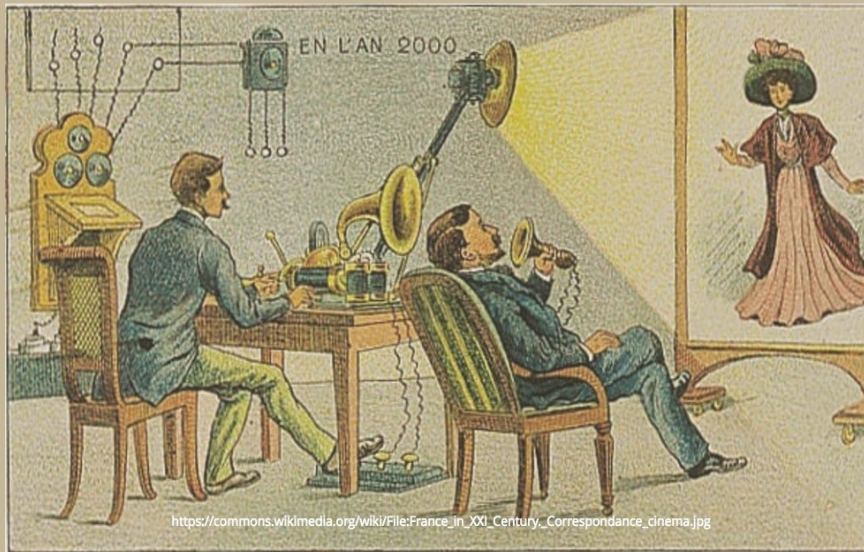




SIP

Session Initiation Protocol

**Contexte, protocole,
analyses et mise en oeuvre**

François-Emmanuel GOFFINET

Table of Contents

Introduction	1.1
[Avant-propos: README.md]	1.2
I. Contexte VoIP	1.3
1. POTS	1.3.1
2. Protocoles Multimédia	1.3.2
3. Marchés VoIP	1.3.3
4. Connexion SIP	1.3.4
5. Infrastructure VoIP	1.3.5
6. Migration VoIP	1.3.6
7. Conception VoIP	1.3.7
8. Aperçu des logiciels Open Source	1.3.8
9. Exemples de périphériques SIP	1.3.9
10. Exercices de mise en œuvre de l'infrastructure physique	1.3.10
II. Analyse Wireshark	1.4
1. Analyseurs de paquets	1.4.1
2. Analyse de paquets	1.4.2
3. Placement de l'analyseur de paquet	1.4.3
4. Introduction à Wireshark	1.4.4
5. Menus de Wireshark	1.4.5
6. Capture de paquets	1.4.6
7. Travailler avec des captures Wireshark	1.4.7
8. Statistiques Wireshark	1.4.8
9. Analyse VoIP Wireshark	1.4.9
10. Wireshark en ligne de commande	1.4.10
III. Protocole SIP	1.5
1. Architecture SIP	1.5.1
2. Aperçu des opérations SIP	1.5.2
3. INVITE SIP UAC/UAS	1.5.3
4. Réponses SIP	1.5.4
5. SDP	1.5.5
6. Enregistrement REGISTER	1.5.6
7. Proxy SIP UDP	1.5.7
8. B2BUA	1.5.8
9. Flux SIP Trapézoidal	1.5.9
10. Extensions SIP	1.5.10
11. Sécurité SIP	1.5.11
12. RFCs SIP	1.5.12

IV. Asterisk PBX	1.6
1. Solution FreePBX	1.6.1
2. Asterisk Core PABX	1.6.2
3. Asterisk Base	1.6.3
4. Asterisk Intermédiaire	1.6.4
5. Asterisk Avancé	1.6.5

Protocole SIP

Auteur : François-Emmanuel Goffinet

Titre : Le Protocole SIP, Session Initiation Protocol, Contexte, protocole, analyses et mise en oeuvre.

Description : Support de formation sur le protocole SIP.

version **1.0** droits **CC-BY-SA** Auteur **F.-E. Goffinet**

Sources Github : <https://github.com/goffinet/protocole-sip>

-  [Format HTML](#)
-  [Format PDF](#), compilation Tue May 15 2018 00:04:50 GMT+0200 (CEST)
-  [Format EPUB](#), compilation Tue May 15 2018 00:04:50 GMT+0200 (CEST)
-  [Format MOBI](#), compilation Tue May 15 2018 00:04:50 GMT+0200 (CEST)

sip.goffinet.org



Le contenu de ce support est en révision. version **in revision**

Logiciels et fichiers nécessaires

- [des captures d'exemples](#)
- Le soutien d'une présentation "Téléphonie IP, État de l'Art"
- [Une série de binaires dont certains ne devraient pas se trouver sur une station de travail professionnelle](#) :
 - Wireshark-win64-2.2.5.exe
 - WiresharkPortable_2.2.5.paf.exe
 - Linphone-3.10.2-win32.exe
 - Zoiper_3.9_Setup.exe
 - nmap-7.60-setup.exe
 - ca_setup.exe

Programme

Le canevas de ce support de formation est le suivant.

Architecture de SIP

Rappel sur la téléphonie classique

- Le réseau PSTN
- Les composants d'un réseau VoIP
- Les clients
- Les passerelles voix

La signalisation

- Comparaison à H.323
- Les serveurs d'application
- Les composants spécifiques au protocole SIP
 - Les UA
 - UAC
 - UAS
- Les proxy
- Les registrar

Les messages SIP

- INVITE
- REGISTER
- ACK
- BYE
- CANCEL
- UPDATE

Les codes de statuts

- Codes provisionnels
- Codes de réussite
- Codes de redirection
- Code d'échec client
- Code d'échec serveur

- Code d'échec global

Le protocole SDP

La qualité de service

Processus d'enregistrement

- Enregistrement basique
- Messages de notification
- Les WMI
- Configuration du registrar
- Configuration des clients
- Validation de l'enregistrement

Session SIP

- Initiation du dialogue
- Requête client
- Réponse du serveur
- Routage de l'appel
- Perte du routage
- Routage strict
- Modification de la session SIP
- Terminaison de la session SIP
- Utilisation d'un proxy
- Connexion au réseau PSTN
- Gestion de la présence

Les extensions SIP

- Principe et rôles des extensions
 - Info
 - Comet
 - Subscribe
 - Publish
 - Notify
 - Message
 - Refer
 - Prack

SIP et la sécurité

- Support du NAT
- Utilisation de Firewall
- Les types d'attaques sur les réseaux convergents
- Le hijacking
- Déni de service
- Amplification
- Bots
- Les mesures de sécurité

- Contrôle des accès
 - Cryptage
 - Authentification
 - Gestion des autorisations
 - Systèmes de détection d'intrusion
-

Photo de couverture : Correspondance Cinéma - Phono - Télégraphique. Chromolithographie. Villemard, 1910.
(https://commons.wikimedia.org/wiki/File:France_in_XXI_Century._Correspondance_cinema.jpg)

Introduction au contexte VoIP et des communications unifiées

Ce chapitre est une introduction aux sujets d'analyse VoIP avec Wireshark, sur le protocole SIP ou sur un logiciel de téléphonie.

En voici le sommaire :

- [1. POTS](#)
- [2. Protocoles Multimédia](#)
- [3. Marchés VoIP](#)
- [4. Connexion SIP](#)
- [5. Infrastructure VoIP](#)
- [6. Migration VoIP](#)
- [7. Conception VoIP](#)
- [8. Aperçu des logiciels Open Source](#)
- [9. Exemples de périphériques SIP](#)
- [10. Exercices de mise en œuvre de l'infrastructure physique](#)

POTS (Plain Old Telephony Systems)

- 1. POTS
- 2. Commutation de circuit
- 3. Commutation de paquet
- 4. Boucle locale
- 5. Trunk
- 6. Signalisation dual-tone multi-frequency signaling (DTMF)
- 7. FXO/FXS
- 8. ISDN/RNIS
 - Définition
 - Accès de base BRI
 - Accès primaire PRI
- 9. Numérotation standard E.164
- 10. Signalisation SS7

1. POTS

POTS est un sigle anglais qui signifie Plain Old Telephone System que l'on peut traduire en français par "le bon vieux téléphone". Dans certains pays on parle de réseau fixe ou de téléphone fixe. Il s'agit en fait des services rendus par une ligne téléphonique analogique avant l'avènement des technologies ISDN, téléphone mobile, ADSL et VoIP.

Le service POTS existe depuis l'introduction du téléphone à la fin du XIXe siècle, sous une forme pratiquement inchangée pour l'utilisateur lambda malgré l'introduction de la numérotation par tonalité ou de la fibre optique en remplacement des fils de cuivre qui composaient les lignes. Outre la communication de la voix à travers le réseau RTC, on peut citer ces services communs comme :

1. [Voicemail](#), service de boîte vocale
2. [Caller ID](#), service d'identification de l'appelant
3. [Call waiting](#), service de mise en attente
4. [Speed dialing](#), composition rapide
5. [Conference call](#) (three-way calling), chambres de conférences
6. [Enhanced 911](#), services d'urgences améliorés
7. [Centrex](#), central téléphonique
8. [Fac-simile \(FAX\)](#), télécopie
9. Communications digitales

Le réseau RTC/PSTN utilise des technologies, des supports et des protocoles différents de ceux qui supportent TCP/IP.

Les principaux concepts utiles à retenir sont :

1. Communications digitales mais à commutation de circuit
2. Architecture : Boucle locale/Trunks
3. Composition Pulse Dialing/DTMF
4. Trunks
5. Numérotation standard [E.164](#)
6. Signalisation SS7

Ils permettent de mieux comprendre les architectures de connexions d'accès aux téléphones des entreprises et des particuliers sur lignes fixes et mobiles. Des interfaces et des protocoles permettent de connecter les deux mondes RTC et VoIP.

2. Commutation de circuit

La commutation de circuit est un mode d'établissement d'une liaison de télécommunication pour laquelle :

- un chemin physique ou logique est établi entre deux équipements

et

- est bloqué pour toute la durée de la communication.

L'établissement de circuit est aujourd'hui exécutée de manière électronique.

Dans la commutation par circuit, il y a un risque de sous-utilisation du support en cas de "silence" pendant la communication.

RNIS (ISDN) est un exemple de technologie à commutation de circuit qui digitalise la voix en tant que service.

Le temps passé est facturé.

