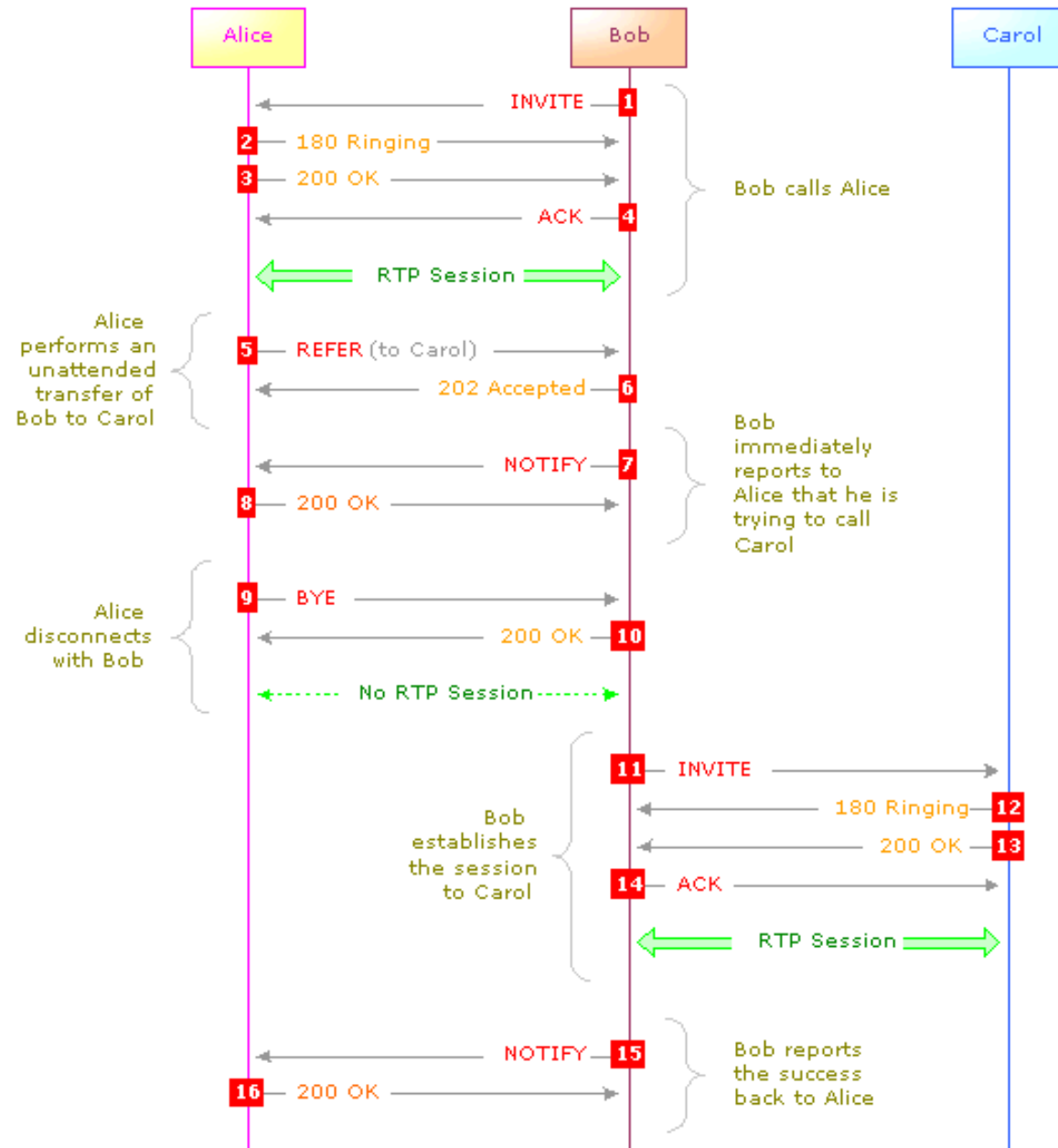
Mensajería Instantánea



Notificaciones Asíncronas de Eventos



Transferencia de sesiones



Ejercicio

Capturar y analizar trazas SIP llamando entre 2 terminales directamente, sin ningún proxy/b2bua entre ellos.

Herramientas necesarias:

- Softphone
- Ngrep

Uso de ngrep:

```
ngrep -d any -W byline -T -P '' port 5060
```

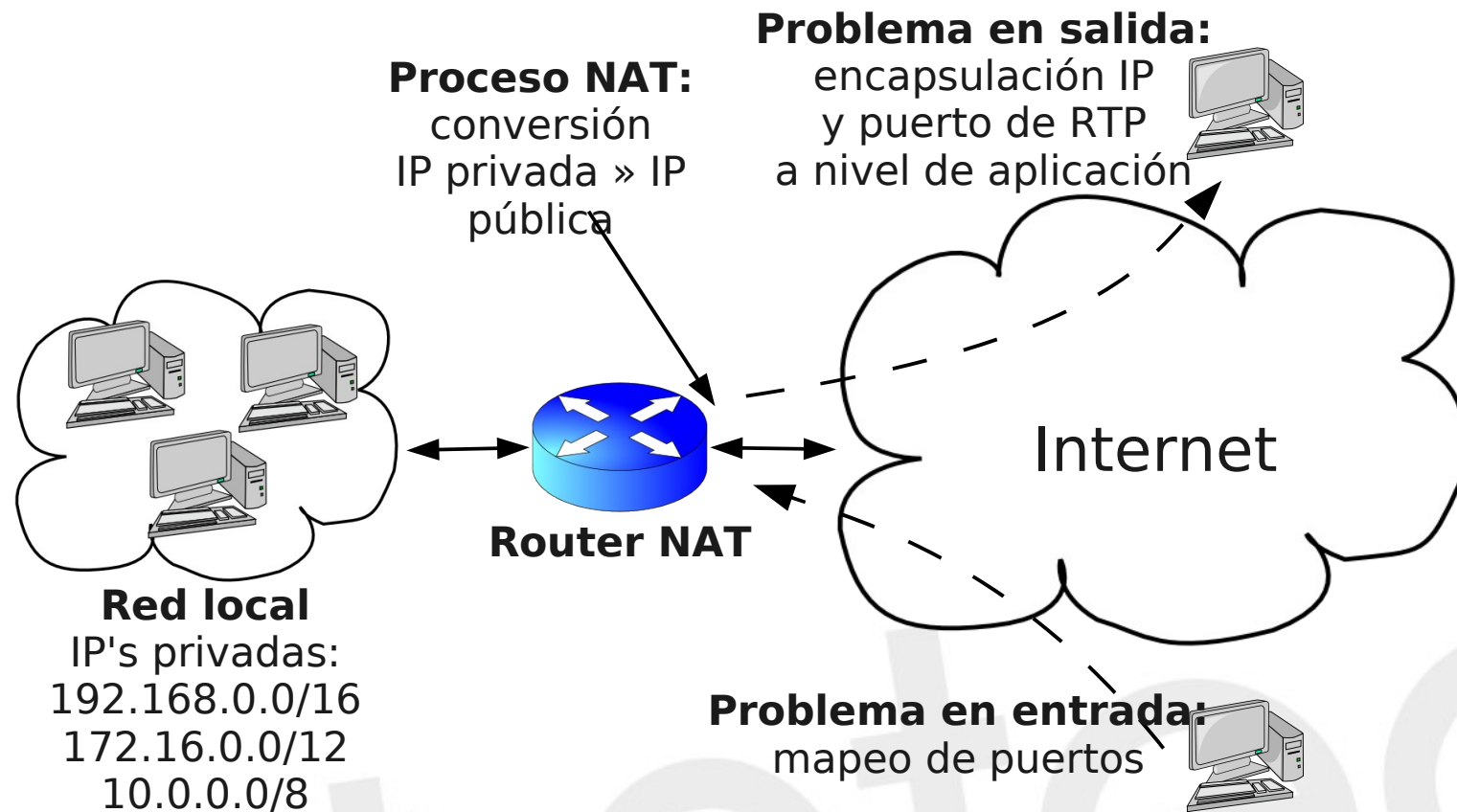
SIP y el NAT

ironotec
GNU/Linux

Problema del NAT

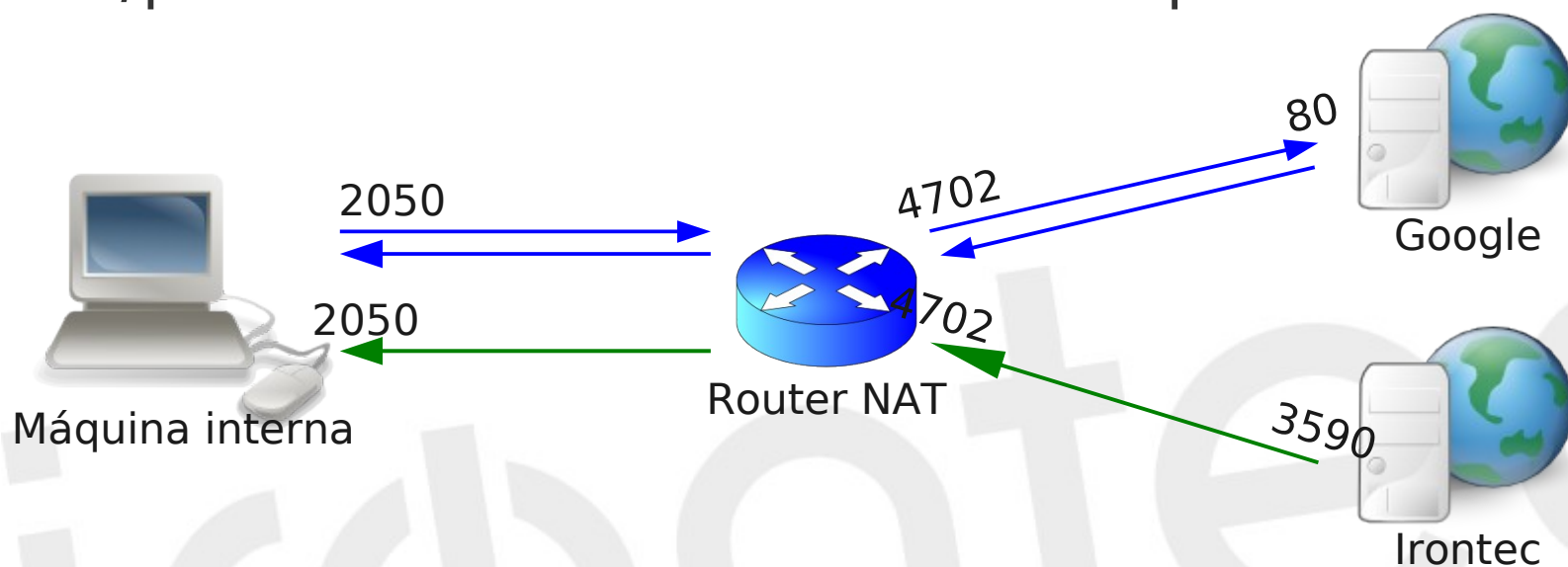
- El 'supuesto' agotamiento de los rangos de direcciones IP utilizables en Internet ha obligado a utilizar direcciones IP privadas dentro de las redes de empresas y usuarios domésticos.
- Un equipo IP para ser alcanzado en Internet debe utilizar una IP pública para sus comunicaciones. Es necesario por tanto 'enmascarar' la red interna en una o varias IPs públicas (Source NAT).
- El proceso de NAT no es nada sencillo: varios tipos de NAT, varios tipos de soluciones.
- Tesis Heinz Herlitz:
 - <http://www.uct.cl/biblioteca/tesis-on-line/heinz-herlitz/tesis.pdf>

Esquema Base



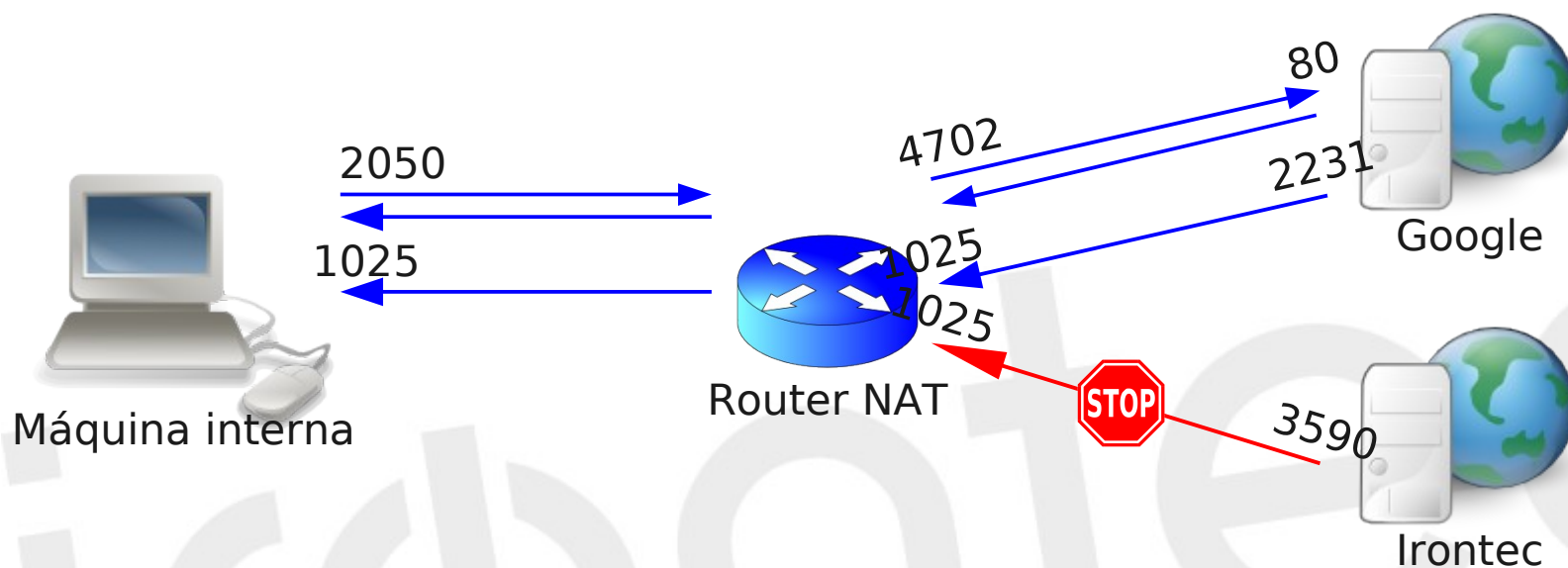
Tipos de NAT (I)

- **NAT full cone:** Todas las peticiones desde la misma IP/puerto de la LAN son mapeadas a la misma IP/puerto público. Cualquier máquina puede enviar paquetes a la máquina interna por esa IP/puerto mediante redirección de puertos.



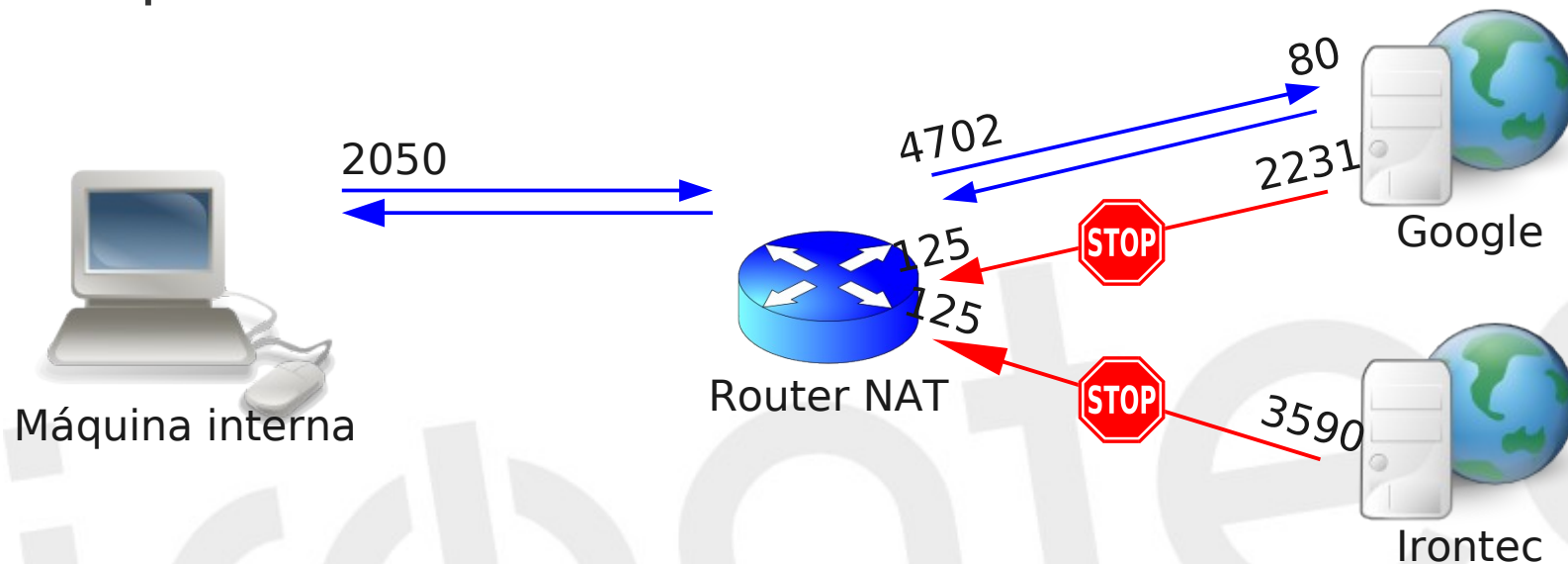
Tipos de NAT (II)

- **NAT restringido:** Lo mismo, pero una máquina externa con IP X puede enviar paquetes a la máquina interna sólo si ésta le ha enviado paquetes previamente. No importa el puerto.