3. Commutation de paquet

La commutation de circuits s'oppose au principe de la commutation de paquets qui optimise le canal de transmission laissant le soin à des commutateurs intermédiaires d'acheminer les paquets (Ethernet, Wi-Fi, IP) ou en établissant des Circuits Virtuels (ATM, Frame-Relay).

Ces technologies sont facturées par quantité de données échangées.

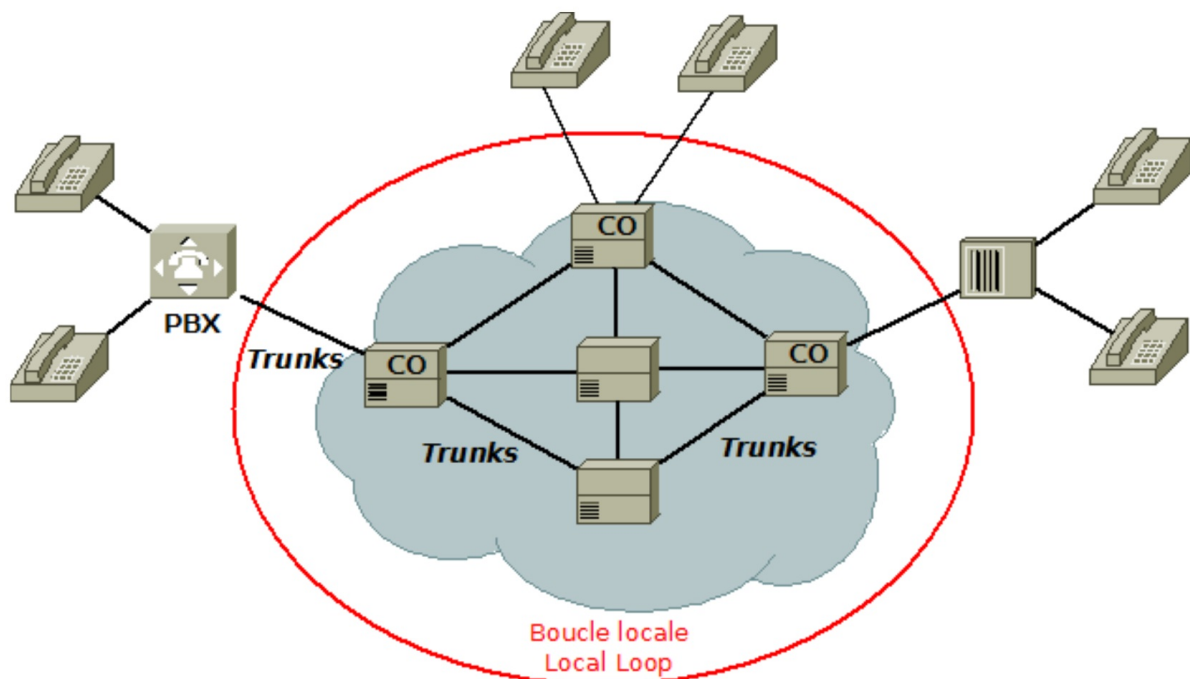
Le protocole MPLS permet de construire des réseaux IP cohérents sur ces architectures préexistantes avec une forme de confidentialité et de la gestion de qualité de service.

4. Boucle locale

Le réseau téléphonique commuté (RTC ou PSTN) peut transporter la voix mais aussi des données. Il utilise le principe de la commutation de circuit qui est celui de l'établissement d'un circuit dédié pour une communication.

Les téléphones sont connectés à des commutateurs téléphoniques (manuels, automatiques, électroniques) qui constitue des points d'échange téléphonique.

Ils sont situés chez l'opérateur au **central office (CO)** ou dans l'entreprise (en tant que Private Branch Exchange, PBX).



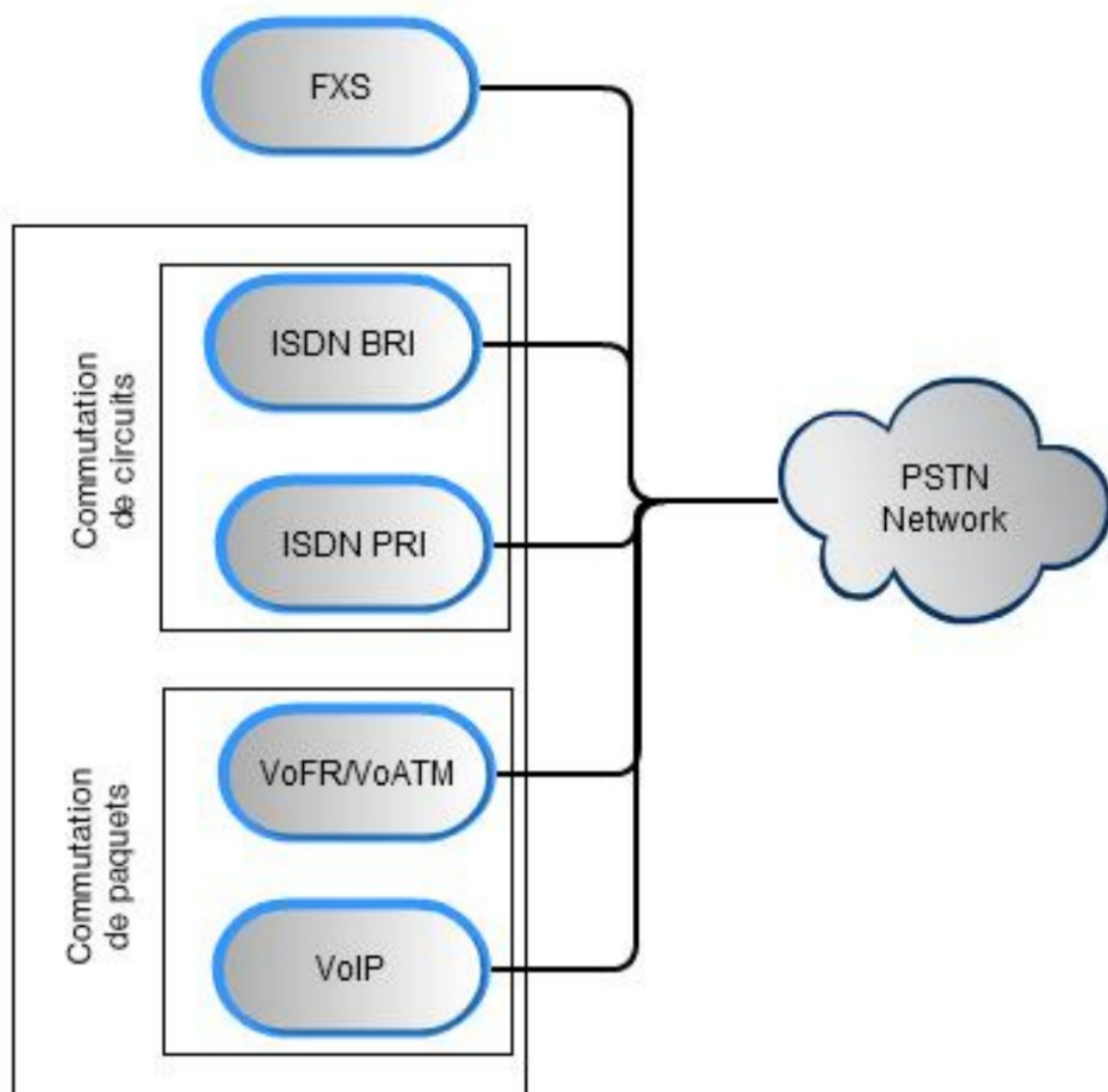
Ces commutateurs s'interconnectent entre eux via des **Trunks digitaux ou analogiques**.

Si le nuage opérateur fonctionne entièrement en technologies digitales, du client à l'opérateur il reste souvent une connexion analogique avec des fréquences audibles pour transmettre la voix et la signalisation (Dial Tone Multi Frequency, DTMF). Cette zone est appelée **boucle locale** du CO de l'opérateur au bâtiment du client par une paire (téléphonique) de fils en cuivre.

5. Trunk

Un trunk est une liaison spéciale vers le réseau PSTN. Les opérateurs proposent des services digitaux sur la boucle locale analogique et parmi ceux-ci le transport de la voix :

- sur ISDN
- mais aussi sur TCP/IP supporté par des technologies d'accès plus intéressantes telles que xDSL, DOCSIS (câble/Fibre), Metro Ethernet, etc.



Chaque communication vocale digitalisée prend 64 Kbps de bande passante.

Sur une technologie à commutation de circuits comme ISDN cette bande passante est monopolisée (même le silence est codé).

Sur une technologie à commutation de paquets, comme TCP/IP cette bande passante est optimisée en fonction de l'occupation.

6. Signalisation dual-tone multi-frequency signaling (DTMF)

La signalisation **Dual-tone multi-frequency signaling (DTMF)** est celle qui est utilisée "in band", c'est-à-dire dans le canal de communication, pour établir les destinations des appels auprès d'un central téléphonie privé ou public.

DTMF est utilisé sur les claviers alphanumériques téléphoniques qui ont peu à peu remplacé les cadran à disque.

Chaque touche correspond à deux fréquences audibles qui sont jouées simultanément.

Fréquences de touches DTMF :

	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	7	8	9	C
941 Hz	*	0	#	D

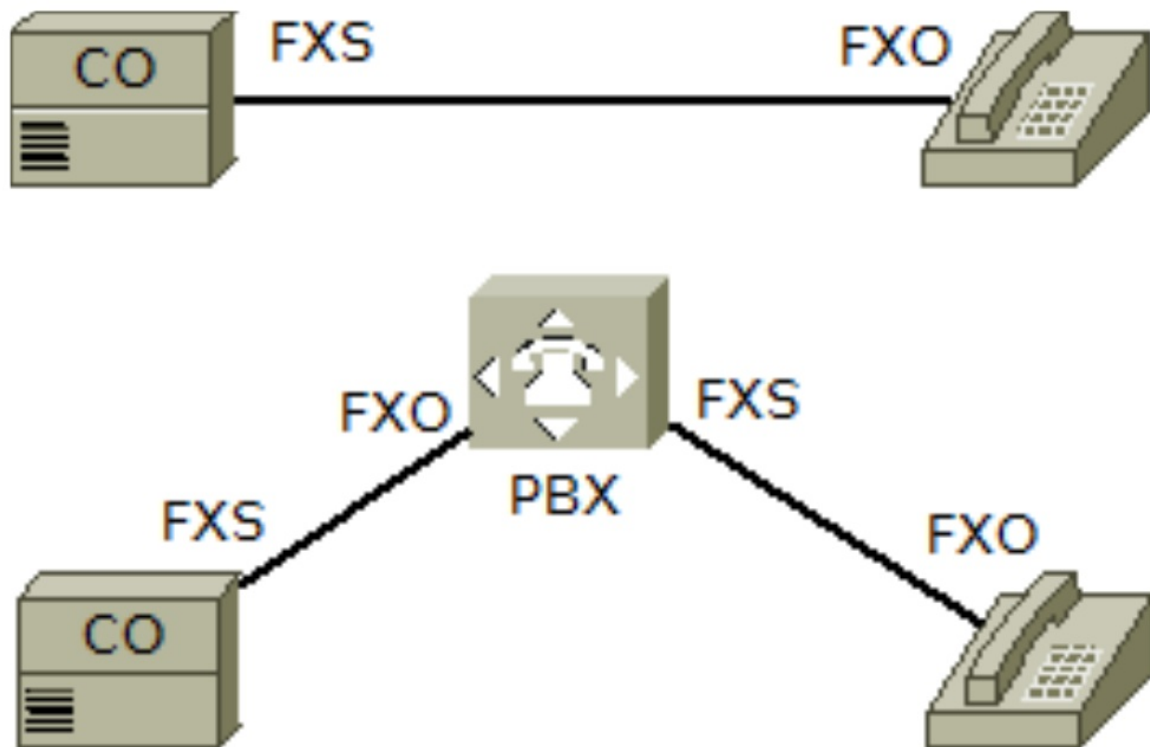
Certaines fréquences audibles sont réservées, en Belgique :

	Fréquence en Hz	Cadences en secondes
Busy tone	425	0.5 on 0.5 off
Congestion tone	425	0.167 on 0.167 off
Dial tone	425	continuous
Special dial tone	425	1.0 on 0.25 off
Holding tone	1400	0.4 on 15.0 off
Ringing tone	425	1.0 on 3.0 off
Special information tone	900/1400/1800	3x0.33 on 1.0 off
Call waiting tone	1400	0.175 on 0.175 off 0.175 on 3.50 off

7. FXO/FXS

En téléphonie, un **Foreign eXchange Station ou FXS** est une interface téléphonique qui **fournit la tonalité**, le courant de charge et la tension électrique nécessaire pour faire fonctionner la sonnerie.

Un périphérique qui connecte un FXS est une interface FXO qu'elle soit un téléphone analogique ou l'interface d'un PBX pour recevoir des appels. L'interface **Foreign eXchange Office** est le port qui reçoit la ligne analogique.



8. ISDN/RNIS

Définition

ISDN (Integrated Services Digital Network), RNIS en français, est une technologie qui utilise la boucle locale pour fournir divers services de transmission digitales de la voix, de la vidéo et des sonnées. ISDN permet de se connecter au réseau PSTN, à Internet, à des sites distants.

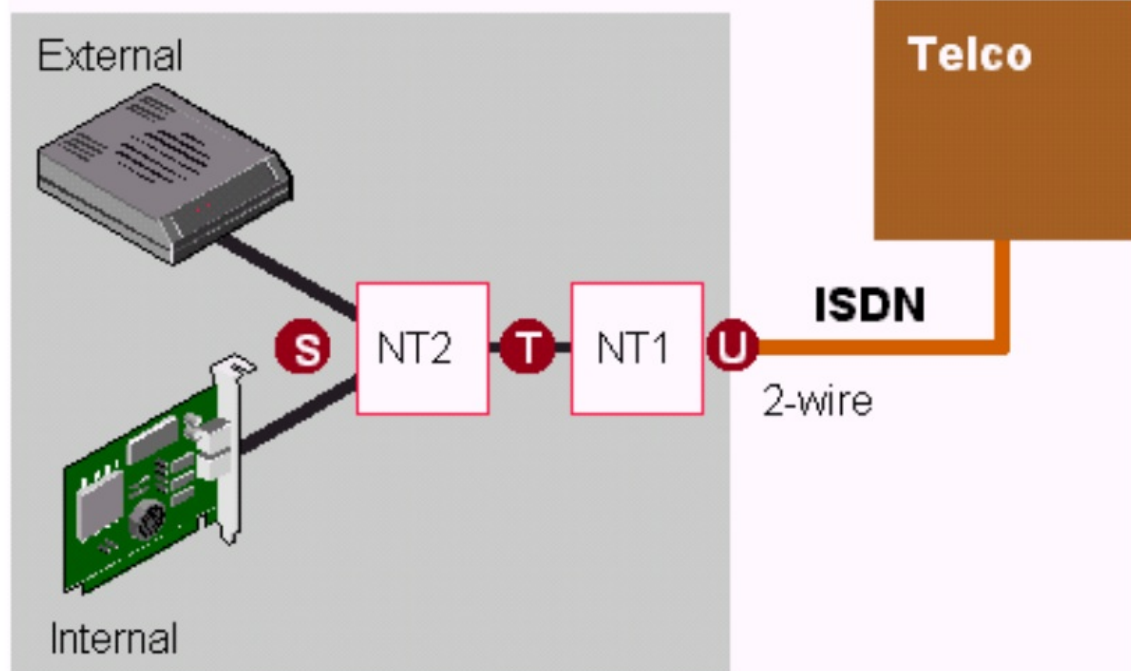
ISDN fournit plusieurs type de canaux numériques :

1. Canaux B de 64 Kbps pour la voix et les données (1 canal B = un canal voix)
2. Canal D de 16 Kbps pour la signalisation.

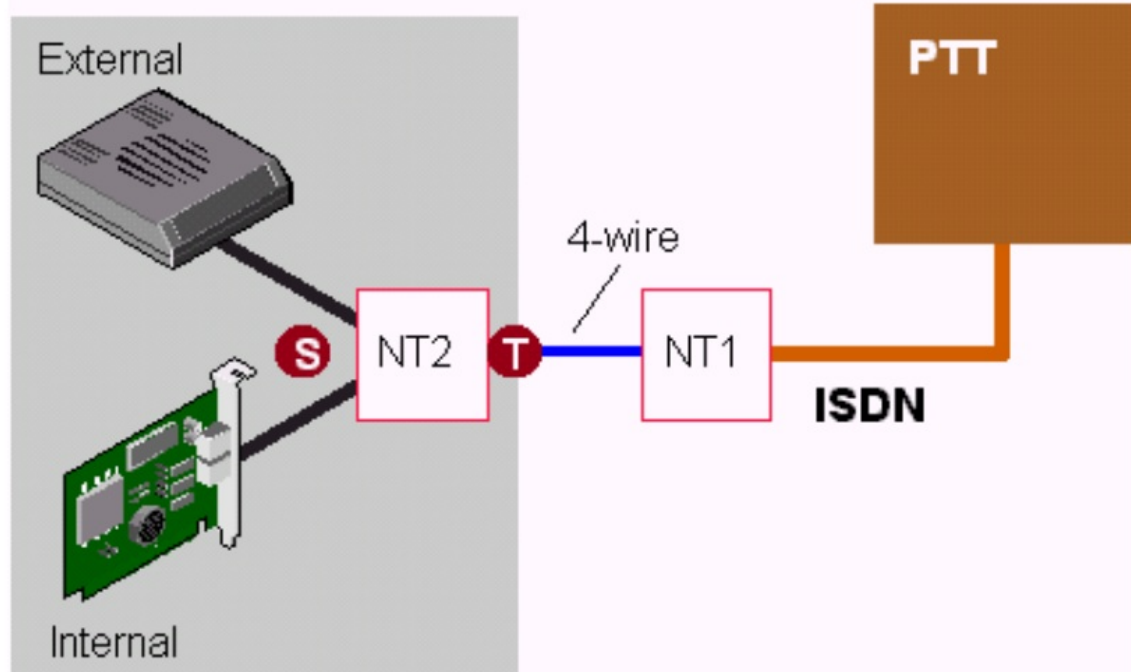
Accès de base BRI

L'accès de base ou Basic Rate Interface (BRI ou T0) comprend deux canaux B et un canal D. Soit un total de deux canaux voix simultanés ou 128 Kbps de bande passante.

Terminal Adapter (U.S.)



Terminal Adapter (Europe, Japan, etc.)



Source : Cisco Systems

Accès primaire PRI

L'accès primaire (PRA) ou Primary Rate Interface (PRI ou T2) propose trente canaux B et un canal D. Soit un total de 30 canaux voix simultanés ou 2 Mbps de bande passante. Ce type de ligne à 2 Mbps se vend aussi comme ligne dite E1.

9. Numérotation standard E.164

E.164 est une recommandation ITU-T qui définit le plan de numérotation international utilisé dans le réseau PSTN. Il définit aussi le format des numéros de téléphone de maximum 15 chiffres et habituellement écrits avec un sigle + en préfixe.

10. Signalisation SS7

Signaling System #7 (SS7) ou système de signalisation #7 est un ensemble de protocoles de signalisation téléphonique qui sont utilisés dans la grande majorité des réseaux téléphoniques mondiaux. SS7 fournit, dans le réseau téléphonique, une structure universelle pour :

1. la signalisation,
2. l'envoi de messages,
3. l'interconnexion et la maintenance réseau.

Il gère

1. l'établissement d'appels,
2. l'échange d'informations utilisateur,
3. le routage d'appels, la facturation et
4. supporte les services du réseau intelligent (en anglais Intelligent Network (IN))

L'utilisation principale de SS7 est de délivrer un appel téléphonique à travers le RTC. L'appel peut avoir à traverser plusieurs réseaux de différents opérateurs téléphoniques. (Par exemple sur un appel international ou s'il y a plusieurs opérateurs nationaux...). À chaque étape sur le chemin de l'appel les commutateurs téléphoniques ont besoin de savoir d'où vient l'appel (quelle ligne téléphonique, quel canal ou quel circuit) et vers où il va. Cela nécessite beaucoup de coordination. ISUP (ou ISDN User Part signaling) est un type de communication SS7 qui s'occupe de l'établissement d'un appel tout au long de ces différents liens. Les messages ISUP sont passés de commutateur en commutateur et à chaque point du circuit l'appel est étendu un peu plus jusqu'à l'aboutissement de l'appel (établissement de bout en bout).

Remarque : En VoIP, à l'échelle des communications globales, le protocole IETF SIP a l'ambition de remplacer SS7.