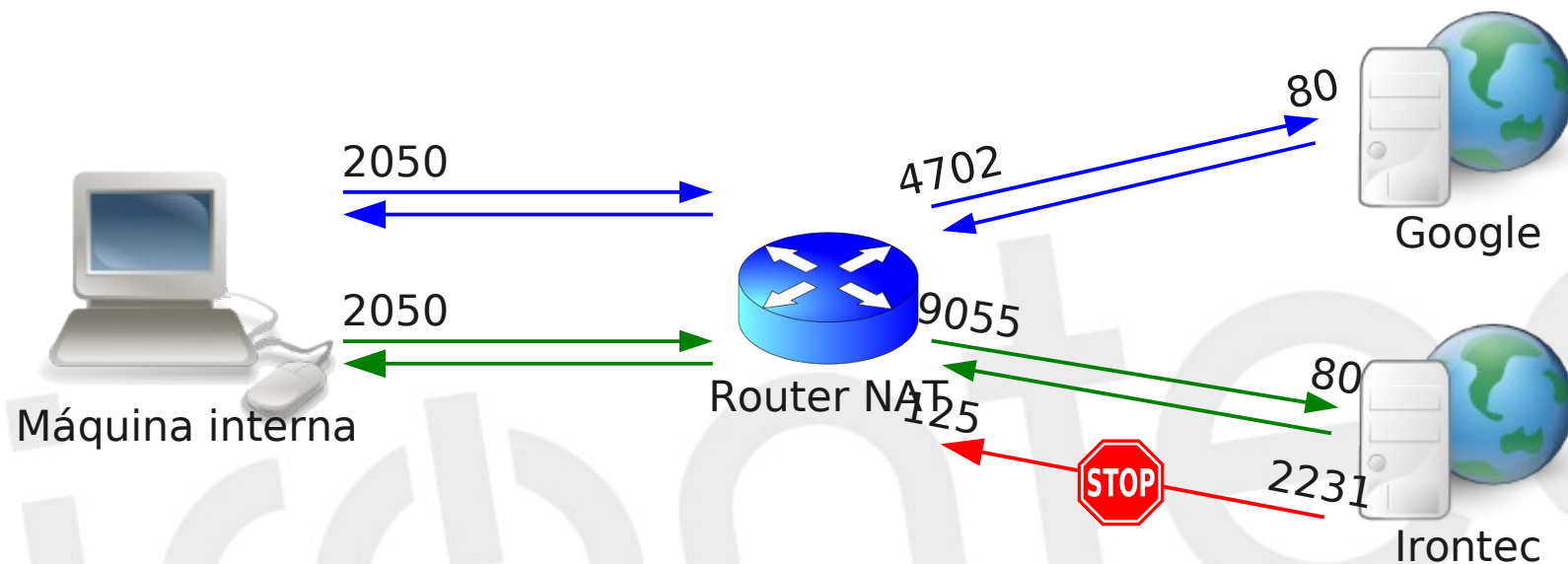
Tipos de NAT (III)

- **NAT puerto restringido:** Lo mismo que NAT restringido, pero la máquina externa con IP X y puerto P sólo puede enviar paquetes a la máquina interna si ésta le ha enviado previamente paquetes al puerto P.



Tipos de NAT (IV)

- **NAT simétrico:** Todas las peticiones desde la misma IP/puerto de la LAN a una IP/puerto externos específicos son mapeadas a la misma IP/puerto público. Si la máquina interna envía un paquete a una IP/puerto distintos el mapeo cambia. Por lo tanto, sólo la máquina externa que recibe un paquete puede devolver paquetes a la máquina interna.



Tipos de NAT (V)

- La clasificación anterior está abandonada hoy en día. Muchas implementaciones NAT oscilan entre varios de los tipos.
 - **Preservación de puerto:** Se mapea la misma IP/puerto externo para la misma IP/puerto interno. Si dos máquinas internas tratan de conectar con la misma IP/puerto externo, el puerto exterior mapeado a la segunda máquina se elige aleatoriamente. También se conoce como **NAT restricted cone**.

¿Cómo nos afecta el NAT?

- El paquete SIP que enviamos contiene Ips privadas:
 - Cabecera 'Via'
 - Cabecera 'Contact'
 - Campo 'c' en el SDP



NAT y SIP

INVITE sip:destino@mydomain.org SIP/2.0
Via: SIP/2.0/UDP **192.168.1.33:5060**;rport;branch=z9hG4bKjyofoqmp
Max-Forwards: 70
To: <sip:destino@mydomain.org>
From: "Iñaki" <sip:ibc@mydomain.org>;tag=nrrrx
Call-ID: xetazdjyktlpsfo@192.168.1.33
CSeq: 800 INVITE
Contact: <sip:ibc@**192.168.1.33:5060**>
Content-Type: application/sdp
Allow:
INVITE,ACK,BYE,CANCEL,OPTIONS,PRACK,REFER,NOTIFY,SUBSCRIBE,INFO,MESSAGE
Supported: replaces,norefersub,100rel
User-Agent: Twinkle/1.1
Content-Length: 312

v=0
o=ibc 1090098764 894503441 IN IP4 192.168.1.33
c=IN IP4 **192.168.1.33**
t=0 0
m=audio **8000** RTP/AVP 98 97 8 0 3 101

Soluciones para NAT

- Solución por parte del cliente:
 - Utilización de servidores STUN.
 - SIP ALG
- Soluciones de en los equipos de comunicaciones IP:
 - VPN
 - Mapeo de puertos
- Soluciones en los servidores SIP:
 - NAT helpers.

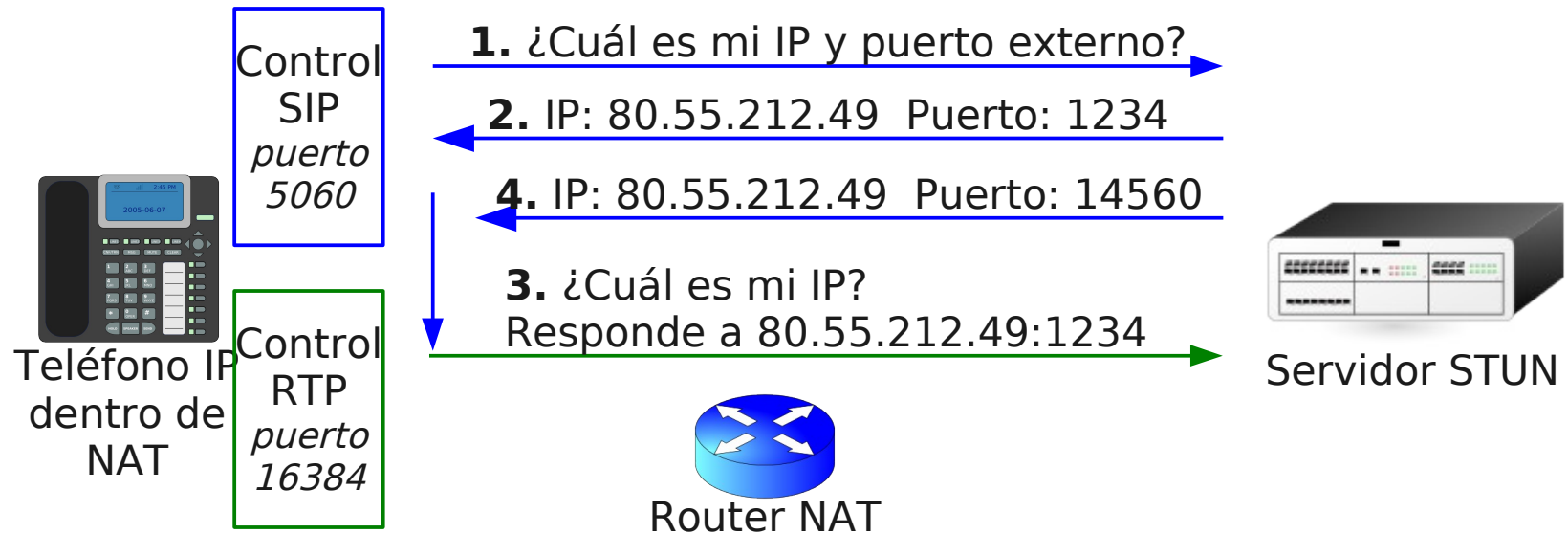
STUN: Simple Traversal of UDP through NATs

Teoría de Funcionamiento

- Protocolo de red que permite a clientes detrás de NAT averiguar su IP pública, tipo de NAT y puerto exterior.
- El cliente STUN solicita a un servidor STUN la IP y puerto por los que ha salido a Internet. En función de varios test contra el servidor STUN el cliente averigua el tipo de NAT en el que se encuentra.
- El servidor STUN dispone de dos IPS públicas.
- **No soluciona el problema del NAT simétrico.**
 - ¿Por qué?
- En VoIP se utiliza para facilitar la recepción de los datos de voz RTP (UDP).
- Servidores STUN públicos:
 - stun.fwd.net, stun.xten.com, ...

STUN: Simple Traversal of UDP through NATs

Ejemplo



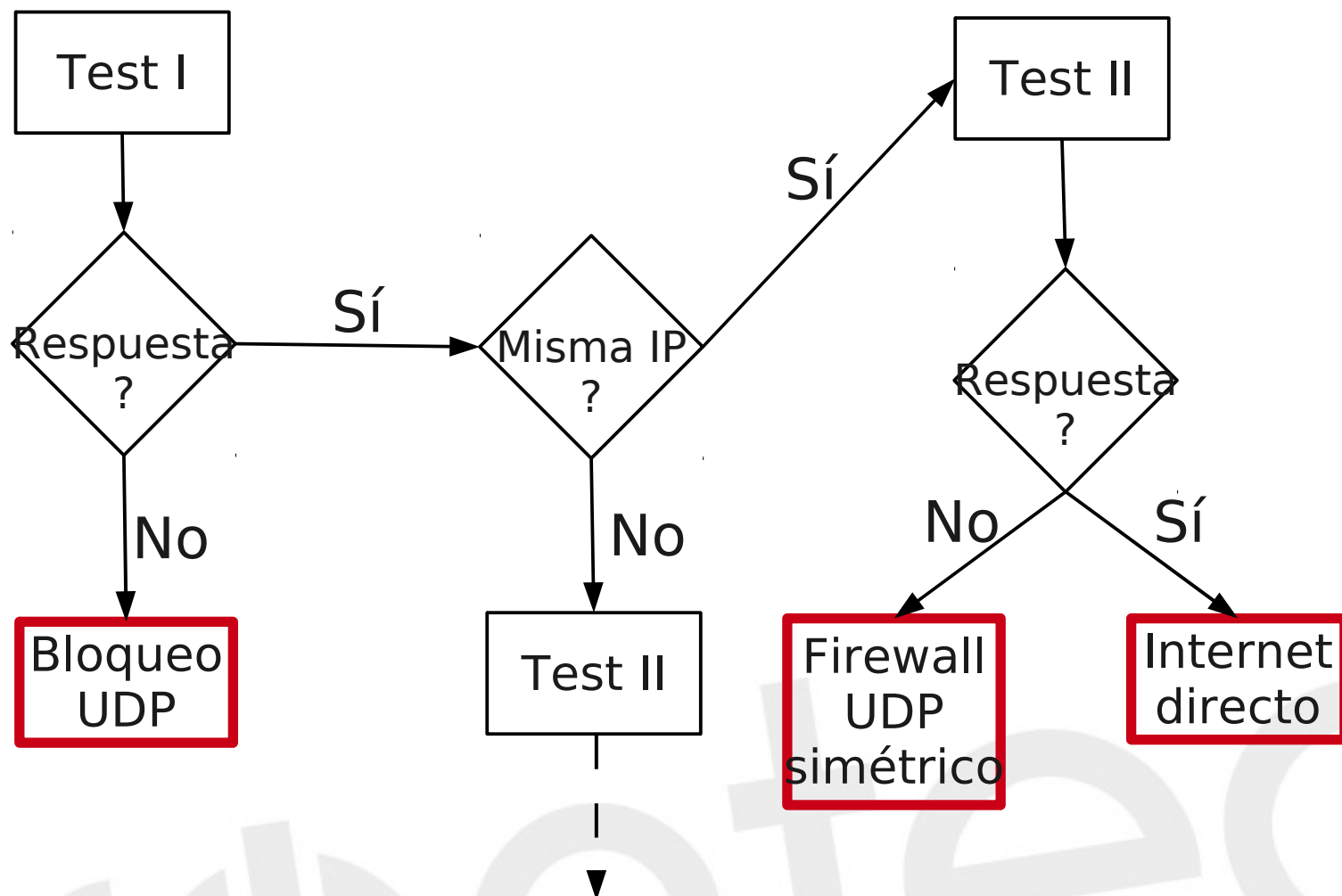
- El control SIP ya sabe qué IP y puerto encapsular en la negociación con el extremo para el canal RTP:
 - IP: 80.55.212.49
 - Puerto: 14560

STUN: Simple Traversal of UDP through NATs

Test cliente-servidor (I)

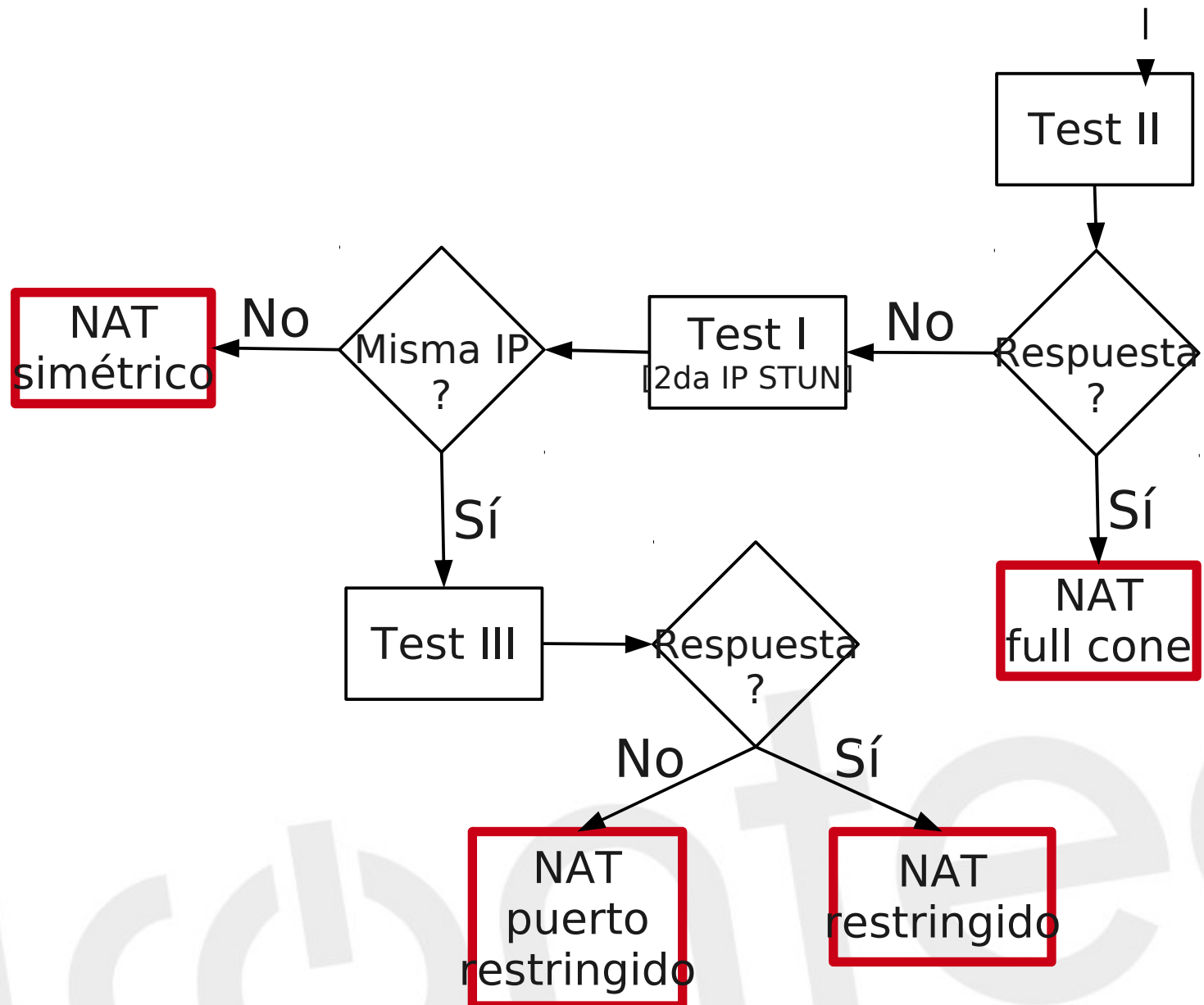
- Test I:
 - El cliente STUN solicita al servidor STUN (UDP port 3478) la IP y puerto exterior suyos (del cliente).
- Test II:
 - El cliente STUN repite la petición pero solicitando al servidor STUN que responda desde otra IP y puerto.
- Test III:
 - Igual que el Test II pero solicitando sólo que responda desde otro puerto.

STUN: Simple Traversal of UDP through NATs



continúa...

STUN: Simple Traversal of UDP through NATs



- SIP Application Level Gateway
- Implementado en routers de todas las gamas
 - Comtrend de Telefónica
 - SpeedTouch de Orange
 - ...
- El 99,99% de los ALG no funcionan
- Es necesario deshabilitarlo para poder utilizar VoIP
- <http://www.voip-info.org/wiki/view/Routers+SIP>

El protocolo IAX2

ironotec
GNU/Linux

Protocolo IAX2

- IAX2: Inter Asterisk eXchange
- Creado y estandarizado por la centralita Asterisk.
- Utiliza el puerto 4569 UDP.
- Características Principales:
 - Media y señalización por el mismo flujo de datos.
 - Trunking
 - Cifrado

Protocolo IAX2: Ventajas

- **NAT:** Al enviar tanto señalización como streaming por el mismo flujo de datos (flujo UDP), se evitan los problemas derivados del NAT. No es necesario abrir rangos de puertos para RTP.
- **Trunking:** Es posible enviar varias conversaciones por el mismo flujo, lo cual supone un importante ahorro de ancho de banda (overhead de la capas IP y transporte UDP).

Asterisk

ironotec
GNU/Linux

¿Qué es Asterisk?

- Software
 - Cumple todas las funcionalidades de una centralita tradicional y más.
 - Open Source.
 - **“Asterisk is an OpenSource Multiprotocol PBX”**
- Distintos tipos de uso
 - Para uso doméstico.
 - Uso empresarial
 - Operadores de telefonía IP.
 - etc...

- Inicialmente desarrollado por Mark Spencer (creador de GAIM).
- Tenía una empresa (Linux Support Services Inc.) y necesitaba una centralita.
- Decidió hacerla él mismo.
- Se juntó con Jim Nixon, originalmente soportaba las tarjetas de Zapata Telephony
- Liberó el código bajo licencia GPL.



- Viendo la evolución de Asterisk, la empresa cambió de nombre a Digium.
- Actualmente Mark no es el CEO, es el CTO.
- Adquisiciones y alianzas con diversas empresas para aumentar el “ecosistema”.