Protocoles Multimédia

- 1. Voix et vidéo sur IP
- 2. Voice over Internet Protocol (VoIP)
 - Architecture VoIP
 - Protocoles VoIP
- 3. Le protocole RTP
- 4. Signalisation VoIP
- 5. Protocole SIP
- 6. ENUM
- 7. XMPP
- 8. NAT Traversal
 - STUN
 - TURN
 - ICE
- 9. Enregistrement DNS SRV
 - Format DNS SRV
 - Vérification de SRV DNS
- 10. Fax T.38

1. Voix et vidéo sur IP

- Les services Internet comme Facetime, Skype, Webex, ...
- Les CPE (https://fr.wikipedia.org/wiki/Customer_Premises_Equipment), soit les "box" des opérateurs domestiques comme la Bebox de Belgacom
- Les services de cartes pré-payées avec des tarifs avantageux pour certaines destinations.

... ont popularisé les technologies Voix et vidéo sur IP.

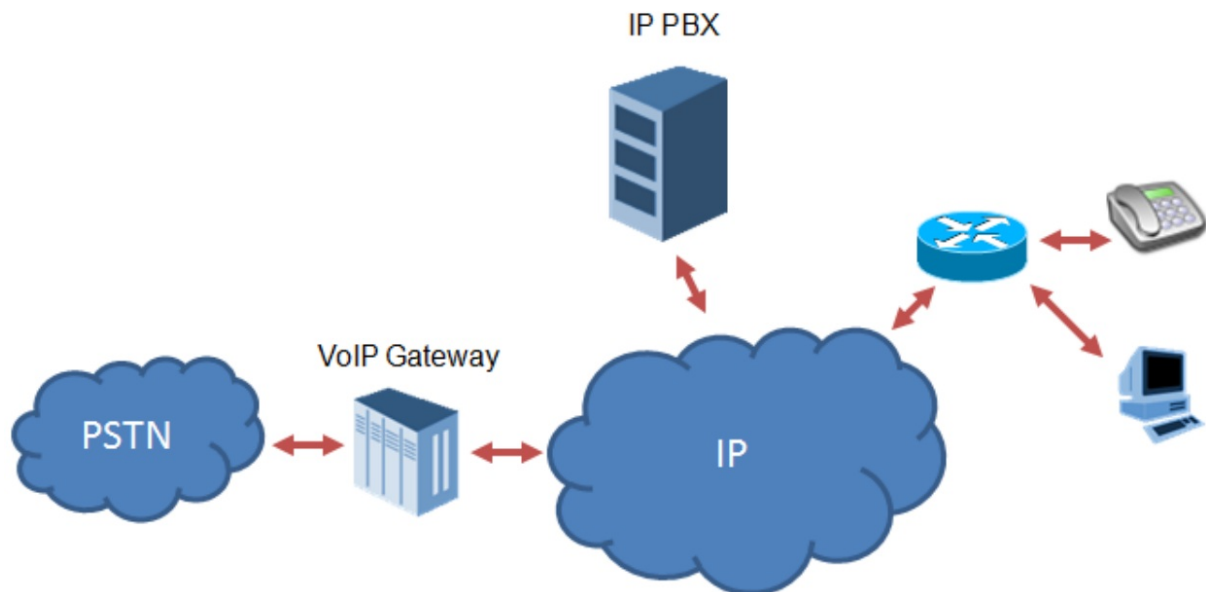
2. Voice over Internet Protocol (VoIP)

La Voice over Internet Protocol (VoIP) est l'usage des réseaux Internet pour assurer des services téléphoniques. Ce récent paradigme technologique :

1. Se base sur l'infrastructure de données informatiques : on parle de convergence IP. TCP/IP
2. remplace la téléphonie analogique classique, l'ensemble des technologies ISDN d'entreprise ou d'opérateur.
3. Supporte le service téléphonique de type POTS comme n'importe quelle application du réseau.
4. S'intègre et complète les nouveaux services de communications unifiées et mobile.

Architecture VoIP

Le diagramme suivant représente un réseau qui utilise la même infrastructure pour la voix et les données.



On retrouve principalement :

- des éléments d'infrastructure du réseau LAN/WAN qui transportent le trafic : des routeurs, des commutateurs, des lignes, mais aussi des services classiques : DHCP, DNS, QoS, etc.
- des téléphones IP qui codent la voix : embarqués, logiciels, sur tablette ou smartphone.
- Un central téléphonique IP (matériel, logiciel, composé de plusieurs éléments, virtualisé, dans le Cloud).
- Des protocoles propres à la VoIP remplissant divers fonctions de contrôle et de transport des médias.
- Des Trunks passerelles VoIP/PSTN indépendantes (comme ici) ou installées dans l'IP PBX, voire un Trunk SIP vers le réseau PSTN.

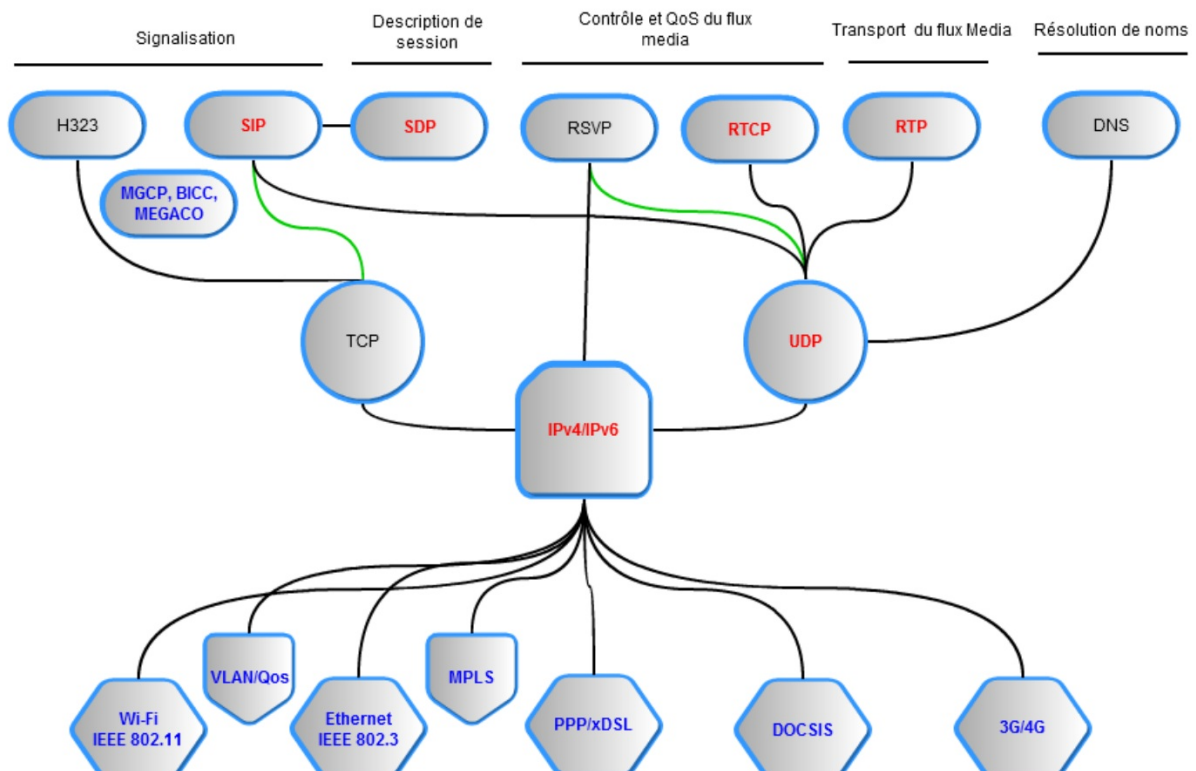
Pour offrir un service de qualité similaire à ce qu'offre la commutation de circuit, les réseaux IP doivent répondre aux défis que représente la transmission du trafic en temps réel. En effet, TCP/IP est conçu pour transmettre des données qui supportent les délais, les accusés de réception, etc. Pour qu'une transmission vocale soit audible, le réseau IP ne doit pas dépasser certains critères de délais, de pertes et de stabilité du flux. La gestion du QoS peut s'avérer nécessaire si ces critères de qualité ne sont pas respectés. Aussi, une architecture du réseau robuste et évolutive permet de s'adapter aisément à ces critères. Dans le LAN, on utilisera l'étiquetage de trame (IEEE 801.1q/802.1p) ou DSCP/RSVP dans un internet.

Le transport des médias est un élément critique qui ne supporte pas les retransmission et les délais. On peut connaître différents problèmes lors d'une transmission RTP :

1. **Packet loss – Perte de paquets.** Quand le seuil de 5% de perte est atteint, on peut considérer une dégradation significative de la qualité de la voix.
2. **Delay – Délais :** le temps requis pour que les paquets arrivent à destination. Des délais supérieurs à 150 ms affectent également la qualité perceptible de la voix.
3. **Jitter (delay variation) – Gigue :** variation de délais. Trop de variabilité affecte également la qualité des transmissions vocales. Pour atténuer ce problème, on utilise des tampons de gigue.

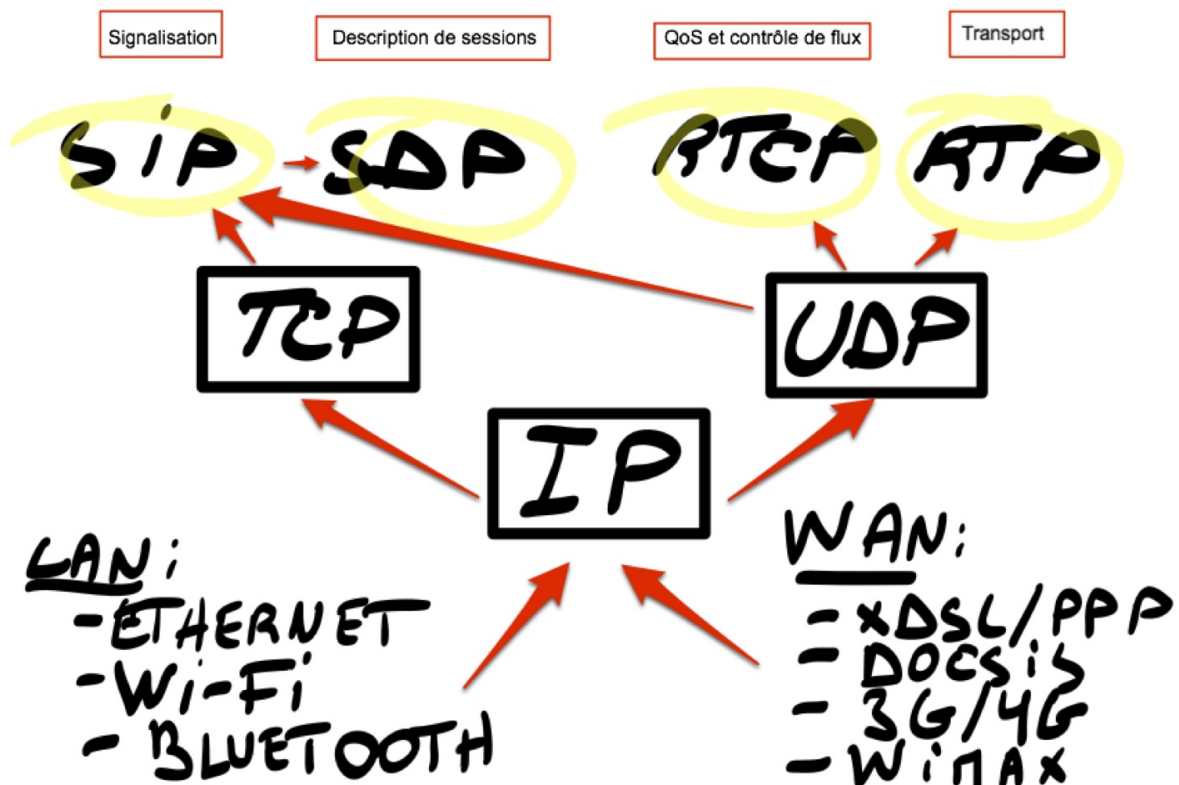
Protocoles VoIP

Le transport et la signalisation des services associés à la VoIP reposent sur plusieurs protocoles. Ils sont formalisés dans la pile des protocoles TCP/IP par l'IETF. TCP/IP est le nom donné au modèle de communication.



Ce modèle se base sur les principaux protocoles des deux couches fonctionnelles (Internet et Transport) qui fournissent des services (couche Application) sur une multitude de technologies (couche Accès Réseau). Ces dernières sont aveugles à l'égard de TCP/IP.

Au milieu, pour assurer des services sur des connexions de données, on trouve de manière centrale le protocole IP. La liaison vers les applications se réalise via un protocole de transport orienté connexion tel que TCP ou non orienté connexion tel que UDP.



Typiquement dans une communication VoIP, différents protocoles interviennent quel que soient les technologies LAN ou WAN traversées d'une extrémité à l'autre du réseau :

1. Pour la signalisation, par exemple : IP+UDP+SIP(+SDP)+charge
2. Pour le transport de la voix, de la vidéo, etc. : IP+UDP+RTP+charge

D'autres protocoles interviennent mais il ne sont pas exclusivement destinés à l'usage VoIP, on devrait les trouver dans tout réseau bien géré :

1. Pour la gestion du réseau au niveau global : DNS mais aussi NTP, HTTP, FTP, TFTP, DHCP, etc.
2. Pour la gestion du réseau au niveau LAN/WAN : IEEE 802.1q (VLAN) et IEEE 802.1p (QoS) sur une infrastructure LAN commutée.

3. Le protocole RTP

RTP définit une méthode standardisée de transport réseau pour transmettre des échantillons de voix et de vidéo sur le réseau Internet. RTP, protocole de transport pour applications en temps-réel, est formalisé dans le [RFC 3550](#).

RTP connaît d'autres applications que la téléphonie telles que la messagerie instantanée, l'IPTV, etc. RTP n'est pas un protocole fermé et complet mais plutôt un cadre pour spécifier d'autres protocoles.

Concrètement, RTP n'est pas responsable de la signalisation (SIP ou H.323 par exemple), de la gestion du QoS (RTCP), de la négociation des protocoles d'encodage (SDP) ou de la conversion analogique/digital (encodeurs).

La transmission s'effectue d'un téléphone IP à un autre de bout en bout, sans intermédiaire, dans une session ou en streaming.

RTP est en fait composé de deux protocoles, RTP à proprement dit et un sous-protocole RTCP qui se charge d'échanger des informations sur la QoS pour une session.

Bien que RTP se veuille indépendant des protocoles sous-jacents, il utilise largement le protocole UDP. Il peut utiliser RTSP (basé TCP) ou DCCP.

En effet, le fonctionnement simple et léger d'UDP permet de se passer de délais supplémentaires qui ne conviennent pas au transport de la voix.

Secure Real-time Transport Protocol (ou SRTP) définit un profil de RTP qui a pour but d'apporter le chiffrement, l'authentification et l'intégrité des messages, et la protection contre le replay de données RTP en unicast et multicast (RFC 3711).

Veuillez analyser la capture RTP et reconstruire la transmission vocale :

<https://www.cloudshark.org/captures/71c1b394bbbf>.

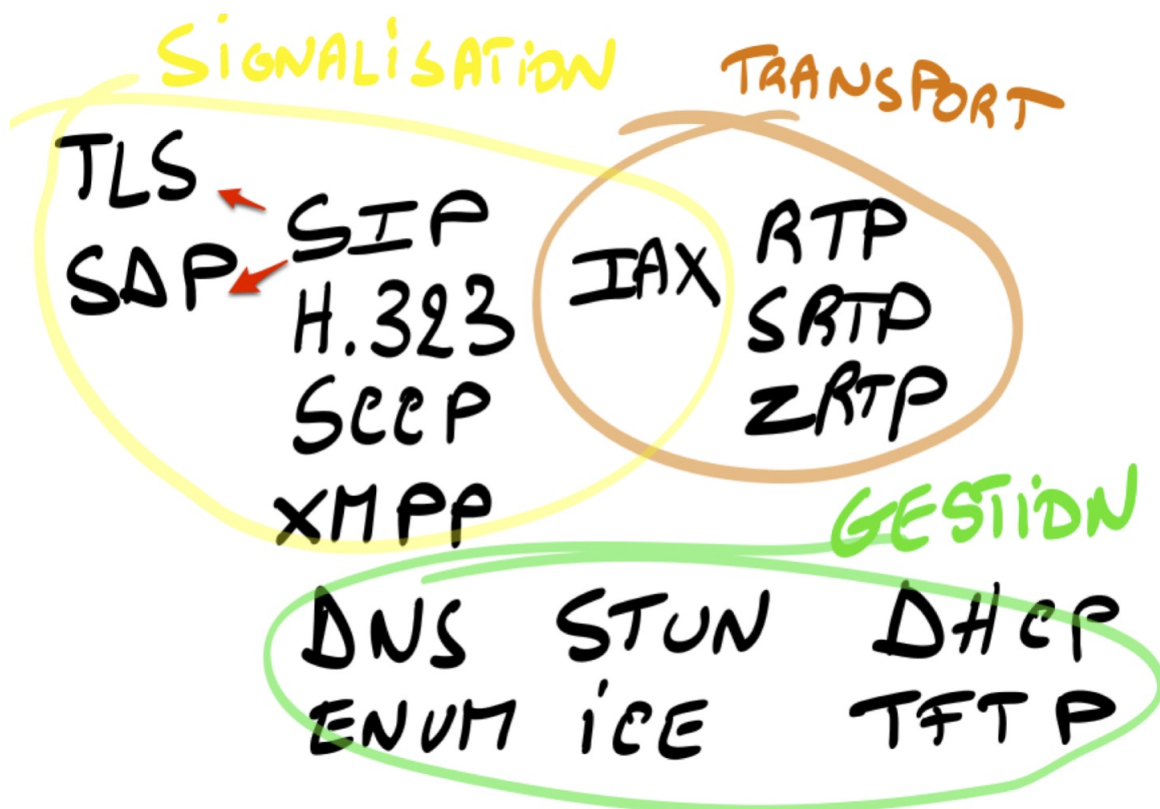
4. Signalisation VoIP

Un protocole de signalisation est celui qui permet d'établir, maintenir et fermer une communication entre deux postes (téléphoniques) terminaux.

En VoIP, on trouvera plusieurs protocoles de signalisation :

1. SIP
2. H.323
3. IAX
4. Skinny/SCCP
5. MGCP
6. MEGACO
7. BICC

SIP s'impose comme le protocole majeur dans les déploiement VoIP.



5. Protocole SIP

Session Initiation Protocol (dont l'abréviation est SIP) est un protocole normalisé et standardisé par le [RFC 3261](#), et est complété par le [RFC 3265](#). Il a été conçu pour établir, modifier et terminer des sessions multimédia.

- Il se charge de l'authentification et de la localisation des multiples participants.
- Il se charge également de la négociation sur les types de média utilisables par les différents participants en encapsulant des messages SDP (Session Description Protocol).
- Il se charge également de prendre en compte la disponibilité (présence) des utilisateurs.

SIP ne transporte pas les données échangées durant la session comme la voix ou la vidéo.

SIP étant indépendant de la transmission des données, tout type de données et de protocoles peut être utilisé pour cet échange.

Cependant le **protocole RTP (Real-time Transport Protocol)** assure le plus souvent les sessions audio et vidéo.

Dans le contexte de la Téléphonie IP, SIP remplace progressivement H.323 et dans une autre mesure SS7.

Il est supporté par UDP (communément) ou TCP sur le port 5060 et/ou 5061 (sécurisé par TLS). Il est supporté aussi bien par IPv4 et IPv6. Il ne fonctionne pas sans d'autres protocoles TCP/IP qui le soutienne dans sa tâche de signalisation.

SIP est protocole de **couche Application**. Son fonctionnement sera plus aisé si on s'abstient de regarder les couches inférieures. Toutefois, la bonne compréhension des architectures SIP nécessite que l'on s'occupe des paramètres TCP/IP.

SIP est une boîte à outils pour développer sur le marché des solutions de communication en temps réel chez les particuliers et les entreprises via des fournisseurs historiques ou via des nouveaux arrivants à partir des marchés des télécommunications vers celui de l'Internet (des objets).

Mais le protocole SIP est ouvert à bien d'autres usages ou contextes. Il peut être utilisé *de facto* comme protocole de communication au sein des solutions de back-end vocal (régie des émissions TV), protocole de liaison en direct pour de la transmission TV, dans des solutions de domotique, de télé-présence, étant supportant sur n'importe quel objet TCP/IP ...