Versiones de Asterisk

- Principalmente hay 2 versiones
- La versión **Open Source**.
 - Es posible obtener soporte directo de Digium
- Asterisk Business Edition: edición comercial.
 - Se basa en la Open Source, pero se eliminan todos aquellos elementos susceptibles de causar problemas.
 - Fuertemente probada.
 - Al comprarla incluye soporte durante un año.
 - Licencias.
 - Va a ser utilizada solo para productos OEM.

Modelo de desarrollo de Asterisk

- Actualmente conviven 3 grandes ramas de desarrollo:
 - Asterisk 1.2
 - Solo se solucionan bugs de seguridad
 - Asterisk 1.4
 - Release 'congelada'
 - Se solucionan todo tipo de bugs que no necesiten cambios en la arquitectura
 - Nada de nuevas 'features'
 - Asterisk 1.6
 - Es donde tiene lugar todo el desarrollo actualmente

Modelo de desarrollo de Asterisk (2)

- Asterisk 1.6 admite todo tipo de nuevas 'features'
- Por cada nuevo 'mayor release' se crea un nuevo branch
 - 1.6.0, 1.6.1, 1.6.2, ...
- Cada una de estas ramas solo admite un cambio importante
- Se van a mantener 3 'point releases' de cada branch, y 3 branches al mismo tiempo
 - Cuando salga Asterisk 1.6.0.4 se deja de mantener la 1.6.0.0 (solo bugs de seguridad)
 - Cuando salga Asterisk 1.6.4.0 se dejarán de mantener Asterisk 1.6.0.X (solo bugs de seguridad)

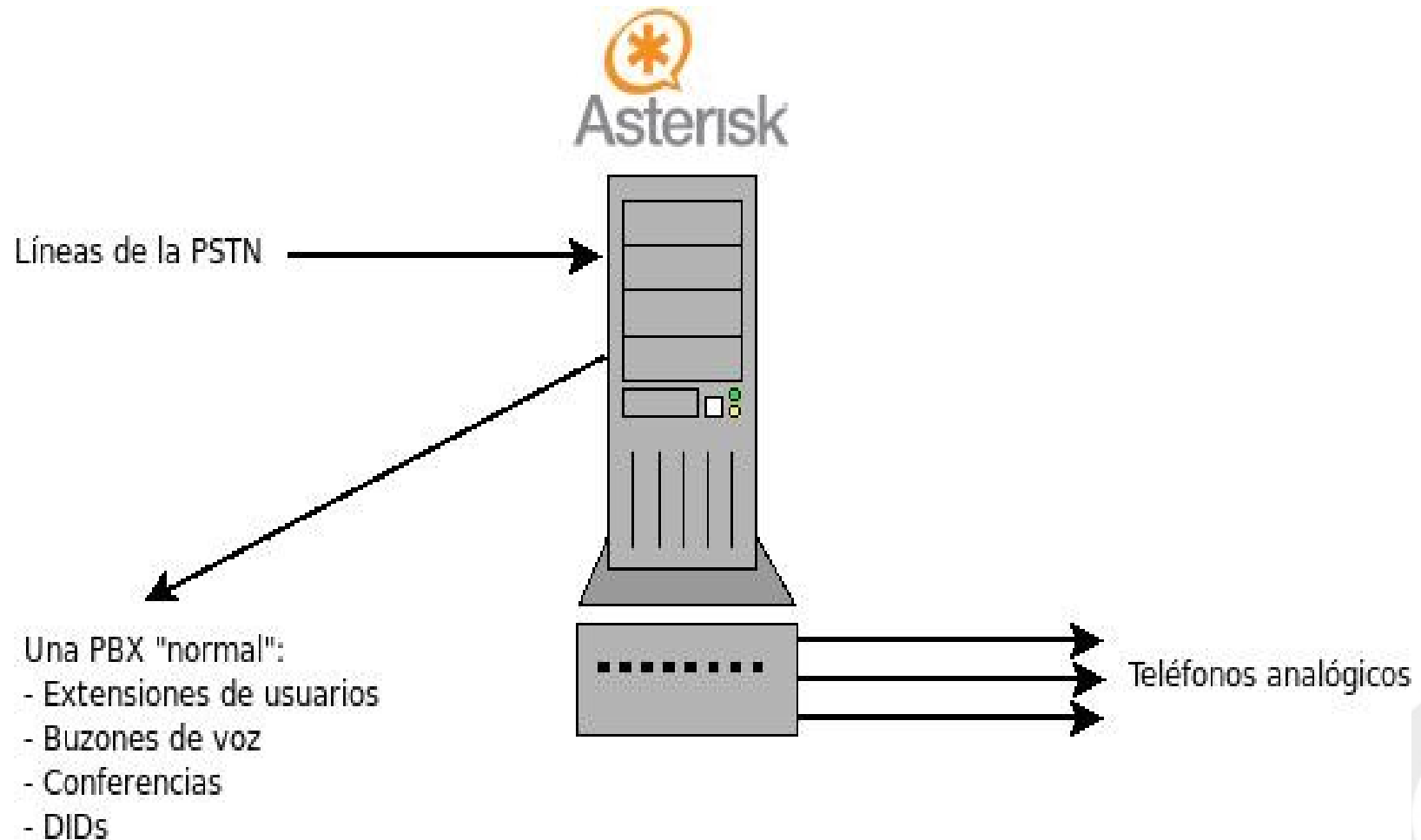
Distintos usos de Asterisk

ironotec
GNU/Linux

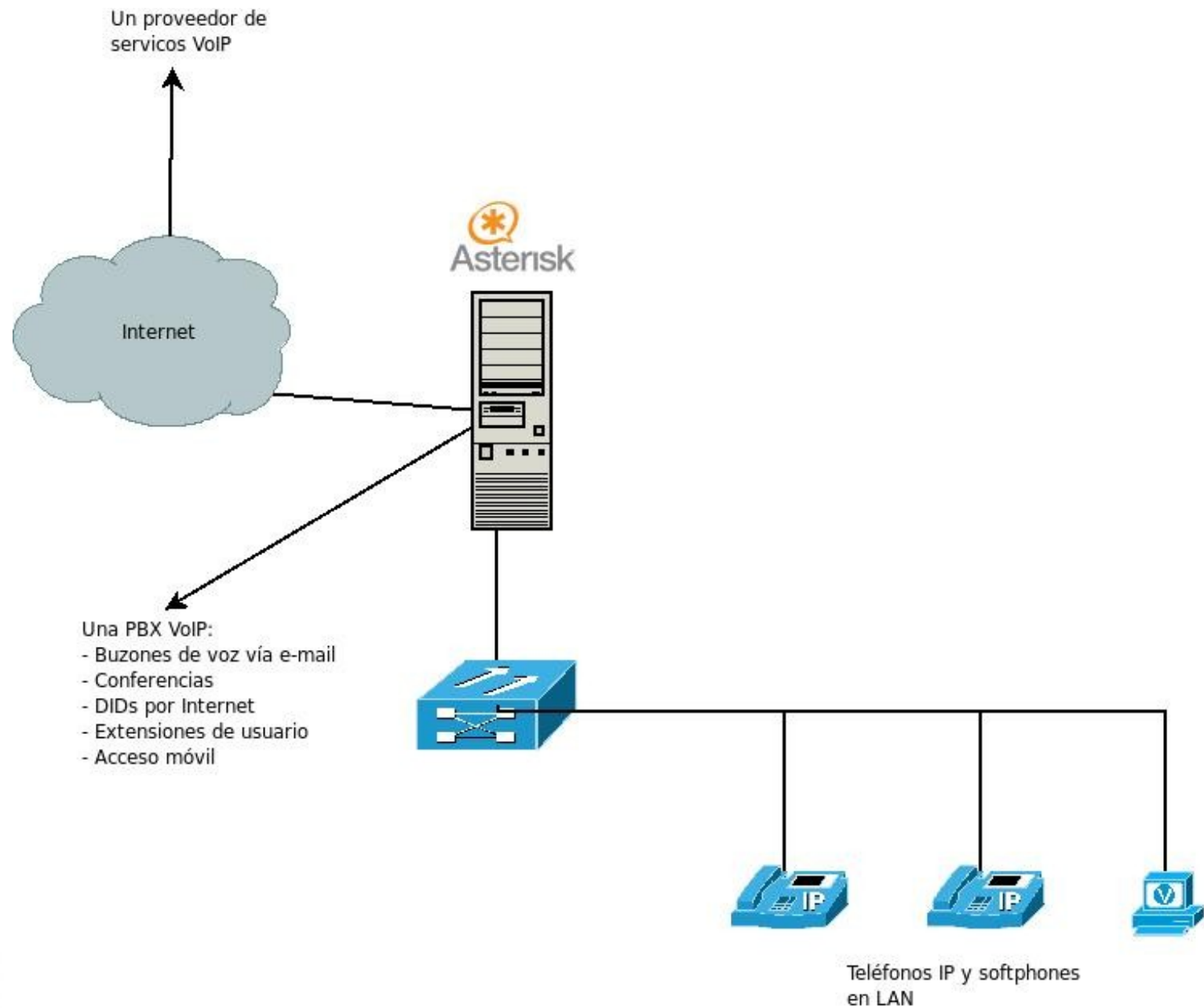
Funcionalidades de Asterisk

- Características típicas de centralita:
 - Transferencias
 - Música en espera
 - ...
- **Multiprotocolo y OpenSource!!**
- Extensiones, DIDs para usuarios.
- Buzones de voz, desvíos de llamada, follow-me, ...
- Colas y agentes.
- Menús IVR.
- Protocolos: SIP, IAX2, H.323, MGCP, ...
- PSTN: T1/E1, ISDN BRI, FXO/FXS.

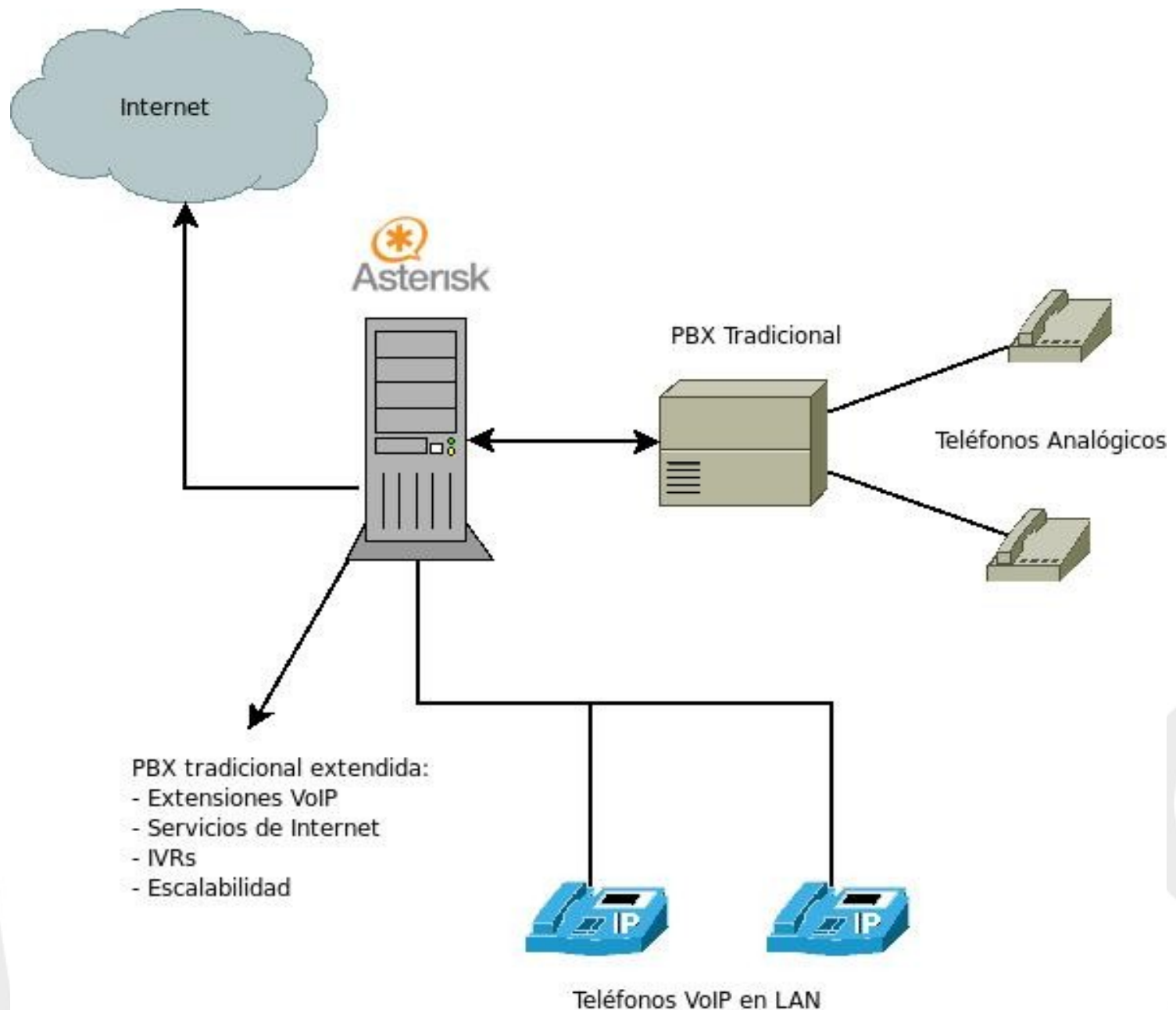
Asterisk como PBX "normal"



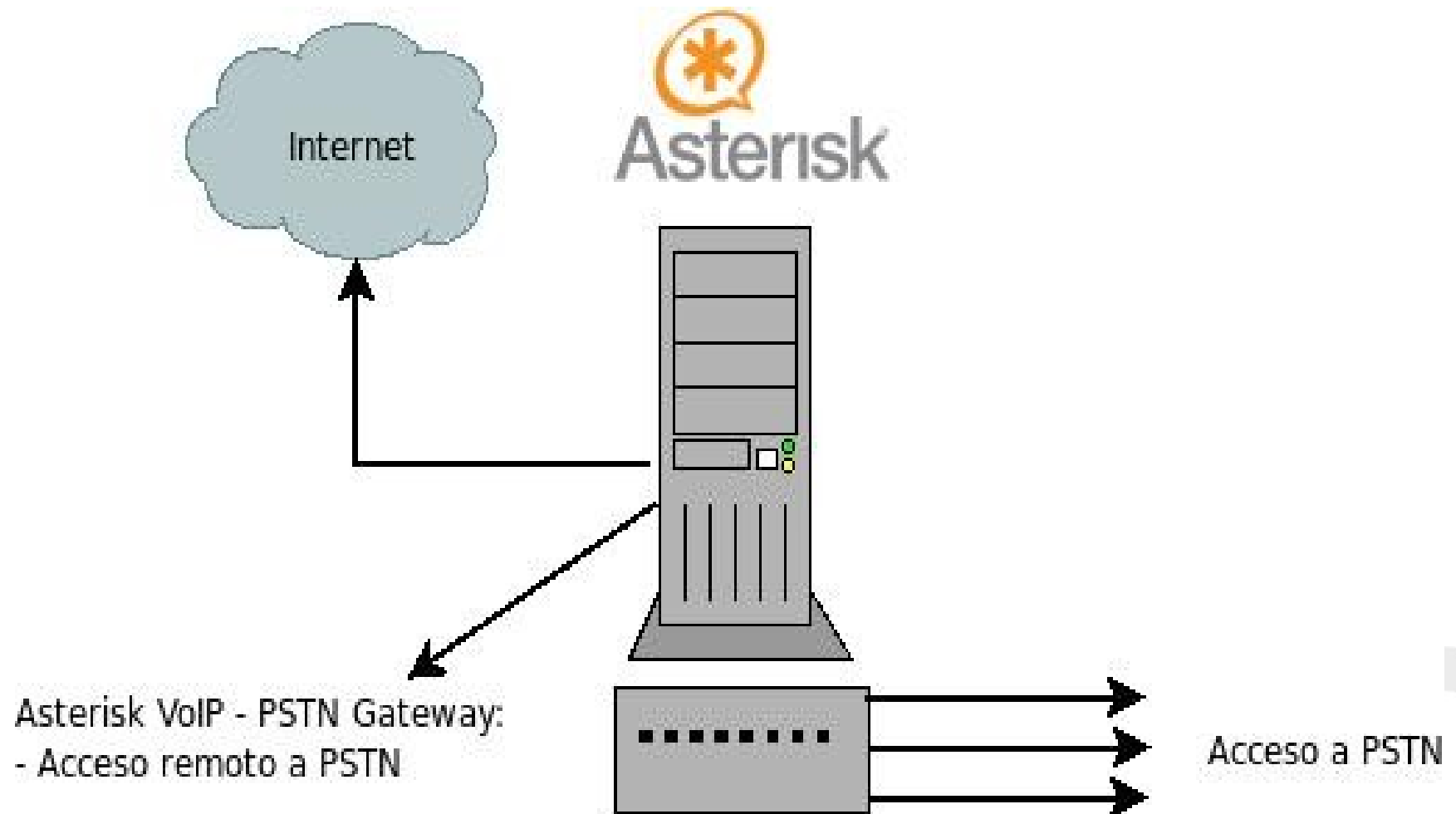
Una PBX VoIP



Complemento/ampliación de una PBX tradicional



Gateway VoIP - PSTN

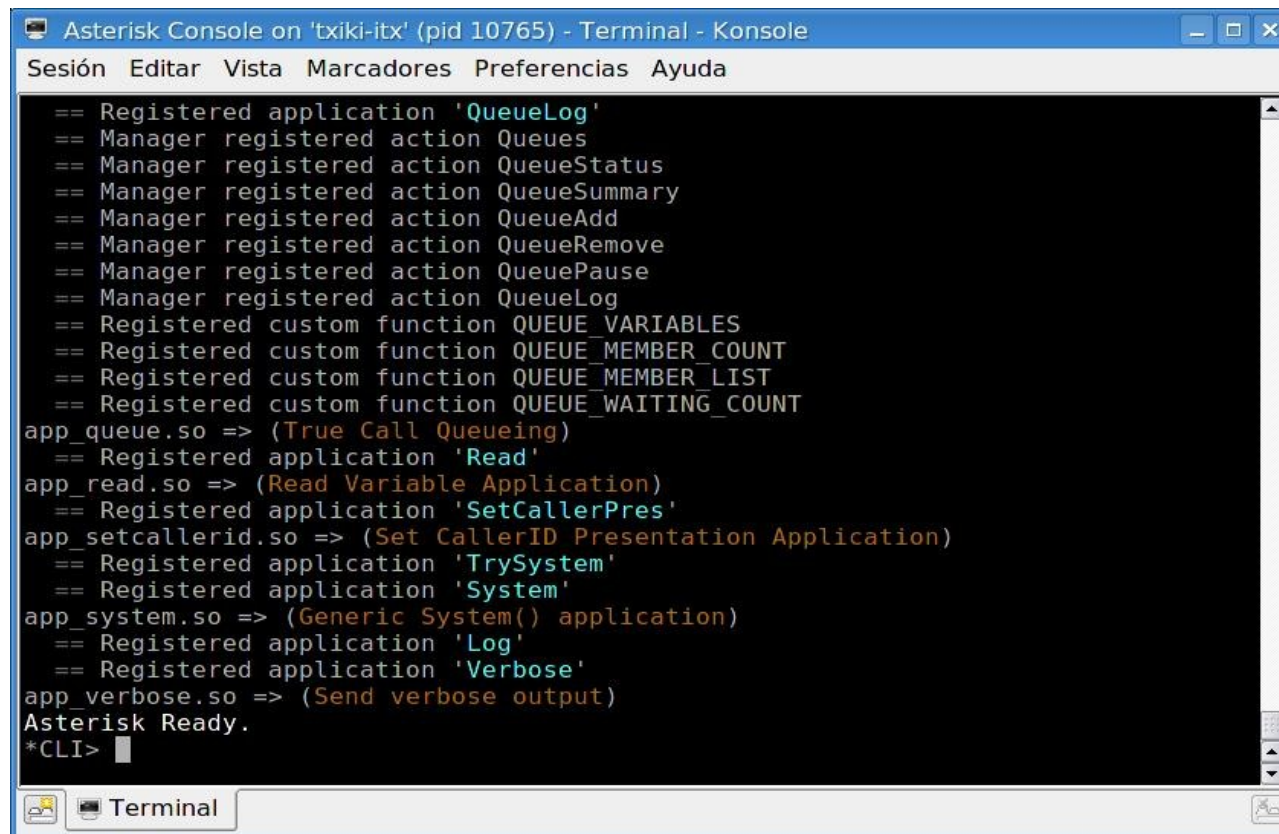


Asterisk “fácil-version”

- Distribuciones con Asterisk:
 - AsteriskNow: Basada en CentOS, incluye FreePBX o Asterisk-GUI.
 - TrixBox (antes Asterisk@Home): Basado en CentOS, incluye FreePBX.
- GUIs para Asterisk:
 - Asterisk-GUI: Desarrollada por Digium. Asterisk Appliance.
 - FreePBX: Desarrollada por terceros.