L'idée fondamentale de SIP est d'offrir la possibilité d'établir, de maintenir et de terminer une session quelle que soit son contenu (voix, vidéo, commande, présence, message, texte) entre un ou plusieurs **et** un ou plusieurs hôtes TCP/IP.

Ces ressources TCP/IP sont des ressources logiques identifiées par un URL tel que l'on trouve des URLs pour identifier des ressources SIP, de type `sips:goffinet@goffinet.org` ou `sips:francois@test.tf`, soit :

- un protocole,
- un identifiant
- dans un domaine

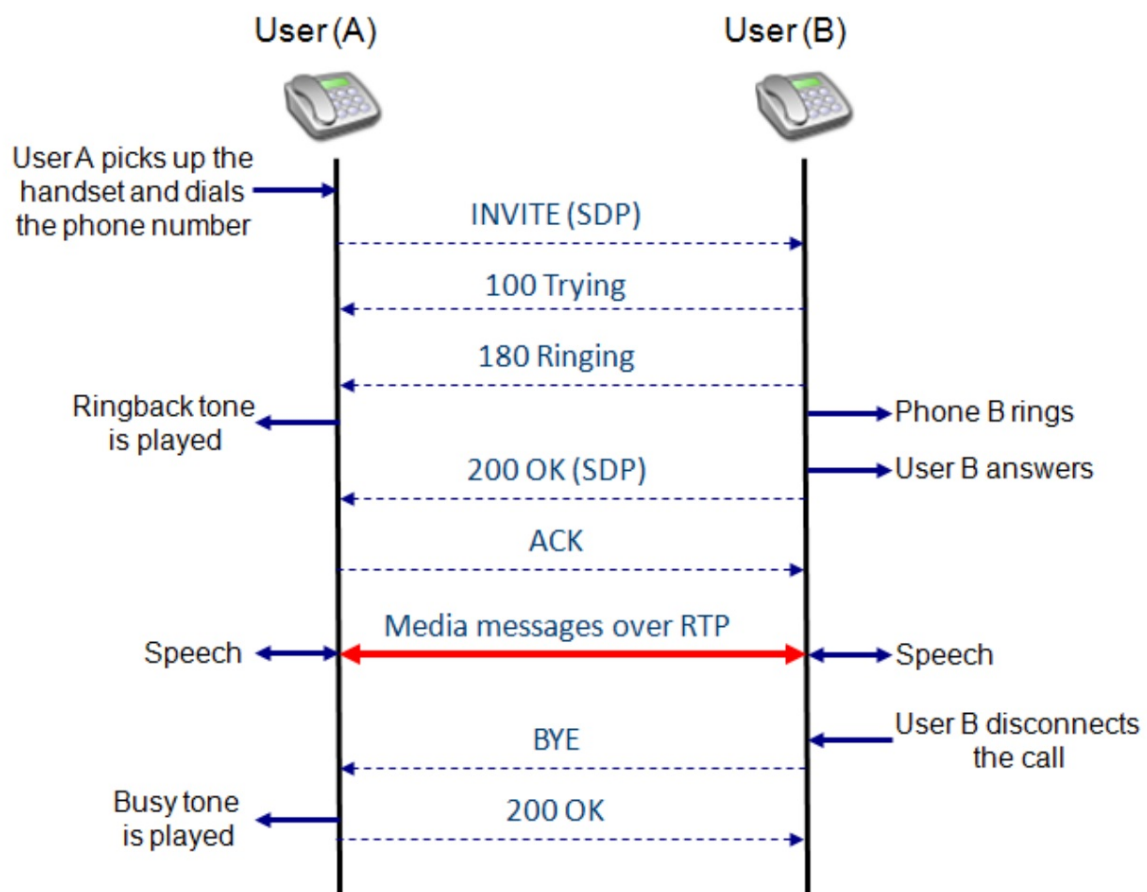
en sachant que l'identifiant et son domaine représente une machine probablement joignable en TCP/IP soit disposant d'une adresse IP.

Un premier rôle fondamental est celui d'agent utilisateur (UA, User Agent) qui prend le nom de client (UAC) s'il établit l'appel ou de serveur (UAS) s'il reçoit l'appel.



Dès qu'un appel SIP aboutit, les hôtes d'extrémité devraient être capables de se transmettre un message dans un format négocié au préalable de l'appel grâce une charge SDP (Session Description Protocol). + RFC

Sur le plan fondamental, une session SIP pour être illustré par un appel direct entre deux machine TCP/IP connectées au même réseau :



On peut trouver un exemple de cette transaction entre 10.33.6.101 (UAC) et 10.33.6.100 (UAs) sur <https://www.cloudshark.org/captures/7d2a7dab3e8b>.

On constate un échange de méthodes (INVITE, ACK, BYE) et de réponses (100, 180, 200)

Ce type de procédure fonctionne dans un monde idéal où toute ressource s'identifie par adresse IP et que celle-ci est parfaitement localisée.

Or les utilisateurs font appel à un service pour trouver les destinataires SIP. Ces services peuvent **authentifier** les utilisateurs, **localiser** les correspondants et les passerelles, **transférer** les messages d'établissement d'appels jusqu'à leur destination, les **réécrire**, les **rediriger** ou même les **traduire**.

SIP propose une série de rôles protocolaires tels que : REGISTRAR, PROXY, REDIRECT SERVER, B2BUA, SBC, ... qui permettent aux messages SIP de trouver leur destinataire. Ces rôles peuvent être comparés à ceux que l'on trouve au niveau des couches TCP/IP : Switches, Routeurs, Pare-feux et contrôleurs divers.

A un niveau d'autant plus applicatifs, on peut trouver n'importe quel service qui prendra en charge un appel vidéo ou vocal tel que de l'IVR, du *text-to-Speech*, du chiffrement de signalisation et de transport, des services de passerelles vers le PSTN, des services de centre d'appel, un couplage à l'informatique d'entreprise, des services IP-PBX et Softswitch, ...

Aussi, les utilisateurs identifient leur ressources par des noms sous un format commun qui correspond à une adresse de courrier électronique : `sips:goffinet@goffinet.org`

DNS est utilisé via des résolutions NAPTR et SRV pour trouver l'endroit de livraison du message d'appel.

Parce que les adresses IP sont embarquées dans les messages SIP, la traduction des adresses au passage des routeurs NAT (Network Address Translation) peut poser des problèmes.

Ceci dit, différentes solutions peuvent être envisagées pour déployer des solutions qui contournent ces écueils :

1. Se fonder sur IPv6
2. Utiliser des solutions basées sur une implémentation des services ICE, STUN ou TURN
3. Utiliser des passerelles et/ou des protocoles qui embarquent le trafic entre les correspondants

6. ENUM

ENUM, spécifié dans le [RFC 3761](#), est un mécanisme permettant d'utiliser un numéro de téléphone comme clé de recherche dans le DNS pour trouver la manière de joindre une personne ou une autre entité. Les hypothèses de départ sont :

1. qu'il existe déjà une infrastructure pour allouer les numéros de téléphone et que des milliards sont déjà attribués,
2. que ces numéros sont des clés "naturelles" pour beaucoup de gens.

ENUM s'appuie sur le système DDDS ('Dynamic Delegation Discovery System', RFC 3401) et sur les enregistrements NAPTR du DNS.

Une recherche ENUM par un résolveur ENUM commence par un numéro de téléphone à la norme UIT E.164, comme +33 1 23 45 67 89.

Ce numéro est transformé en nom de domaine en l'inversant (comme on fait pour trouver un nom de domaine à partir d'une adresse IP, ici, cela donnerait 9.8.7.6.5.4.3.2.1.3.3.e164.arpa).

On cherche alors les enregistrements NAPTR pour ce nom de domaine. Si on trouve (ici, avec dig) :

```
% dig NAPTR 9.8.7.6.5.4.3.2.1.3.3.e164.arpa
```

```
NAPTR 10 100 "u" "E2U+sip" "!^.*$!sip:francois@test.tf!" .
```

```
NAPTR 10 102 "u" "E2U+msg" "!^.*$!mailto:goffinet@goffinet.org!" .
```

Cela signifie que le titulaire du numéro de téléphone +33 1 23 45 67 89 peut être joint en SIP à `francois@test.tf` et par courrier électronique à `goffinet@goffinet.org`.

La racine de ENUM est donc e164.arpa, géré politiquement par l'UIT et techniquement par le RIPE-NCC. On la nomme 'Tier 0' dans ENUM. Les 'Tier 1' sont les organismes nationaux d'attribution des numéros de téléphone et les 'Tier 2' les opérateurs. Ensuite, on peut déléguer comme on veut, comme avec n'importe quel domaine.

7. XMPP

Extensible Messaging and Presence Protocol (qu'on peut traduire par « Protocole extensible de présence et de messagerie »), souvent abrégé en XMPP, est un ensemble de protocoles standards ouverts de l'Internet Engineering Task Force (IETF) pour la messagerie instantanée, et plus généralement une architecture décentralisée d'échange de données. XMPP est également un système de collaboration en quasi-temps-réel et d'échange multimédia via le protocole Jingle, dont la Voix sur réseau IP (téléphonie sur Internet), la visioconférence et l'échange de fichiers sont des exemples d'applications.

XMPP est constitué d'un protocole TCP/IP basé sur une architecture client-serveur permettant les échanges décentralisés de messages instantanés ou non, entre clients, au format Extensible Markup Language (XML). XMPP est en développement constant et ouvert au sein de l'IETF.

Les serveurs peuvent être privés (en intranet, par exemple Facebook) ou bien reliés entre eux au sein du réseau Jabber. XMPP est ainsi utilisé à travers le monde par des centaines de serveurs publics et privés, et des millions d'utilisateurs. De nombreux acteurs industriels utilisent XMPP, comme Apple, Cisco, Gizmo5, GNOME, Google, IBM, Oracle Corporation, etc.

Le protocole XMPP est séparé en deux parties différentes :

1. Le protocole de base contient les concepts fondamentaux pour faire fonctionner une infrastructure Jabber. Il est défini par les RFC 61202, 61213, 61224 (qui remplacent depuis 2011 les 39205 et 39216), 39227 et 39238. Théoriquement, une telle infrastructure ne peut pas fonctionner sans appliquer complètement ces protocoles.
2. Les XMPP Extension Protocols (XEP) sont des propositions d'ajout de fonctionnalités au protocole Jabber. Les serveurs ou clients ne sont pas obligés d'adopter ces extensions. Cela peut bloquer certaines fonctionnalités entre deux utilisateurs.

XMPP est conçu de manière plus large et ouverte que la simple messagerie instantanée populaire et propriétaire. Il est ainsi utilisé par les entreprises et administrations dans le cadre d'échanges de données entre applications (ETL, EAI, ESB) au sein des systèmes d'informations, mais aussi dans le cadre du grid computing, des notifications d'alertes ou d'informations, de la supervision système et réseau, ou le cloud computing. Enfin, XMPP est également utilisé dans le domaine du partage et de la collaboration en quasi-temps-réel comme le tableau blanc (« whiteboard ») ou l'édition et le développement collaboratifs, mais aussi des jeux sur Internet (notamment les jeux de cartes et de plateau).

On ira lire des documents tels que :

1. [Extensible Messaging and Presence Protocol](#)
2. <http://en.wikipedia.org/wiki/XMPP>
3. [Comparison of instant messaging protocols](#)
4. <http://en.wikipedia.org/wiki/SIMPLE>

8. NAT Traversal

Le NAT cache les adresses privées sur le réseau public que les UAs peuvent embarquer dans leurs messages de couche application. Pour connaître l'adresse publique et les ports utilisés, les UAs passent par des techniques comme STUN, TURN, ICE ou encore d'autres techniques de type NAT Traversal.

STUN

Session Traversal Utilities for NAT (STUN), RFC 5389, est un ensemble de méthode standardisées qui permet à un hôte terminal de découvrir son adresse IP publique si il est situé derrière un routeur NAT. Il est utilisé pour permettre du NAT Transversal pour de la voix, de la vidéo, de la messagerie et autres communications interactives en temps réel. Il est censé être utilisé par d'autres protocoles comme Interactive Connectivity Establishment (ICE) ou WebRTC. STUN nécessite l'usage d'un serveur STUN disponible dans l'Internet public.

TURN

STUN ne peut pas traverser tous les types de NAT (Symetric NAT). **Traversal Using Relays around NAT (TURN)** est un autre protocole qui relaie à travers un serveur public le trafic TCP/UDP émanant de réseaux privés. Il permet de traverser les pare-feux et les routeurs NAT. Il autorise seulement des connexions peer-to-peer comme en téléphonie IP.

ICE

Interactive Connectivity Establishment (ICE) est une méthode qui permet de traverser les routeur NAT pour établir des connexions directes entre des réseaux privés. Il permet de choisir entre STUN ou TURN (plus consommateur de ressources).

9. Enregistrement DNS SRV

Source : https://fr.wikipedia.org/wiki/Enregistrement_de_service

Un enregistrement SRV ou enregistrement de service est une catégorie de données du DNS d'Internet qui vise à indiquer quels sont les services disponibles. Ce type d'enregistrement est défini dans la RFC 2782.

Les enregistrements SRV sont souvent utilisés par les clients Microsoft Windows afin de trouver le contrôleur de domaine pour un service donné.

Par ailleurs, les enregistrements SRV sont communément utilisés en association avec les protocoles standards ci-dessous :

- XMPP (Jabber)
- SIP
- LDAP

Format DNS SRV

Un enregistrement SRV contient les informations suivantes :

- **Service**: le nom symbolique (commençant généralement par un symbole de soulignement) du service concerné (par exemple `_sip`).
- **Protocole** : généralement, c'est soit `_tcp` pour TCP, soit `_udp` pour UDP.
- **Nom.de.domaine** : le domaine de validité de l'enregistrement (pleinement qualifié au format FQDN ou local à la zone DNS en cours de définition pour la même autorité d'origine).
- **TTL** : champ standard DNS indiquant la durée de validité (Time-To-Live, durée de vie) de la réponse (en secondes).
- **Classe** : champ standard DNS indiquant la classe d'adressage (c'est toujours IN pour Internet).
- **Type** : l'identifiant du type d'enregistrement DNS (toujours SRV ici pour un enregistrement de service)
- **Priorité** : la priorité du serveur cible (valeur entière non négative, plus elle est faible, plus ce serveur sera utilisé s'il est disponible). S'il y a plusieurs enregistrements à différentes priorités pour le même service et un même

protocole, un seul enregistrement pour chacune des priorités sera retourné aux clients DNS (les clients sont censés alors tenter de se connecter en priorité au serveur retourné ayant la plus faible valeur de priorité parmi les enregistrements retournés, mais si cela échoue, ils peuvent utiliser le serveur suivant de priorité plus élevée dans la liste de serveurs qui lui sont retournés). Différentes priorités permettent de mettre en place des services de secours (éventuellement distincts dans leur contenu et plus limité dans les possibilités, par exemple un ou plusieurs serveurs alternatifs fonctionnant en lecture seule sans support de certaines modifications par le service).

- **Poids** : poids relatif pour les enregistrements de même priorité (valeur entière non nulle, permet au serveur DNS de retourner aléatoirement un des serveurs cibles ayant la même priorité, avec une distribution correspondant au poids indiqué, relativement au poids total des autres enregistrements de même priorité). Le poids indiqué n'a pas d'effet s'il n'y a qu'un serveur cible de la même priorité pour le même service et le même protocole (ce paramètre permet une distribution de charge assez sommaire entre plusieurs serveurs pour les services très fréquentés par de nombreux clients effectuant des requêtes DNS séparées, mais il n'aura pas d'effet si ces clients font leur requêtes DNS via un même serveur DNS proxy cache).
- **Port** : le numéro de port (TCP ou UDP selon le protocole ci-dessus) où le service est disponible.
- **Cible** : le nom du serveur qui fournit le service concerné (doit être résolu en adresse IPv4 ou IPv6 par d'autres requêtes DNS sur les enregistrements A ou AAAA du nom de service cible) avec le protocole et sur le port indiqué.

Par exemple, des enregistrements SRV peuvent ressembler à ceci :

```
_sip._tcp.example.com. 86400 IN SRV 0 33 5060 serveursip1.example.com.
_sip._tcp.example.com. 86400 IN SRV 0 33 5060 serveursip2.example.com.
_sip._tcp.example.com. 86400 IN SRV 0 33 5060 serveursip3.example.com.
_sip._tcp.example.com. 86400 IN SRV 10 100 5060 serveursip-offline.example.com.
```

Ils pointent vers trois serveurs nommés `serveursip{1,2,3}.example.com` qui attendent des connexions SIP sur le port TCP numéro 5060. Ici, la priorité indiquée est de 0 (priorité standard par défaut), et le poids de ces serveurs est de 33 (le serveur DNS retournera équitablement aléatoirement un des 3 serveurs aux clients DNS, mais de façon équitable). La durée de validité indiquée (86400 secondes soit une journée entière) permet aux clients DNS de garder dans leur cache local cette information pendant une journée entière avant de devoir réinterroger le serveur DNS.

Le serveur retournera aussi aux client un second serveur pour leurs requêtes, pointant sur `serveursip-offline.example.com` qui fournira un service SIP (éventuellement limité) si les clients ne parviennent pas à se connecter à l'un des trois premiers serveurs (ce serveur de secours à une priorité indiquée à 10 au lieu de 0, il n'est pas destiné à une utilisation permanente par les clients mais peut servir le temps d'une opération de maintenance sur l'un des 3 serveurs SIP de base, son poids indiqué à 100 n'a pas d'effet si c'est le seul serveur de secours).

Vérification de SRV DNS

Pour vérifier la présence d'un enregistrement SRV sur un domaine, les commandes `nslookup` et `dig` peuvent être utilisées.

Par exemple :

```
dig SRV _sip._tcp.example.com.
```

Sur quel port écoute le service SIP/UDP du domaine `anveo.com` ?

```
$ dig @8.8.8.8 SRV _sip._udp.anveo.com.

; <<>> DiG 9.8.3-P1 <<>> @8.8.8.8 SRV _sip._udp.anveo.com.
; (1 server found)
;; global options: +cmd
```

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37038
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;_sip._udp.anveo.com.      IN      SRV

;; ANSWER SECTION:
_sip._udp.anveo.com.      3599    IN      SRV      10 100 5010 sip.anveo.com.
_sip._udp.anveo.com.      3599    IN      SRV      20 100 5010 sip.ca.anveo.com.
_sip._udp.anveo.com.      3599    IN      SRV      30 100 5010 sip.de.anveo.com.

;; Query time: 39 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Sun Nov 13 21:43:34 2016
;; MSG SIZE rcvd: 142
```

Y a-t-il un serveur SIPS (TCP 5061) sur le domaine cisco.com ?

```
$ dig @8.8.8.8 SRV _sips._tcp.cisco.com.

; <<>> DiG 9.8.3-P1 <<>> @8.8.8.8 SRV _sips._tcp.cisco.com.
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 51584
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;_sips._tcp.cisco.com.      IN      SRV

;; ANSWER SECTION:
_sips._tcp.cisco.com.      3599    IN      SRV      10 10 5061 vcsgw.cisco.com.

;; Query time: 142 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Sun Nov 13 21:48:07 2016
;; MSG SIZE rcvd: 73
```

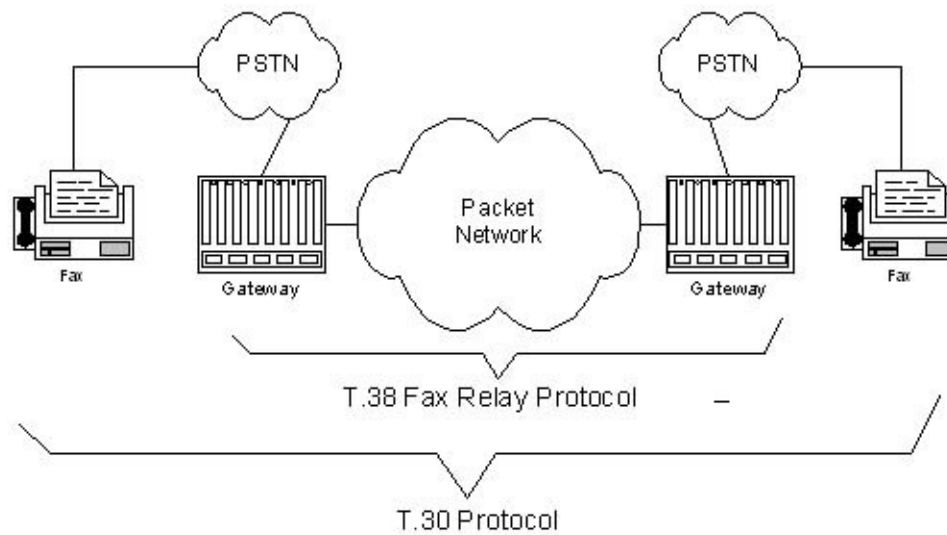
ou encore :