Asterisk “fácil-version” (2)



```
Asterisk Console on 'txiki-itx' (pid 10765) - Terminal - Konsole
Sesión  Editar  Vista  Marcadores  Preferencias  Ayuda

== Registered application 'QueueLog'
== Manager registered action Queues
== Manager registered action QueueStatus
== Manager registered action QueueSummary
== Manager registered action QueueAdd
== Manager registered action QueueRemove
== Manager registered action QueuePause
== Manager registered action QueueLog
== Registered custom function QUEUE_VARIABLES
== Registered custom function QUEUE_MEMBER_COUNT
== Registered custom function QUEUE_MEMBER_LIST
== Registered custom function QUEUE_WAITING_COUNT
app_queue.so => (True Call Queueing)
== Registered application 'Read'
app_read.so => (Read Variable Application)
== Registered application 'SetCallerPres'
app_setcallerid.so => (Set CallerID Presentation Application)
== Registered application 'TrySystem'
== Registered application 'System'
app_system.so => (Generic System() application)
== Registered application 'Log'
== Registered application 'Verbose'
app_verbose.so => (Send verbose output)
Asterisk Ready.
*CLI> 
```

Console roolz!!

Asterisk-GUI

The screenshot shows the Asterisk-GUI interface for configuring conference bridge extensions. The left sidebar contains a navigation menu with options: Home, Users, Conferencing (selected), Voicemail, Call Queues, Service Providers, Calling Rules, Incoming Calls, Voice Menus, Record a Menu, Active Channels, System Info, Backup, and Options. The main content area is titled "Conference Bridge Extensions Configuration" and includes a "Logout" link. It features a list of extensions on the left, a configuration form on the right, and an "Advanced" settings panel at the bottom right. The extensions list shows "6001 -- Conference Bridge" selected. The configuration form includes fields for Extension (6001), PIN Code (1234), and Administrator PIN Code (4355), along with checkboxes for "Play hold music for first caller", "Enable caller menu", and "Announce callers". The "Advanced" panel includes a "Room Override" field and checkboxes for "Record conference", "Quiet Mode", "Wait for marked user", and "Set marked user". A "Configuration saved!" message is displayed. At the bottom, there are "New", "Delete", "Save", and "Cancel" buttons. A copyright notice is visible in the footer.

digium|Asterisk

System Configuration | About Digium | Report a Bug | Help

Conference Bridge Extensions Configuration Logout

Extensions:

- 6000 -- Test User1
- 6001 -- Conference Bridge**
- 8500 -- Check Voicemail

Extension:

PIN Code:

Administrator PIN Code:

Play hold music for first caller ☒

Enable caller menu ☐

Announce callers ☒

Configuration saved!

Advanced

Room Override:

Record conference ☐

Quiet Mode ☒

Wait for marked user ☒

Set marked user ☐

MeetMe conference bridging allow quick, ad-hoc conferences with or without security.

Copyright ©2006-2007 Digium, Inc. Digium® and Asterisk® are registered trademarks of Digium, Inc. All Rights Reserved. Legal Information

freePBX administration - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://100.100.100.10/admin/config.php?display=extensions&tech=sip

freePBX™ • Setup • Tools • Reports • Panel • Recordings

Logged in: admin (logout) :: Setup

Module Admin	<h3>Add SIP Extension</h3> <p>Add Extension</p> <p>Demo <399></p> <hr/> <p>Add Extension</p> <p>Extension Number: <input type="text" value="299"/></p> <p>Display Name: <input type="text" value="Test"/></p> <hr/> <p>Extension Options</p> <p>Outbound CID: <input type="text"/></p> <p>Emergency CID: <input type="text"/></p> <p>Record Incoming: <input type="text" value="On Demand"/></p> <p>Record Outgoing: <input type="text" value="On Demand"/></p> <hr/> <p>Device Options</p> <p>secret <input type="text" value="1111"/></p> <p>dtmfmode <input type="text" value="dtc2833"/></p> <hr/> <p>VoiceMail & Directory: <input type="text" value="Disabled"/></p> <hr/> <p><input type="button" value="Submit"/></p>
Conferences	
On Hold Music	
Digital Receptionist	
Extensions	
Inbound Routes	
Outbound Routes	
Trunks	
General Settings	
Administrators	

Done

Asterisk: Terminología básica

ironotec

CNU/Linu

- PSTN: Public Switched Telephony Network.
- FXS: Foreign eXchange Station.
 - Lo que hay en nuestra pared.
 - **Alimenta** dispositivos.
 - Utiliza señalización FXO.
- FXO: Foreign eXchange Office.
 - El conector de nuestro teléfono donde entra la línea.
 - Recibe alimentación.
 - Utiliza señalización FXS.

Redes y protocolos

- LAN, WAN, ...
- Protocolos:
 - Capa de Transporte:
 - TCP
 - UDP
 - Capa de Aplicación:
 - SIP: Session Initiation Protocol
 - IAX2: Inter Asterisk eXchange

Arquitectura de Asterisk

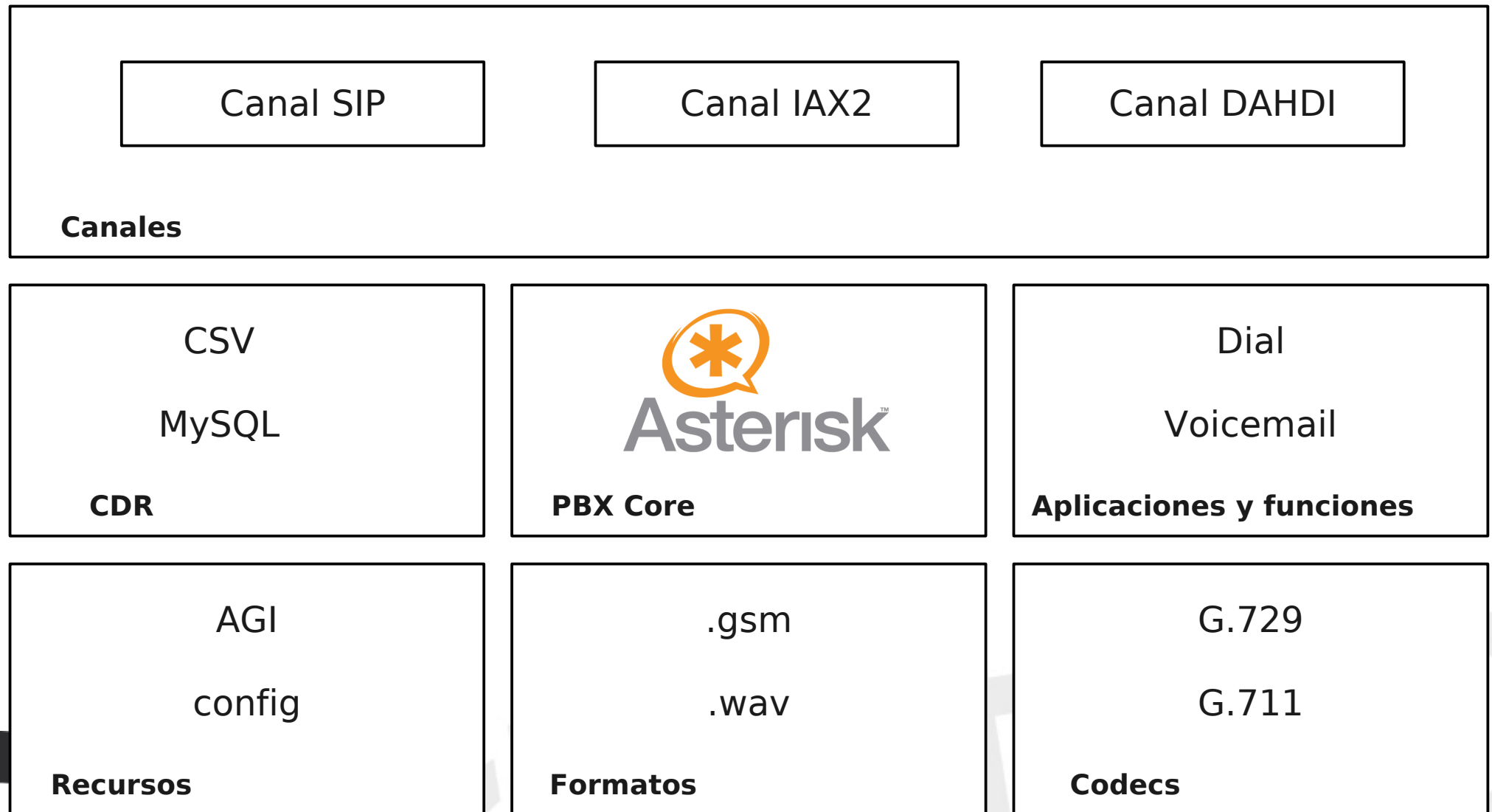
ironotec
GNU/Linux

Arquitectura de Asterisk

- Asterisk es muy modular.
- Nos permite ajustarlo a nuestras necesidades (escalabilidad).
 - Asterisk en dispositivos empujados.



Arquitectura de Asterisk (2)



Formatos

- Asterisk soporta multitud de formatos: wav, mp3, gsm, h.264, etc ...
- Los necesitamos para reproducir ficheros, p.e. en un IVR, Playback, Voicemail, ...



Llamadas en Asterisk



En una llamada hay 2 canales: el que origina la llamada y el que la recibe.

AstChannels: La magia

- Si asterisk es multiprotocolo, ¿como se realiza una llamada entre 2 dispositivos de distinta tecnología?
- **Hay 4 canales!!** 2 dependientes de la tecnología utilizada y 2 AstChannels.
- Asterisk crea los AstChannels para poder llevar a cabo el bridging.
 - Transferencias.
 - Parking.



Bridging



Instalación de Asterisk

ironotec

CNU/Linu

Asterisk PBX: Instalación

Requisitos Técnicos del sistema

- Requisitos: Dependen directamente de:
 - Llamadas concurrentes.
 - Conferencias y Aplicaciones complejas simultáneas.
 - Transcodificaciones necesarias (recodificación).
- Principalmente, Asterisk requiere microprocesador.
- Según Digium: Equipo Dual Intel Xeon 1.8 Ghz 1GB RAM soporta 60 llamadas concurrentes codificando con el codec G.729.
- Difícil determinar con exactitud, mejor apuntar alto para poder escalar.

Preparando la instalación

- Requisitos Hardware
 - PC
 - Tarjetas de telefonía
 - Café
- Requisitos Software
 - Debian GNU/Linux
 - <http://www.voip-info.org>
 - San Google



Asterisk PBX: Instalación

Instalación de dependencias

- Asterisk necesita para su correcta compilación y funcionamiento los siguientes paquetes:
 - Herramientas de compilación: gcc, make
 - Cabeceras de C: libncurses5-dev
 - Librerías SSL: libssl-dev
 - Headers de MySQL: libmysqlclient15-dev
 - Librerías opcionales: libnewt-dev
 - Cabeceras del Kernel: linux-headers-\$(uname -r)
- Además se recomiendan los siguientes paquetes:
 - openssh-server, vim, ...

Asterisk PBX: Instalación

Instalación de dependencias (II)

- En una línea:

```
# apt-get install build-essential linux-headers-$(uname -r) libncurses5-dev libssl-dev libmysqlclient15-dev libnewt-dev
```



Asterisk PBX: Instalación

Descarga del código fuente

- **Asterisk:** Núcleo (core) del sistema.
- **Asterisk-addons:** Software adicional que por motivos de licencias no ha sido incluido en el paquete principal.
- **Libpri:** Librería para gestionar señalización RDSI.
- **DAHDI-linux:** Drivers del Kernel para acceder a tarjetas de comunicaciones para líneas analógicas o digitales.
- **DAHDI-tools:** Herramientas para interactuar con los drivers de las tarjetas de comunicaciones.

Asterisk PBX: Instalación

Descarga del código fuente

- Descargamos los ficheros necesarios:

```
# wget  
http://downloads.asterisk.org/pub/telephony/asterisk/asteris  
k-1.6.0-current.tar.gz  
  
# wget  
http://downloads.asterisk.org/pub/telephony/asterisk/asteris  
k-addons-1.6.0-current.tar.gz  
  
# wget http://downloads.asterisk.org/pub/telephony/dahdi-  
linux/dahdi-linux-current.tar.gz  
  
# wget http://downloads.asterisk.org/pub/telephony/dahdi-  
tools/dahdi-tools-current.tar.gz  
  
# wget http://downloads.asterisk.org/pub/telephony/libpri/  
libpri-1.4-current.tar.gz
```

Compilando Asterisk

- Descomprimos todos los ficheros y comenzamos a compilar:

```
# for file in *.tar.gz; do tar zxvf $file; done
```

```
#cd dahdi-linux-*  
#make  
#make install
```

```
#cd libpri-1.4*  
#make  
#make install
```

```
#cd dahdi-tools-*  
#./configure  
#make menuselect  
#make  
#make install  
#make config
```

Compilando Asterisk

```
#cd asterisk-1.6*  
#./configure  
#make menuselect  
#make  
#make install  
#make samples  
#make config
```

```
#cd asterisk-addons-1.6*  
#./configure  
#make menuselect  
#make  
#make install  
#make samples
```



Estructura de directorios

- **/etc/asterisk:** Contiene los ficheros de configuración. Si al compilar ejecutamos “make samples”, tendremos ejemplos en este directorio.
- **/usr/lib/asterisk/modules:** Contiene los módulos de Asterisk que hemos compilado.
- **/var/lib/asterisk:** Contiene diferentes “librerías” de Asterisk.
- **/var/lib/asterisk/agi-bin:** Directorio para contener los AGI.
- **/var/spool/asterisk:** Directorio para archivos que genera Asterisk (voicemail, etc.)
- **/var/log/asterisk:** Aquí se guardan los log de Asterisk.

Ejecutando Asterisk

- Hay que ser **root**.
- Modificadores de arranque:
 - -c: Modo consola. Asterisk se inicia en primer plano.
 - -v: Verbose. Cuantas más uves se añadan más mensajes descriptivos veremos en la consola.
 - -d: Debug. Cuantas más des, más mensajes de debug.
 - -r: Remote console. Si hemos ejecutado Asterisk en segundo plano, nos permite conectarnos al CLI.

Ejecutando Asterisk (2)

- Para lanzar Asterisk en segundo plano:
`# asterisk`
- Para lanzar Asterisk en primer plano:
`# asterisk -vvvvvvvvvvvc`
- Para conectarnos a la consola si Asterisk esta en segundo plano:
`# asterisk -vvvvvvvvvvvr`



- El CLI (Command Line Interpreter) es la consola de Asterisk.
- Nos permite interactuar con Asterisk.
- Auto-completa los comandos pulsando TAB.
- Múltiples comandos nos resultan útiles para “ver lo que esta pasando”
*CLI>sip show channels
*CLI>core show application Dial

Reload/Restart

- Ambos sirven para reiniciar Asterisk, pero...
- Reload:
 - No todos los módulos se reconfiguran con reload.
 - No corta las llamadas en curso.
- Restart:
 - Se cortan las llamadas en curso.
 - Reconfigura todos los módulos.

Ficheros de Configuración

- Todos los ficheros de configuración de Asterisk tienen una estructura muy similar:

```
[general]  
param=valor  
...
```

```
[seccion]  
param=valor  
...
```

Configuración de canales SIP

ironotec
GNU/Linux

Configuración para canales SIP

- La instalación crea ficheros de ejemplo con la sintaxis bastante comentada a modo de guía.

sip.conf

- En este fichero se definen:
 - Variables generales de SIP.
 - Clientes SIP.
 - Servidores SIP.

sip.conf: Sección General

- En primer lugar existe la sección [general], donde se definen variables globales y aspectos por defecto para todos los canales SIP.
- La sintaxis es la siguiente:

```
[general]  
variable1=valor1  
variable2=valor2
```


sip.conf: Sección General

- Las variables generales más importantes son:
 - **allow y disallow**: indican los codecs permitidos / no permitidos.
 - **dtmfmode**: permite especificar el método por el cual se enviarán los tonos DTMF valores posibles: rfc2833, INFO, inband.
 - **externip**: Dirección Pública del servidor Asterisk.
 - **context**: Contexto por defecto donde entraran las llamadas entrantes por SIP.
 - **bindaddr**: Dirección IP en la que se escucha.
 - **bindport**: Puerto en el que escuchar (5060).

sip.conf: Clientes y Servidores

- En sip.conf se definen tanto los clientes que se conectarán a Asterisk, como los proveedores que se utilizarán para encaminar llamadas. Conceptualmente, se distinguen
 - user: Envía llamadas a Asterisk
 - peer: Recibe llamadas de Asterisk (proveedor).
 - friend: Recibe y Envía llamadas (usuario).
- La syntaxis para definir un friend o un peer es:

```
[nombre]  
type = friend / peer  
variable = valor  
viarable2 = valor
```

sip.conf: Clientes y Servidores

- Las variables más importantes que deben ser configuradas inicialmente son:
 - **type:** peer / friend
 - **context:** Contexto donde entraran las llamadas generadas.
 - **nat:** Indica si el usuario o peer se encuentran tras NAT.
 - **host:** IP remota o dynamic.
 - **username:** nombre de usuario.
 - **secret:** contraseña de acceso.
 - **allow y disallow:** Configuraciones de codecs específicas para cada friend/peer.
 - **qualify:** Evalúa el estado del extremo SIP para conocer su accesibilidad y latencia.

sip.conf: Ejemplo: usuario

- Vamos a declarar de forma básica dos clientes en el fichero sip.conf:

```
[jon]  
type = friend  
secret = jon  
disallow=all  
allow = alaw  
context = desde-usuarios  
host=dynamic
```

```
[leire]  
type = friend  
secret = leire  
disallow = all  
allow = alaw  
context = desde-usuarios  
host=dynamic
```

sip.conf: Ejemplo: peer

- Vamos a configurar un proveedor en sip.conf:

```
[sarenet]  
type=peer  
host = 1.2.3.4  
disallow = all  
allow = g729  
allow = alaw  
fromuser= 100  
fromdomain = 1.2.3.4  
secret=ghost
```

sip.conf: Verificación de la configuración

- Mediante el comando *module reload* en el CLI de Asterisk, le indicamos que recargue la configuración. Aunque es posible recargar la configuración SIP de forma independiente: *sip reload*
- Una vez recargada, podemos comprobar los usuarios y peers que tenemos definidos haciendo: *sip show peers*



Dialplan: El corazón de Asterisk

ironotec

CNU/Linu

El Dialplan

- Como una “tabla de enrutado”.
- Cada número que marca una extensión va al dialplan, y ahí decidimos que hacer.
- Se organiza por contextos, extensiones y prioridades.

exten => extensión, prioridad, aplicación



El Dialplan (2)

- La “**extensión**” puede ser cualquier cosa, no solo números:
exten => saul,1,Dial(SIP/saghul)
- Se utilizan “patrones de marcado” para las extensiones (ino vamos a meter todos los números del mundo!)
 - _ : Comienzo del patrón.
 - X: Cualquier dígito del 0 al 9.
 - Z: Cualquier dígito del 1 al 9.
 - N: Cualquier dígito del 2 al 9.
 - [12-4]: Cualquier dígito entre los corchetes, 1,2,3,4.
 - .(punto): Cualquier cosa.

El Dialplan (3)

- Ejemplos de patrones:
 - Llamadas nacionales (empiezan por 8 o 9):
_[89]XXXXXXXX
 - Llamadas a móviles (empiezan por 6):
_6XXXXXXXX
 - Llamadas Internacionales:
_00X.
 - Llamadas a números especiales SIN números de pago (806, 803, 906, 903)
_[89]0[0-2457-9]XXXXXXXX

El Dialplan (4)

- Las **prioridades** sirven para decidir **el orden de las acciones** al entrar en el dialplan.
- Pueden utilizarse números:
exten => 1234,1,Answer()
exten => 1234,2,Playback(demo-congrats)
- O la prioridad “n”, que va sumando 1 automáticamente:
exten => 1234,1,Answer()
exten => 1234,n,Playback(demo-congrats)
- Si utilizamos números es fácil saltar:
exten => 123,1,Goto(1234,1)
- Si usamos la prioridad “n” podemos hacer “marcas” en el dialplan, para luego poder saltar a ellas:
exten => 1234,n(mi-kosa),...
- Y podemos hacer:
exten => 123,1,Goto(1234,mi-kosa)

El Dialplan (5)

- Las extensiones se agrupan en **contextos**.
- Cada dispositivo SIP, IAX o entrada de DAHDI se define para un contexto.
- Un dispositivo solo puede “llamar” a los números que tiene definidos en el contexto en el que esta.
- En el ejemplo 1, solo podíamos marcar los números definidos en el contexto “default”.
- Un contexto puede incluir a otro:
[micontexto]
include => default
- Ahora aunque mi dispositivo este definido en el contexto “micontexto” tiene acceso a todo el contexto “default”.

El Dialplan (6)

- Hay “extensiones especiales”, que no son para que nadie marque:
 - s: Extensión start. Cuando entramos en un contexto SIN extensión. Macros, IVRs, entrantes DAHDI.
 - h: Extensión hangup. Se llega a esta extensión al finalizar la llamada.
 - i: Extensión inválida. Opciones incorrectas en IVRs.
 - T: Timeout absoluto en Dial.
 - t: Timeout en IVRs
 - fax: Detección de fax en canales DAHDI.

- Ya hemos visto algunas: Dial, Goto, ...
- Se ejecutan directamente desde el dialplan.
- Para una lista completa de las aplicaciones disponibles:
*CLI>core show applications
- Las aplicaciones *en general* hacen algo con un canal.

Variables

- Asterisk tiene distintos tipos de variables
 - Globales: Afectan a todos los canales
 - De canal: Solo afectan al canal actual
 - Del entorno (entorno UNIX)
- Una lista completa de las variables:
<http://www.voip-info.org/wiki-Asterisk+variables>
- Las variables se pueden “recortar”:
 - `${variable:desplazamiento:longitud}` por ejemplo:
 - `${variable}=saghul -> ${variable:1} => aghul`
`${variable:0:2}=sa`
- Se soportan expresiones básicas, encerrándolas entre corchetes:
`exten => 1234,1,Set(kosa=${6*5})`
- Variables globales:
 - `exten => 1234,1,Set(GLOBAL(kosa)=algo)`

Variables de canal definidas automáticamente

- Listado de variables más importantes:
 - `${CALLERID}`: Caller ID actual, nombre y número.
 - `${CONTEXT}`: Contexto actual.
 - `${EXTEN}`: Extensión actual.
 - `${CHANNEL}`: Canal actual.
 - `${DIALSTATUS}`: Estado de la llamada.
 - `${DATETIME}`: Hora actual.
- Una aplicación útil para ver el contenido es NoOp:
 - `NoOp (${VARIABLE})`
 - Nos mostrará en el CLI el valor de la variable.

Funciones

- Se utilizan en el dialplan, pero dentro de las aplicaciones, sirven para trabajar con datos:
 - LEN: Devuelve la longitud de una cadena
`Noop(LEN(kosa)) //devuelve 4`
 - CALLERID: Fija o lee el valor del callerid:
`Set(numero=CALLERID(number))`
- Para obtener una lista completa de funciones podemos ejecutar:

`*CLI>core show functions`

- En general NO actúan sobre un canal y pueden ir en cualquier sitio en el que pueda ir una variable.

La aplicación Dial

- La aplicación Dial permite realizar una llamada a un dispositivo. Su formato más sencillo es: Dial(Tecnología/dispositivo,tiemout,opciones)
- Donde opciones puede ser:
 - t: Permitir que el usuario **llamado** transfiera la llamada.
 - T: Permitir que el **llamante** transfiera la llamada.
 - r: Generar un tono de ring artificial.
 - w: Permite al usuario llamado grabar la llamada.
 - W: Permite que el llamante grabe la llamada.

La aplicación Dial (2)

- Tras un Dial correcto (se contesta la llamada), se termina la ejecución de acciones en el dialplan y se pasa a la extensión h (si existe):

exten => 1234,1,Dial(SIP/saghul)

exten => 1234,n,NoOp(Algo ha ido mal...)

- La variable DIALSTATUS contiene información acerca del último Dial realizado. Puede tomar los siguientes valores, entre otros: BUSY, NOANSWER, CHANUNAVAIL, ...
- Nos puede servir para hacer distintas cosas dependiendo del estado de la última llamada.

La aplicación Dial (3)

Más información sobre Dial:

- `core show application dial` (en el CLI de Asterisk)
- <http://www.voip-info.org/wiki-Asterisk+cmd+Dial>
- <http://www.voip-info.org/wiki/view/Asterisk+variable+DIALSTATUS>



La aplicación Goto

- Permite realizar saltos dentro del dialplan.
- Se pueden realizar saltos en base a:
 - Prioridad o etiqueta.
 - Extensión y prioridad (o etiqueta).
 - Contexto, extensión y prioridad (o etiqueta).
- Por ejemplo:

exten => 1234,1,Goto(servicios,1234,1)



Playback

- Esta aplicación reproduce un fichero de audio (o vídeo) al usuario.
- Sintaxis
Playback(nombre-del-fichero)

- Reproduce el sonido de entrada por el canal de salida.
- Muy útil para comprobar si hay audio bidireccional, a la hora de depurar problemas...

- Receta:
 - Crear un contexto “internas” donde se indica como llamar a las extensiones internas.
 - Crear un contexto “servicios” con los servicios de eco y demo.
 - Crear el contexto “telf-internos” que incluya los 2 contextos anteriores y hacer que los dispositivos SIP utilicen este contexto.
 - Incluir el contexto “internas” en el contexto default, para que si permitieramos llamadas entrantes, solo pudieran llamar a los teléfonos, pero no usar los servicios.

Ejercicio (solución)

extensions.conf

```
[default]
include => internas

[servicios]
exten => 400,1,Answer()
exten => 400,n,Playback(beep)
exten => 400,n,Echo()
exten => 401,1,Answer()
exten => 401,n,Playback(demo-congrats)
exten => 401,n,Hangup

[internas]
exten => 200,1,Dial(SIP/softphone,45,Tt)
exten => 200,n,Goto(200-${DIALSTATUS},1)
exten => 200-BUSY,1,Busy(5)
exten => _200-.,1,Congestion(5)

exten => 201,1,Dial(SIP/hardphone,45,Tt)
exten => 201,n,Goto(201-${DIALSTATUS},1)
exten => 201-BUSY,1,Busy(5)
exten => _201-.,1,Congestion(5)

[telf-internos]
include => internas
include => servicios
```

Ejercicio (solución) (2)

```
[general]          sip.conf  
context=default
```

```
[softphone]  
type=friend  
secret=1234  
context=telf-internos  
host=dynamic  
disallow=all  
allow=alaw
```

```
[hardphone]  
type=friend  
secret=1234  
context=telf-internos  
host=dynamic  
disallow=all  
allow=alaw
```

Aplicaciones del Dialplan

ironotec

CNU/Linu

Asterisk PBX: Aplicaciones en el dialplan

Aplicaciones Generales

- Las aplicaciones generales más importantes son:
 - Wait (n)
 - Espera n segundos, ignorando los dígitos marcados durante.
 - WaitExten (n)
 - Espera n segundos, pero gestionando los dígitos marcados.
 - MusicOnHold(clase, n)
 - Reproduce música en espera durante n segundos.
 - NoOp (mensaje)
 - Imprime el mensaje en el CLI

Asterisk PBX: Aplicaciones en el dialplan

Gestión de llamadas

- Los comandos de gestión de llamadas más importantes:
 - Answer()
 - Acepta la llamada entrante por el canal.
 - Busy()
 - Envía la señal de ocupado al origen.
 - Congestion()
 - Envía la señal de congestión al origen.
 - Hangup()
 - Cuelga la llamada.
 - Ringing()
 - Envía la señal de tono de llamada.

Asterisk PBX: Aplicaciones en el dialplan

Control de flujo

- Algunos comandos de control de flujo y temporización:
 - Goto (contexto, extension, prioridad)
 - Salta al contexto, extensión y prioridad del argumento.
 - GotoIf (expresión ? prioridad1 : prioridad2)
 - Salta a la prioridad1 si la expresión es verdadera.
 - Salta a la prioridad2 si la expresión es falsa.
 - GotoIfTime(<times>|<weekdays>|<mdays>|<months>? [[context]|exten|]priority)

Asterisk PBX: Aplicaciones en el dialplan

Reproducción de sonidos

- Algunas aplicaciones para la reproducción:
 - PlayBack (fichero)
 - Reproduce el fichero, continua la ejecución cuando finaliza.
 - Background (fichero)
 - Reproduce el fichero, pero continua la ejecución inmediatamente.
 - SayDigits (dígitos)
 - Reproduce los dígitos
 - PlayTones (tonos)
 - Reproduce los tonos indicados

Asterisk PBX: Expresiones

Syntaxis de las Expresiones

- Es posible utilizar expresiones en las llamadas a aplicaciones (principalmente: Gotof)
- Syntaxis:

`$(expr1 operador expr2)`

- Operadores Lógicos: |(or) , &(AND)
- Operadores de Comparación: =, !=, <, >, <=, >=
- Operadores Aritméticos: +, -, *, /, %
- [...], Ejemplos:

`exten => 1,1,Set(total=$(1 + 1))`

**`exten => 1,2,GotoIf($[${CALLERID(num)}=123456]?
10:20)`**

Buzones de voz

ironotec

CNU/Lin

- Asterisk dispone de un completo sistema de buzones de voz gestionable mediante 2 aplicaciones:
 - VoicemailMain: Menú interactivo desde el que podemos escuchar los mensajes, grabar nuestros mensajes personalizados, cambiar la contraseña, ...
 - Voicemail: Aplicación para dejar un mensaje en el buzón de un usuario.
- Se configuran en el fichero **voicemail.conf**

Estructura del fichero **voicemail.conf**

[general]

Opciones generales

...

[zonemessages]

Definición de formatos de hora por zona

...

[contexto de buzones]

Buzones

...

voicemail.conf – parámetros importantes

- Language: Idioma del sistema de buzones.
- Format: formato en el que se guardarán los mensajes de voz.
- Attach: Indica si se enviará el fichero de audio como un adjunto en el email de notificación.

Definición de un buzón:

- **Número => clave, nombre, email**
- 1234 => 1234, Saúl Ibarra, saul@irontec.com

Macros

ironotec
GNU/Linux

- Son contextos con un comportamiento distinto.
- Admiten parámetros.
- Empiezan **siempre** con la extensión “s”.
- Son como subprogramas, se utilizan para no repetir código en el dialplan.
- Ejemplo de macro que muestra en el CLI el parámetro que se le pasa:

```
[macro-prueba]  
exten => s,1,Answer()  
exten => s,n,Playback(demo-congrats)  
exten => s,n,Noop(${ARG1})  
exten => s,n,Hangup
```

- Para llamar a la macro:
exten => 1234,1,Macro(prueba,\${variable})

- Todos los argumentos son accesibles mediante ARG1, ARG2, ARG3, ...
- Variables especiales
 - MACRO_EXTEN: Extensión desde la que se llamó a la macro. EXTEN contendría 's'!
 - MACRO_CONTEXT: Contexto desde el que se llamó a la macro. CONTEXT contendría el nombre de la macro!
 - MACRO_PRIORITY: Prioridad desde la que se llamó a la macro. PRIORITY contendría la prioridad dentro de la macro!
- Las macros finalizan al quedarse sin más prioridades para ejecutar
 - Vuelven al contexto desde el que fueron llamadas
 - MacroExit provoca una salida inmediata

Ejercicio

- Diseñar una macro para llamar a los usuarios tal que si esta ocupado le mandaremos al buzón dando un mensaje de ocupado y si no responde le mandaremos al buzón dando un mensaje de no disponible.
- Si hay algún error (ni BUSY ni NOANSWER) le damos tonos de ocupado.



Ejercicio

```
[macro-prueba]
exten => s,1,Dial(${ARG1},30,Tt)
exten => s,n,Goto(s-${DIALSTATUS},1)
exten => s-BUSY,1,VoiceMail(${MACRO_EXTEN},b)
exten => s-BUSY,n,Hangup
exten => s-NOANSWER,1,VoiceMail(${MACRO_EXTEN},u)
exten => s-NOANSWER,n,Hangup
exten => _s-.,1,Answer()
exten => _s-.,n,Playtones(busy)
exten => _s-.,n,Busy(5)
```



IVRs

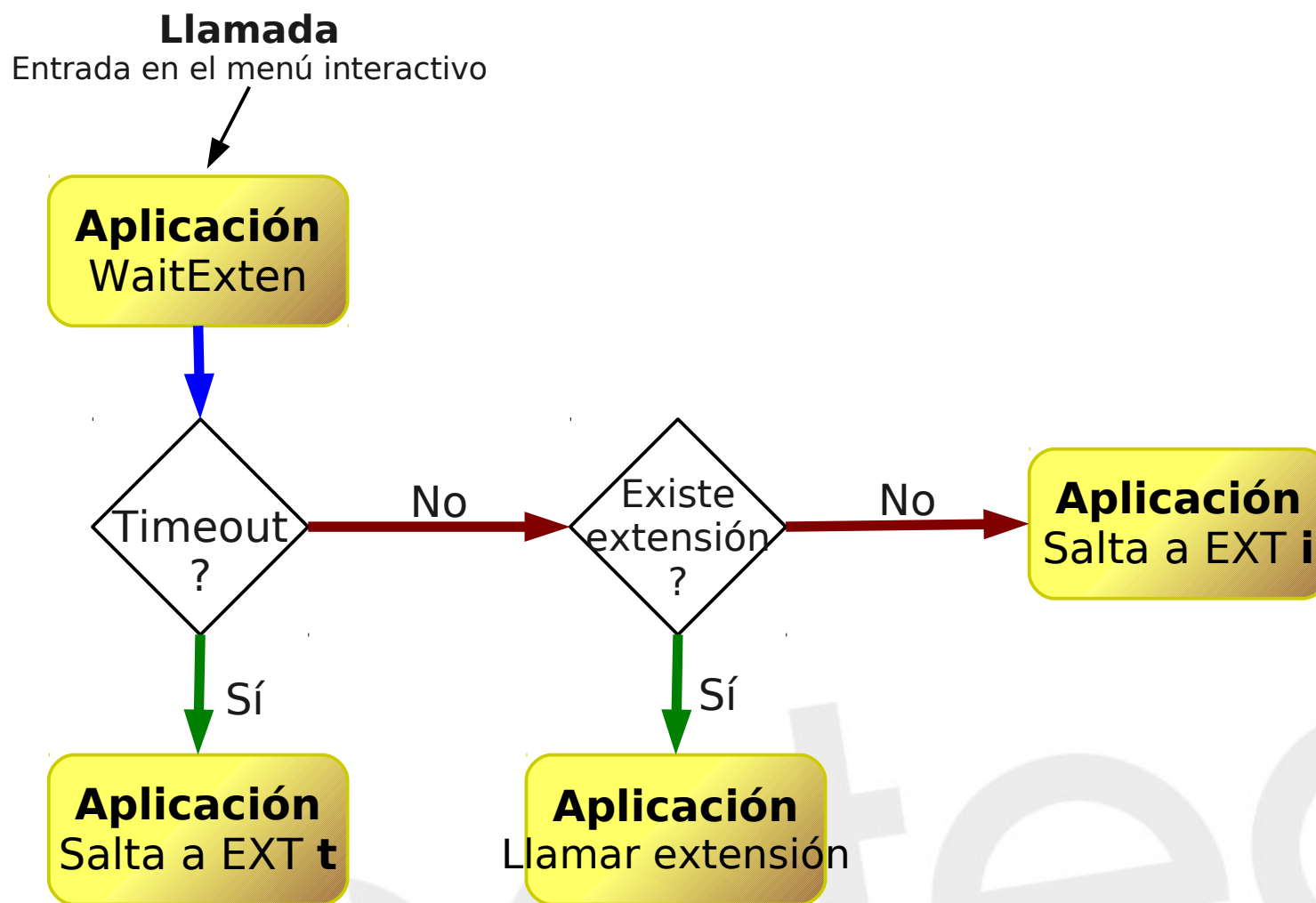
ironotec
GNU/Linux

- Un IVR (Interactive Voice Response) es un menú con el que el usuario puede interactuar mediante pulsaciones DTMF.
- Tradicionalmente comienza con la extensión “s”.
- Se comprueba la hora para decidir la siguiente acción.
- Se reproduce un mensaje de bienvenida y se esperan las pulsaciones del usuario.
- Se controlan las extensiones “t” e “i”, por si el usuario no pulsa nada en el tiempo fijado o realiza una pulsación incorrecta.

- Para que no se entre en un bucle infinito, se suelen fijar 2 tipos de retardo: tiempo inter-digito y el tiempo de respuesta total.

Set(TIMEOUT(digit)=3)

Set(TIMEOUT(response)=9)



- Diseñar un IVR con las siguientes características:
 - Al pulsar 1 llame al softphone
 - Al pulsar 2 llame al hardphone
 - Al pulsar 3 reproduzca la locución de los monos
 - Al pulsar 4 reproduzca música en espera de manera infinita
 - Al pulsar 5 dejemos un mensaje en el buzón del softphone

```
[ivr]
```

```
exten => s,1,Answer
```

```
exten => s,n,Set(TIMEOUT(digit)=3)
```

```
exten => s,n,Set(TIMEOUT(response)=6)
```

```
exten => s,n,Background(bienvenida)
```

```
exten => s,n,WaitExten(3)
```

```
exten => 1,1,Dial(SIP/softphone)
```

```
exten => 2,1,Dial(SIP/hardphone)
```

```
exten => 3,1,Playback(tt-monkeys)
```

```
exten => 3,n,Hangup
```

```
exten => 4,1,MusicOnHold()
```

```
exten => 5,1,VoiceMail(1234,s)
```

Dialplan avanzado

ironotec

- Asterisk incorpora una BD tipo Berkley DB v2.
- En ella el propio Asterisk guarda diversos valores:
 - Registros SIP, IAX.
 - Información sobre colas.
- Nosotros también podemos guardar información en ella y consultarla desde el dialplan.
- Se organiza en familias, y dentro de una familia puedes tener diversas claves, y para cada clave un solo valor.

- Ejemplo desde el CLI:

```
*CLI> database put mifamilia miclave mivalor  
Updated database successfully
```

```
*CLI> database show mifamilia  
/mifamilia/miclave : mivalor
```

```
*CLI> database put mifamilia miclave mivalor2  
Updated database successfully
```

```
*CLI> database show mifamilia  
/mifamilia/miclave : mivalor2
```

- Desde el dialplan:
 - Set(var=\${DB(familia/clave)})
 - Set(DB(familia/clave)=\${var})

Ejercicio

- Desarrollar una macro que implemente DND (Do Not Disturb)
- Si la extensión a la que llamamos tiene el DND activado reproducimos un mensaje de “extensión no disponible”.
- Si no lo tiene activado, le llamamos.
- Si no contesta o esta ocupado, le mandamos al buzón.
- Pistas:
 - Función DB_EXISTS.
 - Aplicación Gotof.
 - Variable MACRO_EXTEN.

Ejercicio

```
[macro-llamar]
```

```
exten => s,1,Gotolf($[${DB_EXISTS(DND/${MACRO_EXTEN})} = 0]?s,llamar)
```

```
exten => s,n,Playback(vm-extension)
```

```
exten => s,n,Playback(vm-isunavail)
```

```
exten => s,n,Hangup
```

```
exten => s,n(llamar),Dial(${ARG1},45,Tt)
```

```
exten => s,n,Goto(s-${DIALSTATUS},1)
```

```
exten => s-BUSY,1,VoiceMail(${MACRO_EXTEN}|b)
```

```
exten => s-BUSY,n,Hangup
```

```
exten => s-NOANSWER,1,VoiceMail(${MACRO_EXTEN}|u)
```

```
exten => s-NOANSWER,n,Hangup
```

```
exten => _s-.,1,Answer()
```

```
exten => _s-.,n,Playtones(busy)
```

```
exten => _s-.,n,Busy(5)
```



Ejercicio

- Ampliar la macro anterior para que implemente CF (Call Forward)
- Primero se comprueba el DND, y luego el CF.
- El desvío puede realizarse a cualquier número que los teléfonos internos puedan marcar (contexto telf-internos)
- Pistas:
 - Función DB_EXISTS.
 - Variable DB_RESULT.



Ejercicio

```
[macro-llamar]
exten => s,1,Gotolf([$${DB_EXISTS(DND/${MACRO_EXTEN})} = 0]?s,comp-cf)
exten => s,n,Playback(vm-extension)
exten => s,n,Playback(vm-isunavail)
exten => s,n,Hangup
exten => s,n(comp-cf),Gotolf([$${DB_EXISTS(CF/${MACRO_EXTEN})} = 0]?s,llamar)
exten => s,n,Goto(telf-internos,${DB_RESULT},1)
exten => s,n(llamar),Dial(${ARG1},45,Tt)
exten => s,n,Goto(s-${DIALSTATUS},1)
exten => s-BUSY,1,Voicemail(${MACRO_EXTEN}|b)
exten => s-BUSY,n,Hangup
exten => s-NOANSWER,1,Voicemail(${MACRO_EXTEN}|u)
exten => s-NOANSWER,n,Hangup
exten => _s-,1,Answer()
exten => _s-,n,Playtones(busy)
exten => _s-,n,Busy(5)
```

Otras funcionalidades de PBX

ironotec
GNU/Linux

Asterisk como PBX

- Toda la secuencia y programación del dialplan es el verdadero núcleo del sistema centralita, si bien, las siguientes funcionalidades se configuran en features.conf:
 - Transferencias de llamadas: transferencia de llamadas entre diversos usuarios, independientemente de la tecnología que usen.
 - Call Parking: Parking de llamadas.
 - Call Pickup: Auto-transferencia (o robo de llamada) de un teléfono que esté sonando.

Transferencias

- En caso de SIP e IAX2: La transferencia DEBE ser nativa. El usuario SIP es el dueño de su llamada!
- En features.conf se especifica:
 - blindxfer => secuencia
 - Permite realizar una transferencia de llamada a ciegas marcando la secuencia.
 - atxfer => secuencia
 - Permite realizar una transferencia de llamada atendida. El origen es puesto en espera, mientras el destino se comunica con el nuevo destino para anunciarle la llamada. Si el nuevo destino cuelga, la llamada no se transfiere.
 - pickupexten => secuencia
 - Especifica como coger una llamada del grupo (callgroup).

Música en Espera

Asterisk puede poner un canal dado en espera ('HOLD'), principalmente en las siguientes situaciones:

- Durante una transferencia.
- Durante una llamada si se ha especificado el parámetro 'm', que indica que no se oirá tono de llamada sino música en espera.
- Durante una espera en el parking.
- Si la aplicación MusicOnHold ha sido llamada desde el DialPlan.
- Si el destino de la llamada ha solicitado explícitamente que la llamada sea puesta en espera.
- Es posible tener distintos tipos de música en espera.
- La música en espera se configura en el fichero **musiconhold.conf**

CallParking (I)

- El callparking es una funcionalidad que permite transferir la llamada a un 'parking' virtual. Pudiendo colgar sin que la llamada origen sea desconectada de Asterisk, ya que se encontrará aparcada.
 - Usuario: “Tienes una llamada por la 3”
- Para operar, el usuario transfiere la llamada a una extensión especial (parkext en features.conf), Asterisk aparca la llamada y anuncia la posición en el parking.
- Cualquier usuario que tenga incluido en su contexto el contexto especial 'parkedcalls' puede recuperar la llamada aparcada llamando directamente a su posición en el parking.

CallParking (II)

1) A y B están en conversación.

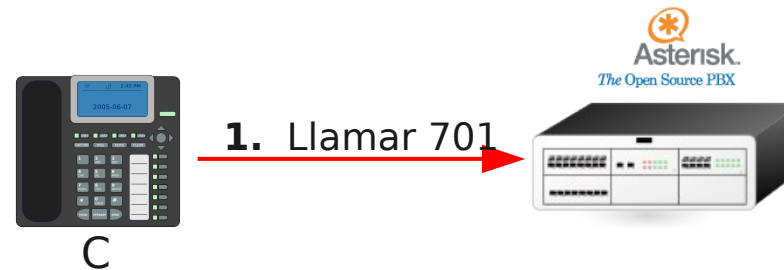


2) A transfiera al 700 y 'aparcas' a B en el Parking 701.



CallParking (III)

- 3) Desde otro teléfono C se puede recoger a B del Parking marcando el 701.



- 4) C y B están en conversación.



CallPickup

- 'CallPickup' es el hecho de poder descolgar y responder a la llamada entrante a un teléfono o grupo de teléfonos determinado desde un tercer teléfono que no está sonando.
- La configuración de los códigos DTMF para 'CallPickup' se configura en features.conf
- Se definen dos conceptos:
 - callgroup: Toda llamada que entra a una extensión determinada pertenece al/los callgroups de la extensión.
 - pickupgroup: Es el/los callgroups de llamadas entrantes que el usuario puede hacer 'pickup'.
- En cada usuario de la centralita se pueden definir esos dos parámetros.

CallPickup (II)

- La aplicación PickUp nos permite hacer una captura 'dirigida' de una llamada
 - Podemos seleccionar *cual* de las llamadas queremos capturar.

exten => *8XXX,1,PickUp(\${EXTEN:2}@usuarios)



DISA

- La aplicación DISA nos permite ofrecer un tono de marcado al usuario para que éste (opcionalmente) se autentique y llame desde el contexto indicado

exten => 1234,1,DISA(no-password, desde-disa)



Construyendo un dialplan sostenible

ironotec
GNU/Linux

Escenario "sencillo"

- Muy típico, una oficina con:
 - Telefonía SIP interna.
 - Entrantes vía PSTN:
 - 2 numeraciones, 2 departamentos, horario, buzón de voz.
 - Salientes vía PSTN y proveedor VoIP:
 - Según destino.
 - Servicios internos:
 - Consulta de buzón, conferencia.

Dialplan poco mantenible (I)

– sip.conf

```
[plantilla-usuarios](!  
    type = friend  
    host = dynamic  
    secret = ****  
    canreinvite = yes  
    context = usuarios  
  
[200](plantilla-usuarios)  
[201](plantilla-usuarios)  
[202](plantilla-usuarios)  
...
```

Dialplan poco mantenible (II)

– extensions.conf

[usuarios]

```
; A usuarios internos:
exten => _2XX,1,Dial(SIP/${EXTEN}|80|tT)
; Nacionales:
exten => _[6789]XXXXXXXX,1,Dial(mISDN/g:bri/${EXTEN}|80|T)
; Extranjero:
exten => _00.,1,Dial(SIP/proveedor-voip/${EXTEN}|80|T)
; Urgencias:
exten => 112,1,Dial(mISDN/g:bri/112||T)
; Números cortos:
exten => _1XXX,1,Dial(mISDN/g:bri/${EXTEN}|80|T)
; Consulta buzón de voz:
exten => 500,1,VoiceMailMain(buzon-entrantes)
; Sala de conferencia:
exten => 501,1,MeetMe(501)
```

Dialplan poco mantenible (III)

```
[entrantes-rdsi]
```

```
; Departamento de ventas:
```

```
exten => 999000111,1,Macro(comprueba-horario-ventas)
```

```
exten => 999000111,n,Goto(999000111-${HORARIO},1)
```

```
exten => 999000111-DENTRO,1,Queue(ventas|wt||45)
```

```
exten => 999000111-FUERA,1,PlayBack(fuera-de-horario)
```

```
exten => 999000111-FUERA,n,VoiceMail(buzon-entrantes,s)
```

```
; Departamento técnico:
```

```
exten => 999000222,1,Macro(comprueba-horario-tecnicos)
```

```
...ídem...
```

```
[macro-comprueba-horario-ventas]
```

```
...
```

```
[macro-comprueba-horario-tecnicos]
```

```
...
```

Dialplan poco mantenible (V)

- Pero las cosas cambian con el tiempo:
 - Numeraciones
 - Líneas
 - Personas
 - Necesidades
 - etc...
 - **iii Y esto nos lo piden estando Asterisk en producción !!!**



Dialplan poco mantenible (VI)

- ¿Qué desemboca un mínimo cambio?
 - Ej: Contratan 2 becarios y no quieren que llamen al extranjero (¿?¿? ... pero el cliente manda)
 - Solución "a-toda-prisa":

- sip.conf

```
[plantilla-becarios](!)  
context = becarios  
secret = *****  
...
```

```
[210](plantilla-becarios)  
[211](plantilla-becarios)
```


Dialplan poco mantenible (VII)

- extensions.conf

[becarios]

; A usuarios internos:

exten => _2XX,1,Dial(SIP/\${EXTEN}|80|tT)

; Nacionales:

exten => _[6789]XXXXXXXX,1,Dial(mISDN/g:bri/\${EXTEN}|80|T)

~~; Extranjero:~~

~~exten => _00.,1,Dial(SIP/proveedor-voip/\${EXTEN}|80|T)~~

; Urgencias:

exten => 112,1,Dial(mISDN/g:bri/112||T)

; Números cortos:

exten => _1XXX,1,Dial(mISDN/g:bri/\${EXTEN}|80|T)

; Consulta buzón de voz:

exten => 500,1,VoiceMailMain(buzon-entrantes)

; Sala de conferencia:

exten => 501,1,MeetMe(501)

Dialplan poco mantenible (VIII)

- Ahora nos piden alargar la duración de la llamada.
 - Ala, a cambiarlo en 30 sitios.
- Nos piden otra sala de conferencia 502.
 - Lo mismo.
- ¡ Se nos olvidó los números de emergencias 0XX !
 - Añadir en ambos contextos:
`exten => _0XX,1,Dial(mISDN/g:bri/${EXTEN}||T)`
- Y puede ser peor...



Dialplan poco mantenible (IX)

- Nos piden conectar una FCT por FXO para llamadas vía GSM (ahorro de coste).
 - A duplicar código otra vez.
- Cambian los números RDSI.
 - Toca modificar muchas líneas.
- Permitir entrantes vía GSM al dept. ventas.
 - Duplicar código o la "super-ñapa":

```
[entrantes-trac]
```

```
exten => s,1,Goto(entrantes-rdsi,999000111,1)
```

!!! ahhhhhh !!!

Dialplan mantenible (I)

– sip.conf

```
; Empleados:
```

```
[plantilla-empleados](!)
```

```
...
```

```
context = desde-empleados
```

```
[200](plantilla-empleados)
```

```
[201](plantilla-empleados)
```

```
[202](plantilla-empleados)
```

```
; Becarios:
```

```
[plantilla-becarios](!)
```

```
context = desde-becarios
```

```
[210](plantilla-becarios)
```

```
[211](plantilla-becarios)
```

Dialplan mantenible (II)

– extensions.conf

```
; Por comodidad lo dividimos en varios ficheros:  
#include dialplan/*.dialplan
```

```
[globals]
```

```
; Canales:
```

```
RDSI = mISDN/g:bri
```

```
VOIP = SIP/proveedor-voip
```

```
; Constantes:
```

```
TIMEOUT = 80
```



Dialplan mantenible (III)

– /etc/asterisk/dialplan/usuarios.dialplan

[desde-empleados]

```
include => a-usuarios  
include => salientes-nacionales  
include => salientes-internacionales  
include => salientes-otras  
include => servicios
```

[desde-becarios]

```
include => a-usuarios  
include => salientes-nacionales  
include => salientes-otras  
include => servicios
```

[a-usuarios]

```
exten => _2XX,1,Dial(SIP/${EXTEN}|${TIMEOUT}|tT)
```

Dialplan mantenible (IV)

– /etc/asterisk/dialplan/salientes.dialplan

```
[salientes-nacionales]
```

```
    exten => _[6789]XXXXXXXX,1,Dial(${RDSI}/${EXTEN}|${TIMEOUT}|T)
```

```
[salientes-internacionales]
```

```
    exten => _00.,1,Dial(${VOIP}/${EXTEN}|${TIMEOUT}|T)
```

```
[salientes-otras]
```

```
    ; Urgencias:
```

```
    exten => 112,1,Dial(${RDSI}/112||T)
```

```
    exten => _0XX,1,Dial(${RDSI}/${EXTEN}||T)
```

```
    ; Números cortos:
```

```
    exten => _1XXX,1,Dial(${RDSI}/${EXTEN}|${TIMEOUT}|T)
```

Dialplan mantenible (V)

– /etc/asterisk/dialplan/entrantes.dialplan

```
[entrantes-rdsi]
```

```
    exten => 999000111,1,Goto(entrantes-ventas,s,1)
```

```
    exten => 999000222,1,Goto(entrantes-tecnicos,s,1)
```

```
[entrantes-gsm]
```

```
    exten => s,1,Goto(entrantes-ventas,s,1)
```

```
[entrantes-ventas]
```

```
    exten => s,1,Macro(comprueba-horario-ventas)
```

```
    exten => s,n,Goto(${HORARIO},1)
```

```
    exten => DENTRO,1,Queue(ventas|wt||45)
```

```
    exten => FUERA,1,PlayBack(fuera-de-horario)
```

```
    exten => FUERA,n,VoiceMail(buzon-entrantes,s)
```

```
[entrantes-tecnicos]
```

```
    ...ídem...
```


Dialplan mantenible (VI)

– /etc/asterisk/dialplan/servicios.dialplan
[servicios]

; Consulta buzón de voz:

exten => 500,1,VoiceMailMain(buzon-entrantes)

; Salas de conferencia:

exten => 501,1,MeetMe(501)

exten => 502,1,MeetMe(502)

Precauciones

- Numeraciones de España (que no se nos olvide ningún número):

http://en.wikipedia.org/wiki/Telephone_numbers_in_Spain

http://es.wikipedia.org/wiki/Números_de_teléfono_de_emergencias

- Cuidado con los parámetros "Dial":
 - No poner "t" en una llamada al exterior (estamos dando permiso al llamado para que nos transfiera a cualquier número -> **i pagamos nosotros !**).



Conectando servidores Asterisk

ironitec
GNU/Linux

Enlaces SIP en Asterisk

- A parte de para registrar teléfonos podemos utilizar SIP para enlazar Asterisk con cualquier PBX, gateway, proxy, ... que hable SIP.
 - Configuración de un peer/user/friend.
 - Register
 - register => usuario:password@host



Ejercicio

- Realiza un enlace por SIP con la PBX de tu compañero.
- Utiliza cuentas de “friend”.
- Haz un contexto para que las extensiones de una PBX puedan llamar a las de la otra.
- Inventa un prefijo para que al marcarlo la llamada sea enviada a la otra centralita.



En la centralita A:

```
[pbxB]  
type=friend  
username=pbxA  
fromuser=pbxA  
secret=1234  
context=entrantes-sip  
host=123.123.123.123  
exten => _666X.,1,Dial(SIP/${EXTEN:3}@pbxB,45)
```

En la centralita B:

```
[pbxA]  
type=friend  
username=pbxB  
fromuser=pbxB  
secret=1234  
context=entrantes-sip  
host=123.123.123.123  
exten => _666X.,1,Dial(SIP/${EXTEN:3}@pbxA,45)
```

Ejercicio

- Realiza un enlace por IAX2 con la PBX de tu compañero.
- Utiliza cuentas de “friend”.
- Haz un contexto para que las extensiones de una PBX puedan llamar a las de la otra.
- Inventa un prefijo para que al marcarlo la llamada sea enviada a la otra centralita.



En la centralita A:

```
[pbxB]  
type=friend  
username=pbxA  
secret=1234  
context=entrantes-iax  
host=123.123.123.123  
exten => _666X.,1,Dial(IAX2/pbxA@pbxB/${EXTEN:3},45)
```

En la centralita B:

```
[pbxA]  
type=friend  
username=pbxB  
secret=1234  
context=entrantes-iax  
host=123.123.123.123  
exten => _666X.,1,Dial(IAX2/pbxB@pbxA/${EXTEN:3},45)
```

Sistema de colas y agentes

ironotec

- Las colas nos permiten manejar de una manera cómoda y eficiente las llamadas entrantes.
- Las llamadas se distribuyen entre los agentes disponibles (que hayan iniciado sesión)
- Consisten básicamente en **miembros** que contestan las llamadas.



Configuración de colas

queues.conf

```
[general]
language=es
persistentmembers = yes           //si reiniciamos...
autofill = yes                    //varias conexiones a la vez

[ventas]
musiconhold = default
strategy = ringall                // ringall, roundrobin
                                   // leastrecent
                                   // fewestcalls, random
                                   // rrmemory

timeout = 15
retry = 5                         // tiempo para reintentar
wrapuptime = 0                   // tiempo para volver a
                                   // llamarle

maxlen = 0
announce-holdtime = no
periodic-announce = queue-periodic-announce
periodic-announce-frequency=20
member => SIP/201
member => Agent/@1
```

Aplicación Queue

- Muy similar a la aplicación Dial.
- Toma parámetros parecidos, pero en lugar del dispositivo se especifica el nombre de la cola.
- Ejemplo:
 - exten => 1234,n,Queue(ventas|t||45|)



- Los agentes son “extensiones virtuales”.
- En realidad son extensiones normales que automáticamente o tras un proceso de autenticación se convierten en agentes y pasan a atender las llamadas de una o varias colas.



Configuración de agentes

agents.conf

```
[agents]  
autologoff= 15  
wrapuptime = 0  
musiconhold = default
```

//tiempo para volver a llamarle

```
group = 1  
agent => 3001,1234,Comercial 1  
agent => 3002,1234,Comercial 2
```



Login de Agentes

- Cualquier extensión puede ser agente.
- Basta con que inicie sesión con un número de agente y contraseña válidos y automáticamente comenzará a recibir llamadas.
- En lugar de utilizarse agentes, utilizaremos **“miembros dinámicos”**.
- Ejemplo:
 - exten => 1234,1,AddQueueMember(ventas)
 - exten
1235,1,RemoveQueueMember(ventas) =>

Informes de llamadas: CDRs

ironotec
GNU/Linux

- En centralitas, proveedores de servicios, etc. es importante tener un control de las llamadas.
- Asterisk puede generar CDRs (Call Detail Record) en distintos formatos
 - CSV
 - MySQL
 - SQLite
 - ...
- Nos dan **todos** los detalles de las llamadas.
 - Billing.
- Para utilizar el CDR en MySQL es necesario compilar asterisk-addons. (instalar paquete libmysqlclient15-dev)

CDRs (2)

- Por defecto el CDR se genera en CSV en /var/log/asterisk/
- Posibilidad para desarrollar aplicaciones de estadística para CallCenters, ...



Ejercicio

- Crear la estructura de tablas necesaria y poner en funcionamiento el sistema de CDRs en base de datos MySQL.
- Receta:
 - Si no lo esta, compilar asterisk-addons.
 - Configurar el fichero cdr_mysql.conf para que apunte a nuestra base de datos.
 - Crear la estructura de la base de datos.
 - Cargar el módulo editando el fichero modules.conf
load => cdr_addon_mysql.so



Aspectos avanzados de Asterisk

ironotec
GNU/Linux

- AEL (Asterisk Extension Language) es una forma diferente de escribir el dialplan.
- Más orientado a programadores:
 - switch
 - while/for
 - ...
- Utiliza el módulo pbx_ael.so (pbx_config.so carga el dialplan normal)
- Internamente Asterisk trabaja con el dialplan clasico, “parsea” y convierte el AEL.
- Muy útil, y mucho más legible.
- Más información:
<http://www.voip-info.org/wiki/view/Asterisk+AEL2>

- **Ejemplo:**

```
context prueba {  
    1234 => {  
        Dial(SIP/saghul,45);  
        switch (${DIALSTATUS}) {  
            case BUSY:  
                Voicemail(b200);  
                break;  
            case NOANSWER:  
                Voicemail(u200);  
                break;  
            default:  
                Noop(Algo raro ha pasado);  
                Hangup;  
        }  
    }  
}
```

- AGI (Asterisk Gateway Interface) nos permite ejecutar en Asterisk software de terceros escrito en casi cualquier lenguaje
- Permite **extender al infinito** las posibilidades de Asterisk, juntando su potencia, con las posibilidades que ofrece un lenguaje de programación.
- Muchos lenguajes soportados: Python, PHP, Perl, Bash, Java, ...
- Conviene utilizar un lenguaje que no resulte demasiado lento, para no demorar demasiado la ejecución.
- Opinión personal:
 - PHP es un buen lenguaje para AGI.
 - Se ejecuta rápido.
 - No es difícil programar en PHP.
 - La librería phpagi nos hace la vida más fácil.
<http://phpagi.sourceforge.net/>

- Ejemplo de AGI con phpagi:

```
#!/usr/bin/php
<?php
require_once("phpagi.php");
$myagi = new AGI();
$myagi->set_variable("VAR","hooola");
$myagi ->exec_dial("SIP","saghul",45);
?>
```

- En el dialplan pondríamos:

```
exten => 1234,1,AGI(prueba.php)
exten => 1234,n,Noop(${VAR})      //nos saldría hooola
```


- **¿Qué es Asterisk AMI?**

Asterisk AMI permite que programas cliente se conecten a Asterisk mediante TCP/IP y sean capaces de ejecutar comandos y leer eventos. Por cada cosa que Asterisk realiza se generan eventos que pueden ser leídos mediante una sesión de manager, y el usuario puede tratarlos a su gusto. Además, AMI permite la ejecución de comandos, lo que proporciona la posibilidad de alterar el comportamiento de Asterisk desde un programa hecho a medida.

- **Funcionamiento:**

Para trabajar con AMI es necesario tener un usuario definido en el fichero manager.conf. A partir de aquí hay que establecer una comunicación TCP/IP con el servidor de Asterisk en el puerto 5038, y una vez conectado y autenticado, se puede comenzar a leer los eventos o ejecutar comandos.

- **¿AJAM?**

De la mano de Asterisk 1.4 viene AJAM (Asynchronous Javascript Asterisk Manager), un nuevo manager, que permite conectar con Asterisk por medio de HTTP. Para poder trabajar con AJAM es necesario configurar los ficheros manager.conf y httpd.conf. Adivina como funciona el nuevo GUI de Asterisk..... premio!

- **Ejemplos de funcionamiento:**

`http://IP_de_Asterisk:8088/asterisk/manager?
action=login&username=nombre_de_usuario&secret=contraseña`

Esto abrirá una sesión de Asterisk Manager. Si ahora ejecutas:

`http://IP_de_Asterisk:8088/asterisk/rawman?action=status`

Verás la salida del comando.

- **Más información:**

<http://www.voip-info.org/wiki-Asterisk+manager+API>

<http://www.voip-info.org/wiki/view/Aynchronous+Javascript+Asterisk+Manager+%28AJAM%29>

- Asterisk permite ser configurado (algunos ficheros) en una Base de Datos, mediante **Asterisk Realtime Architecture**.
- Soporta MySQL, y es posible migrar a Base de Datos lo siguiente:
 - Configuración de dispositivos SIP e IAX.
 - Configuración de buzones de voz.
 - Configuración de colas.
- Las estructuras de la bases de datos están en <http://www.voip.info.org>
- El fichero a configurar es **extconfig.conf**
- El fichero `res_mysql.conf` contiene la configuración de la base de datos.

- ARA dispone de 2 tipos de Realtime: **estático y dinámico**:
 - **Estático**: La configuración esta almacenada en la BD, pero Asterisk la carga al arrancar como si fuera un fichero. Si se realiza algún cambio, es necesario hacer un reload.
 - **Dinámico**: La configuración esta almacenada en la BD y Asterisk realiza una consulta a esa BD cada vez que necesita un dato. No es necesario hacer reload si se han realizado cambios. Mucha carga para el servidor...
- Todos los ficheros **no soportan** Realtime Dinámico.
 - SIP, IAX y los buzones de voz en Realtime Dinámico.
 - Colas en Realtime Dinámico.
 - MeetMe en Realtime Dinámico.

- Ejemplo de configuración (extconfig.conf):

```
[settings]
;;RealTime Dynamic
;sipusers => mysql,asterisk,dispositivos_sip
;sippeers => mysql,asterisk,dispositivos_sip
;iaxusers => mysql,asterisk,dispositivos_iax
;iaxpeers => mysql,asterisk,dispositivos_iax
;voicemail => mysql,asterisk,buzones

;;RealTime Static
;sip.conf => mysql,asterisk,sip_conf
;extensions.conf => mysql,asterisk,extensions_conf
;iax.conf => mysql,asterisk,iax_conf
;queues.conf => mysql,asterisk,queues_conf
;voicemail.conf => mysql,asterisk,voicemail_conf
```

Miscelánea

ironotec
GNU/Linux

- Algunos terminales IP nos ofrecen la posibilidad de monitorizar el estado de otras extensiones mediante teclas con LEDs: funcionalidad BLF.
- Para configurar Asterisk y que el BLF funcione correctamente es necesario hacer lo siguiente:
 - Configurar el fichero sip.conf
 - Añadir 'hints' al dialplan (extensions.conf)

- sip.conf

```
allowssubscribe=yes  
subscribecontext = hints  
notifyingringing = yes  
notifyhold = yes  
limitonpeers = yes  
notifymimetype=application  
/simple-message-summary  
call-limit=2
```

- extensions.conf

```
[hints]  
exten => 2000, hint, SIP/saghul  
exten => 2000, 1, Dial(SIP/saghul)
```

- Verificamos que los hints funcionan correctamente:

- *CLI> core show hints

- Verificamos las subscripciones:

- *CLI> sip show subscriptions

Funciones de grupo

- Asterisk dispone de funciones para contar o agrupar canales
 - Útiles cuando queremos controlar cuantas llamadas salen por un proveedor
 - Cuantas llamadas pueden enviarse a un terminal

exten => s,1,Set(GROUP())=supergrupo)

exten => s,2,GotoIf(\${GROUP_COUNT(supergrupo)} > 5]?error)



Asterisk: Auto-dial out

Auto-dial out

- Permite iniciar llamadas desde aplicaciones externas.
- Se copia un fichero tipo call(callfiles) en: /var/spool/asterisk/outgoing/
 - Asterisk inmediatamente llamará al canal especificado en Channel y lo conectará con el contexto dado (también es posible especificar una aplicación).
- Ejemplo:

Channel: SIP/bt100

MaxRetries: 1

RetryTime: 60

WaitTime: 30

Context: outgoing

Extension: 944048182

Priority: 1

Telefonía tradicional

ironotec

CNU/Lin

Asterisk PBX: Telefonía Tradicional

Asterisk y la Telefonía Tradicional

- Para poder interoperar con la telefonía tradicional, Asterisk necesita hardware específico.
- El principal sponsor y desarrollador de Asterisk: Digium es el principal fabricante de hardware.



Asterisk PBX: Telefonía Tradicional

Asterisk y la Telefonía Tradicional

- Para operar con líneas analógicas, se necesitan tarjetas con interfaces FXO
 - Ejemplo: Digium TDM01B
- Para operar con teléfonos analógicos o centralitas clásicas, se requieren interfaces FXS
 - Ejemplo: Digium TDM10B
- En ambos casos, el driver a utilizar es DAHDI, la configuración se almacena en `/etc/dahdi/`

Asterisk PBX: Telefonía Tradicional

Asterisk y la Telefonía Tradicional (II)

- En líneas digitales (RDSI), en Europa existen dos tipos:
 - BRI : Acceso básico, proporciona 2 canales de VOZ.
 - PRI: Acceso primario, proporciona 30 canales de Voz (E1).
- Asterisk soporta perfectamente ambos tipos de líneas digitales, con hardware específico:
 - Para primarios, Digium proporciona tarjetas de hasta 4 puertos.
 - Para básicos, Digium proporciona tarjetas de hasta 4 puertos.

Asterisk PBX: Telefonía Tradicional

Lineas analógicas

- Para operar con las tarjetas con interfaces FXS / FXO, Asterisk utiliza el subsistema DAHDI, antes conocido como Zaptel.
- Es necesario tener compilado DAHDI, así como Asterisk con su soporte habilitado
- Seleccionar los módulos acordes a nuestro hardware en /etc/dahdi/modules
- Una vez instalado el sistema DAHDI, es necesario configurarlo en /etc/dahdi/system.conf
 - Definir zonas (para frecuencias de tonos)
 - Definir interfaces en los canales: FXS / FXO
- Se puede verificar la configuración correcta con el comando instalado: **dahdi_cfg -v**

Asterisk PBX: Telefonía Tradicional

Lineas analógicas II

- Asterisk utiliza los módulos provistos por DAHDI para acceder al hardware.
- Este enlace se configura en:
`/etc/asterisk/chan_dahdi.conf`
- Aspectos importantes a configurar:
 - context: contexto donde irán las llamadas generadas por cada canal.
 - echocancel: cancelación de echo (problema importante en telefonía).
 - Para utilizar correctamente las lineas españolas son necesarios los siguientes parámetros:

```
answeronpolarityswitch=yes  
hanguponpolarityswitch=yes
```

Tarjetas Analógicas



TDM11B

Configuración TDM

/etc/dahdi/system.conf

```
loadzone=es
defaultzone=es
fxoks=1
fxsks=2
echocanceller=mg2,1-2
```

/etc/asterisk/chan_dahdi.conf

```
[trunkgroups]
[channels]
language=es
hidecallerid=no
callwaiting=yes
echocancel=yes
echocancelwhenbridged=no
echotraining=yes
transfer=yes
usecallerid=yes
callerid=asreceived
rxgain=0.0
txgain=0.0
busydetect=no
busycount=5
answeronpolarityswitch=yes
hanguponpolarityswitch=yes
immediate=no
signalling=fxs_ks
context=entrada-pstn
group=1
channel=2
group=2
signalling=fxo_ks
context=desde-usuarios
channel=1
```

Líneas Digitales

- También utilizan el subsistema DAHDI
 - Las tarjetas de BRI != b410p utilizan mISDN (de momento)
- Los ficheros a configurar son los mismos.
- Las BRI utilizan mISDN en lugar de DAHDI (Junghans no - BriStuff)



Tarjetas PRI



TE120P

Configuración PRI

/etc/dahdi/system.conf

```
loadzone=es
defaultzone=es
span=1,1,0,ccs,hdb3,crc4
bchan=1-15
dchan=16
bchan=17-31
echocanceller=mg2,1-15,17-31
```

/etc/asterisk/chan_dahdi.conf

```
[trunkgroups]
[channels]
callwaiting=yes
transfer=yes
echocancel=yes
echocancelwhenbridged=no
language=es
switchtype=euroisdn
signalling=pri_cpe
usercallerid=yes
callerid=asreceived
rxgain=0.0
txgain=0.0
group=1
context=pri-in
channel=1-15,17-31
```

Tarjetas BRI



B410P

Configuración PRI

/etc/dahdi/system.conf

```
loadzone=es  
defaultzone=es  
span=1,1,0,ccs,ami,crc4  
bchan=1-2  
hardhdlc=3  
encocanceller=mg2,1-2
```

/etc/asterisk/chan_dahdi.conf

```
[trunkgroups]  
[channels]  
callwaiting=yes  
transfer=yes  
echocancel=yes  
echocancelwhenbridged=no  
language=es  
switchtype=euroisdn  
signalling=bri_cpe_ptmp  
usercallerid=yes  
callerid=asreceived  
rxgain=0.0  
txgain=0.0  
  
group=1  
context=bri-in  
channel=1-2
```

Anexo A: Softphones

ironotec
GNU/Linux

¿ Que son ?

- Se trata de software que se ejecuta en estaciones o servidores de trabajo.
- Permiten establecer llamadas de Voz sobre IP.
- El audio es capturado desde:
 - Micrófono Incorporado
 - Entrada de linea (Micrófono Externo).
 - Dispositivos de entrada de audio USB
 - Dispositivos Bluetooth

Tipos de Softphones

- Propietarios
 - Protocolos estándar: SIP, H323 ...
 - Protocolos propios abiertos.
 - Protocolos propios cerrados.
- Libres
 - Protocolos estándar.
 - Protocolos propios abiertos.

Características Principales

- Integración con el entorno (Escritorio)
 - Icono en systray, dock ...
 - Aviso visual de llamadas entrantes.
- Integración con plataformas de acceso y validación de usuarios (LDAP).
- Importación / Exportación de datos: libretas de contactos en XML.
- Soporte de varias conversaciones simultáneamente y en algunos casos de varias líneas.

Softphones Privativos

Skype

- El más conocido de los softphones y quizás un responsable importante de la popularización de la VoIP
- Creado por los fundadores de Kazaa: Zennström y Friis.
- Descargado (según skype.com): 236.259.232 veces
- Skype fue comprada por la firma de subastas por Internet E-Bay por 2.100 millones de dolares.
- Las comunicaciones de Voz viajan cifradas por la red, utiliza un protocolo propietario.



Softphones Privativos

Skype: Ventajas

- Disponible para muchas plataformas: MS Windows, Mac OSX, GNU Linux, Pocket PC
- Codificación de audio con mucha calidad y gran compresión: 3-16 kilobytes / segundo
- Conferencias de llamadas. Envío de Video (V2.x)
- Firewall / Nat discover: En casi todas las situaciones funciona sin necesidad de configurar PNAT



Softphones Privativos

Skype: Problemas, protocolo cerrado

- ¿ Qué están haciendo con mis paquetes de voz ?
 - Creadores de la Red Kazaa bajo sospecha de distribuir spyware de forma intencionada.
- ¿ Qué están haciendo con mi ancho de banda ?
 - Utilizarlo para otros clientes de Skype.
- Interconexión con otra redes: el salto a la red telefónica pública solo puede realizarse con el sistema SkypeOut, lo cual no favorece la competencia.

Softphones Privativos

Counterpath Eyebeam / Bria

- Software privativo, con licencia para distribución con marca propia o compartida.
- Disponible para MS Windows, GNU Linux, Mac OSX y Pocket PC.
- Soporta el estándar SIP y prácticamente todos los codecs disponibles..



Softphones Privativos

Counterpath Eyebeam / Bria: Ventajas

- Disponible para descarga la versión gratuita X-Lite
- Soporte de múltiples conversaciones simultáneas.
- Soporte de múltiples proxys configurados.
- Soporte para utilización de STUN Server.
- Utilización de registros SRV.
- Configuraciones Avanzadas: DTMF, RTP, ...



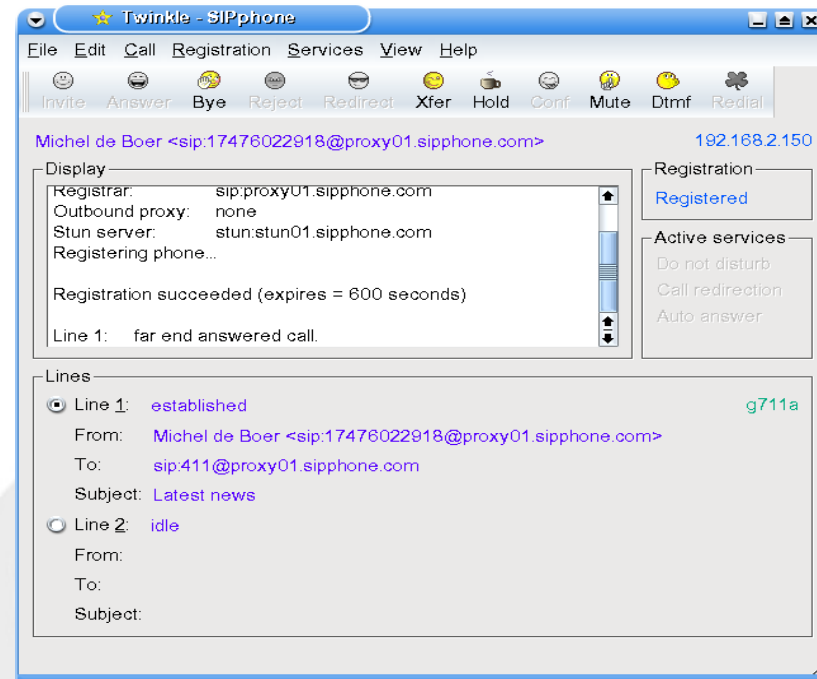
Counterpath Eyebeam / Bria: Desventajas

- No soporta IAX2, H323.
 - ¿desventaja?
- Es software privativo.

Twinkle

- Softphone para GNU Linux
- Entorno KDE (no requisito)

Twinkle 



Twinkle: Ventajas

- Licencia GPL.
- Interfaz de configuración muy amigable.
- Soporta múltiples perfiles.
- Soporta llamadas utilizando SRV DNS.
- Permite la utilización de STUN.
- El más completo
- Buena implementación de estándares



Twinkle: Desventajas

- No soporta el codec G.729 (debido a la necesidad de una licencia), tampoco es posible con fines educativos.
- No soporta IAX2, ni H323.
 - ¿desventaja?
- No hay versiones para MS Windows o Mac OSX.



Anexo B: Terminales físicos

ironotec

CNU/Linu

Teléfonos IP

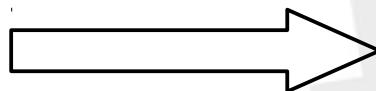
¿ Que son ?

- Son aparatos telefónicos con la misma apariencia física que los teléfonos tradicionales.
- Utilizan tecnologías VoIP y normalmente permiten realizar ciertas funcionalidades avanzadas (llamada en espera, ...).



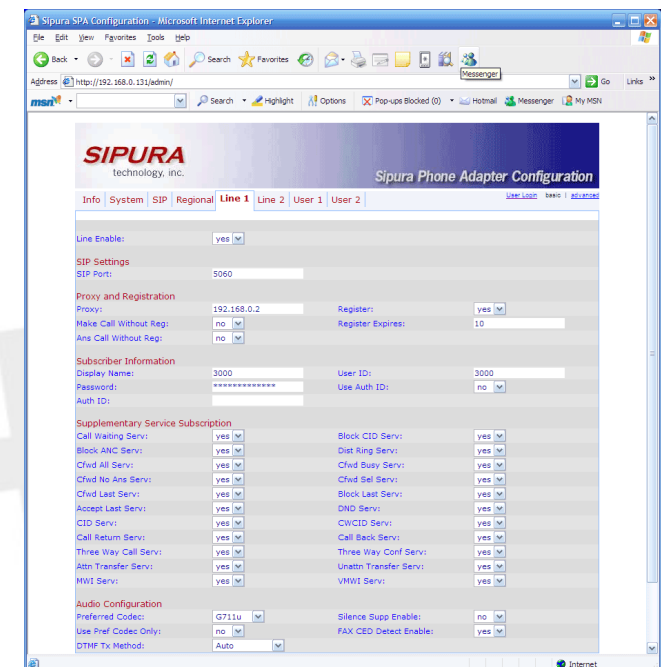
Características Principales

- Normalmente soportan un único protocolo de VozIP (SIP, IAX2, H323). Algunos permiten cambio de protocolo con firmware.
- Soportan una serie de codecs, el famoso G.729 casi siempre está entre ellos.
- Se conectan directamente a la Red IP:



Características Principales (II)

- Se configuran desde los menús del propio teléfono o por interfaz web:



Características Avanzadas

- Dual LAN: La mayoría de los teléfonos disponen de dos conectores RJ45 e implementan funciones de switch, de esta forma no es necesario tirar cableado nuevo para los nuevos dispositivos IP



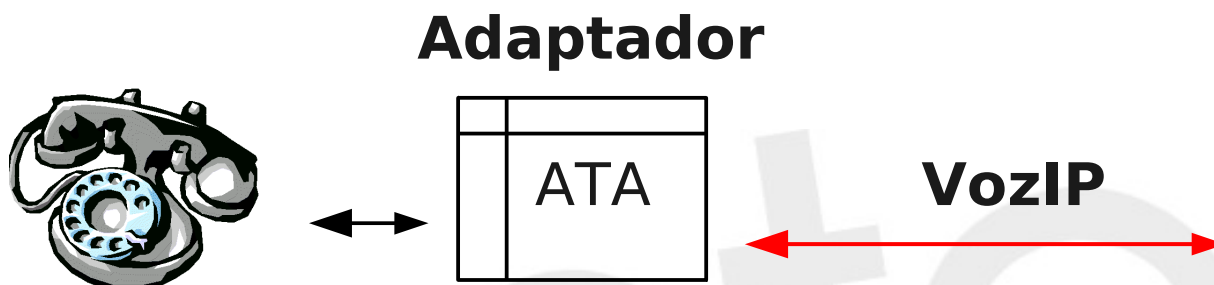
Características Avanzadas (II)

- Manos Libres
- Conector auriculares/micrófono:
- Display LCD: Caller ID / Agenda
- “Autoprovisioning”: Autoconfiguración automática de los parámetros de configuración desde un servidor remoto.
- PoE



Adaptadores IP: ¿ Que son ?

- Son dispositivos (hardware) que permiten conectar un teléfono analógico a la red IP utilizando protocolos de VoIP.



Tipos de Adaptadores

- **ATA:** Analog Telephone Adapter, el caso más normal, tienen un conector FXS para teléfono analógico normal y envían por VoIP a través del conector LAN, soportan SIP normalmente.
- **USB Phone:** Permiten conectar un teléfono normal a un PC, enviando y recibiendo el audio. Requieren un softphone instalado para VoIP.

Adaptadores IP: Características ATA

- Soportan SIP o IAX2 normalmente, varios codecs (entre ellos el G.729).
- Tienen uno o dos interfaces FXS para conectar 2 teléfonos analógicos.
- Cada teléfono puede ser registrado a un servidor VoIP distinto.
- Soportan caller ID, tonos de llamada distintivos, llamada en espera, ...

Adaptadores IP: Ejemplos

- Linksys PAP2: 2 puertos FXS. SIP.
- Digium IAXy: 1 puerto FXS. IAX2.
- Cisco ATA 18x: 2 puertos FXS, 2 LAN. SIP, H.323, MGCP, SCCP
- Atcom AG-168V: 1 puerto FXS, 1 puerto FXO desvío de llamadas por la red telefónica tradicional. SIP, H323, IAX2, Net2Phone.



Adaptadores IP: Ejemplo Linksys PAP2

- 2 FXS, 1 LAN. Configuración vía Web.
- Prestaciones avanzadas: dialplan, llamada en espera, parámetros SIP avanzados, autoprovisioning.
- Coste muy económico.
- Integración sencilla con los proveedores de VoIP.



Gateways: ¿ Que són ?

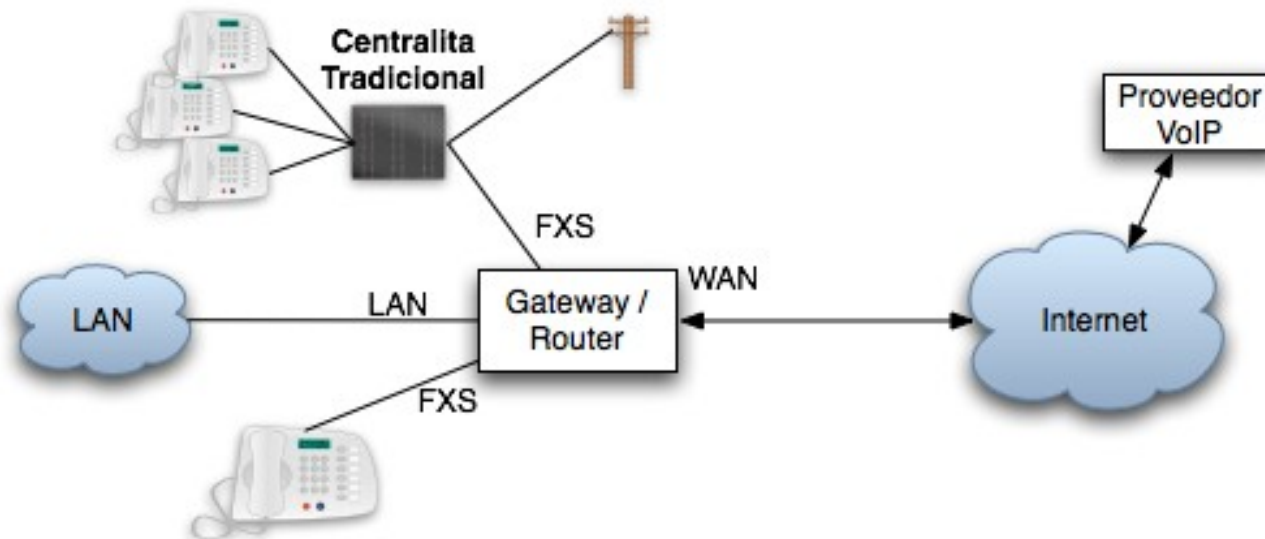
- Por definición aceptada, permiten interconectar la telefonía tradicional con la telefonía por IP (Voz IP).
- Se integran con la red telefónica pública con interfaces analógicos o enlaces digitales.
- Los adaptadores también pueden ser considerados como *gateways*, a pequeña escala.

Gateways: Características Principales

- Generalmente funcionan en dos sentidos: las llamadas recibidas por IP se envían a PSTN/FXS o las llamadas recibidas por interfaces FXS se envían por IP.
- Soportan generalmente SIP o H.323, así como numerosos codecs (G.711, G.729, ...).
- Pueden ser utilizados de forma integrada con las centralitas tradicionales: transformando la llamada analógica de la centralita en llamada por IP, de forma totalmente transparente.

Gateways: FXS -> VoIP

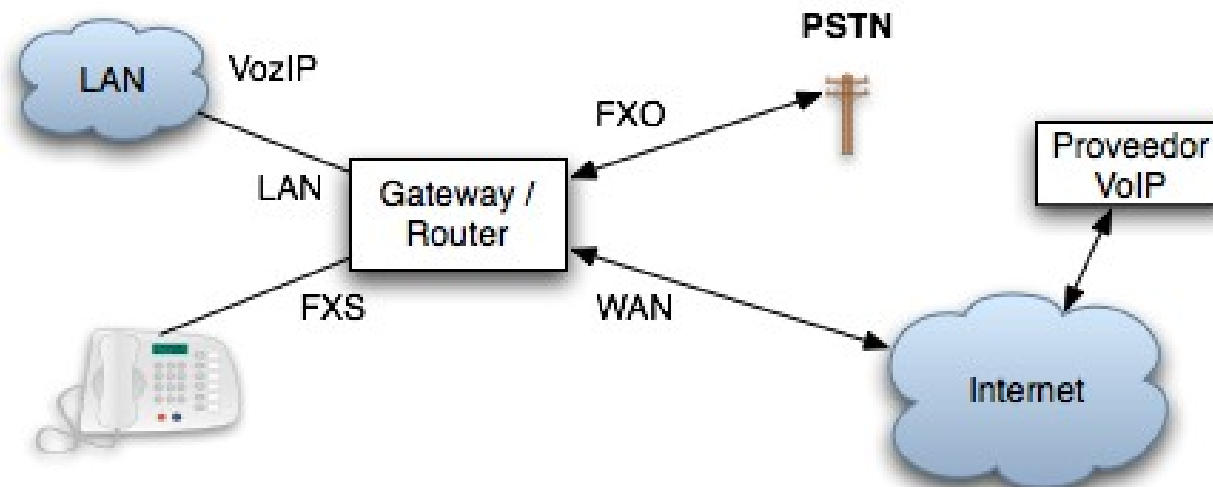
- Disponen 1 o más interfaces FXS para conectar teléfonos o líneas de enlace de centralitas



Gateways Voz IP

Gateways: FXO -> VoIP

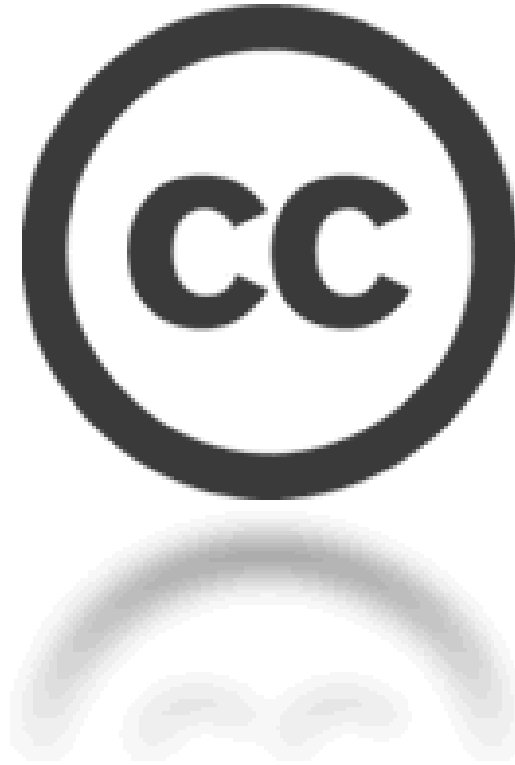
- Interfaz FXO para conectar una línea de operador.
- Tres funciones principalmente:
 - Discriminar en salida: llamar por IP o PSTN
 - Utilizar la línea como backup, es decir, en caso de fallo de Internet o del proveedor VoIP, las llamadas pueden ser encaminadas por PSTN tradicional.
 - Recibir llamadas por PSTN y encaminarlas por VoIP.



- Iñaki Baz (ibc)
- Jon Bonilla (manwe)
- Gorka Gorrotxategi (zgor)
- Saúl Ibarra (saghul)
- David Santamaría (highwayman)
- Igor Ruiz-Agundez

¡Muchas gracias a todos!





<http://creativecommons.org/licenses/by-sa/3.0/>



ironitec