```
nslookup -q=SRV _sip._tcp.example.com.
```

10. Fax T.38

Source : <https://fr.wikipedia.org/wiki/FoIP> et <https://en.wikipedia.org/wiki/T.38>

Le fax sur réseau IP, ou FoIP pour « Fax over IP », est une technique qui permet d'émettre des télécopies via Internet ou tout autre réseau acceptant le protocole TCP/IP. Cette technologie est utilisée pour supporter le service de fax IP. On utilise pour cela les mêmes protocoles et les mêmes architectures que ceux mis en œuvre pour la voix sur IP.



L'une des problématiques du transfert de fax sur de la VoIP est que les codecs habituellement employés pour la voix dégradent de manière significative le signal et empêchent alors les fax de transiter correctement, la bande passante nécessaire à un fax étant supérieure à celle de la voix. Les protocoles utilisés dans la transmission des fax peuvent être le protocole T.38 pour le temps réel ou T.37 pour le mode Store and Forward. Les passerelles de conversion utilisent un codage propre au fax : le codec G.711 u-law.

Marchés

- 1. Marchés VoIP / UC
 - 1.1. Marchés VoIP / UC
 - 1.2. Segment Cibles
 - 1.3. Marché Unified Communications
 - Collaboration Solutions. Unified Communications and Contact Center
 - Video and Web Conferencing
 - 1.4. Etudes Gartner
 - 1.5. Acteurs mondiaux en Communications Unifiées
 - 1.6. Autres marchés
- 2. Aperçu des logiciels Open Source
 - 2.1. Solutions serveurs Open Source
 - 2.2. IP PABX
 - Asterisk
 - Freeswitch
 - 2.3. Proxys SIP
 - Kamailio
 - OpenSips
 - Repro
 - Flexisip
 - 2.4. Clients VoIP Open Source
 - Clients graphiques
 - Librairies

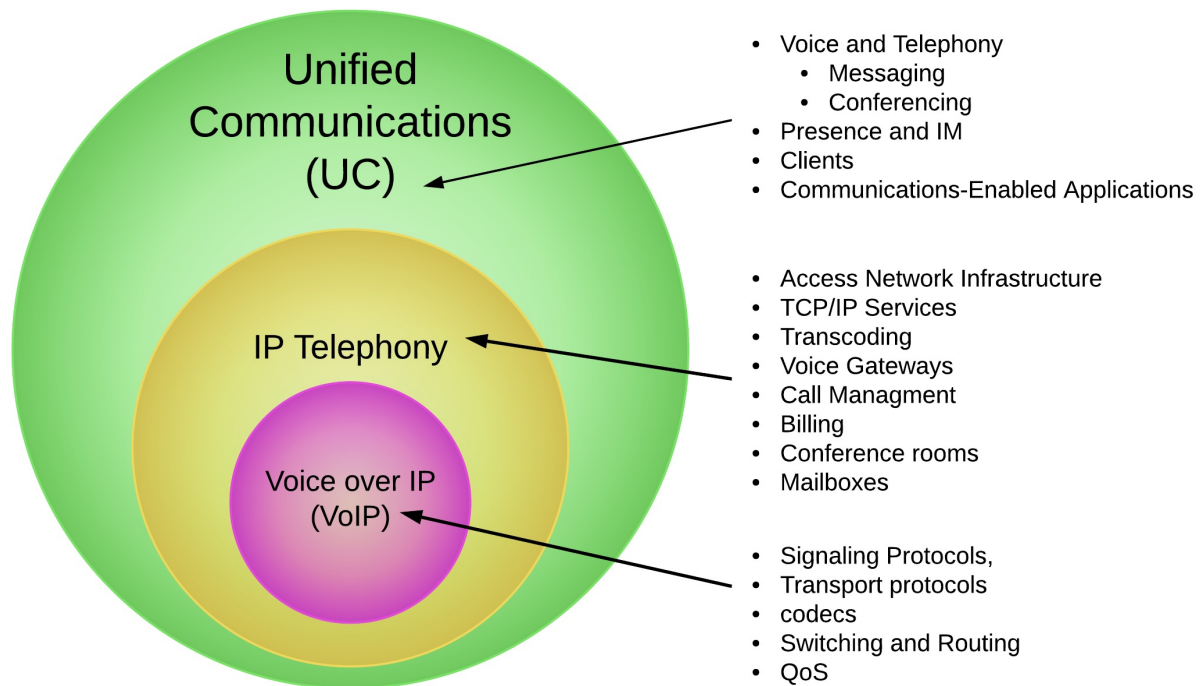
1. Marchés VoIP / UC

Les premières illustrations qui viennent à l'esprit sont les déploiement de centraux téléphoniques d'entreprises de type IP-PBX ou du Trunking SIP en remplacement des lignes ISDN qui mènent vers le PSTN. Mais vu plus largement, le protocole se propose aussi de fédérer les services de téléphonie à l'échelle globale avec les technologies de l'Internet.

1.1. Marchés VoIP / UC

Les Communications Unifiées (Unified communications, UC) est un mot de marketing désignant des services d'entreprise de communication en temps réel comme la messagerie instantanée (IM, chat), les informations de présence, la voix (la téléphonie IP/VoIP incluses), des fonctionnalités de mobilité (suivi des extensions, un seul numéro d'appel), Conférence Web audio et vidéo, convergence fixe-mobile, partage de bureau, partage de données (fichiers, tableaux blancs), contrôle d'appel et reconnaissance vocale avec des services de communications comme de la Messagerie Unifiée (messages vocaux, e-mail, SMS et FAX).

Les Communications Unifiées ne correspondent pas à un seul produit. Le concept correspond à un ensemble de produits qui fournit une interface et une expérience consistante et unifiée quel que soient les multiples interfaces ou modes de communication utilisés.



1.2. Segment Cibles

- Entreprises
- PME
- Cloud

1.3. Marché Unified Communications

Source : <http://www.cisco.com/c/en/us/solutions/collaboration/analysts.html>

Collaboration Solutions. Unified Communications and Contact Center

- Aragon Research Globe for Unified Communications and Collaboration (March 2017) (PDF - 4.2 MB)
- Gartner: Critical Capabilities for Unified Communications (July 2017)
- Gartner: Magic Quadrant for Contact Center Infrastructure, Worldwide (May 2017)
- Gartner: Magic Quadrant for Unified Communications (July 2017)
- IDC MarketScape: Worldwide Unified Communications and Collaboration 2017 Vendor Assessment (June 2017)
- Wainhouse Research: 2015 UC ITDM Survey - UC Deployments - EnterpriseDeployments (December 2015) (PDF - 344 KB)

Video and Web Conferencing

- Gartner: Critical Capabilities for Meeting Solutions (October 2017)
- Gartner: Magic Quadrant for Meeting Solutions (September 2017)
- IDC MarketScape: Worldwide Enterprise Videoconferencing Equipment 2016 Vendor Assessment
- ZK Research White Paper: A Guide to Choosing the Best Online Meeting Provider (September 2017)
- Wainhouse Research: Artificial Intelligence in the Conferencing World (November 2017) (PDF - 971 KB)

1.4. Etudes Gartner

Les études Magic Quadrant de Gartner sont des études qualitatives globales sur les acteurs commerciaux d'un marché défini. Ce ne sont pas des études quantitatives, objectivées scientifiquement ou sources d'informations absolues.

Elles sont souvent offertes par les vainqueurs de ces études. Elles ne sont pas libres de droits.

Elles sont d'excellents outils pour se faire une idée de ce que les entreprises attendent d'un marché et pour prendre connaissance des offres actuellement disponibles. Elles permettent de créer un référentiel de critères pour établir un appel d'offre pour une solution de téléphonie UC. Elles permettent aussi d'établir des critères comparatifs entre différents modèles de déploiement ou commerciaux/open source.

Ces études positionnent les constructeurs et leur offre du moment selon deux axes :

- La capacité du constructeur à répondre aux attentes du marché (ability to execute)
- La complétude de la vision (completeness of vision) est le niveau de stratégie perçu pour atteindre les attentes du marché.

Elles positionnent les acteurs d'un marché dans un des quatre rôles :

- Leaders
- Challengers
- Visionaries
- Niche Players

On se renseignera dans les études Gartner sur :

- La définition du marché
- La portée et le moment de l'étude
- Les attentes du marché
- Le positionnement des acteurs
- Leur actualité, forces et faiblesses

1.5. Acteurs mondiaux en Communications Unifiées

Pour Entreprise, PME et cloud

- Cisco
- Mitel
- Avaya
- Microsoft
- Unify
- Google
- Alcatel-Lucent

1.6. Autres marchés

De manière plus globale celui de la "Collaboration" mais aussi :

- Domotique
- Surveillance
- WebRTC
- Conferencing

2. Aperçu des logiciels Open Source

En concurrence directe avec des solutions intégrées Cisco, Alcatel ou encore Avaya.

- https://en.wikipedia.org/wiki/List_of_SIP_software
- https://en.wikipedia.org/wiki/Comparison_of_VoIP_software

2.1. Solutions serveurs Open Source

- IP PABX
 - Asterisk et dérivatifs
 - Freeswitch
- Proxys SIP (Fed VoIP)
 - Kamailio
 - OpenSips
 - Repro
 - Flexisip

2.2. IP PABX

Ces logiciels font office de B2BUA, de serveurs Multimedia, de transcodeur de medias, d'IVR, etc.

Asterisk

Asterisk Core (<http://www.asterisk.org/>), sous Linux, en installation par paquet ou en installation native par compilation sur des plateformes Intel ou embarquées, voire en VM ou en Container ou encore en VPS.

Il existe aussi des dérivatifs à partir d'Asterisk et de FreePBX (GUI Web) fournis en distribution Linux :

- FreePBX (<https://www.freepbx.org/>) supporté par Schmooze
- Elastix (<http://www.elastix.org/>)
- Trixbox (<http://www.fonality.com/trixbox>)
- AsteriskNow comme demo d'Asterisk (<http://www.asterisk.org/downloads/asterisknow>)

Notons aussi l'endurance du projet **Xivo** (<http://www.xivo.io/>), basé Asterisk avec une interface spécifique.

On trouvera aussi des solutions sur l'embarqué Raspberry PI (voir https://iot.goffinet.org/labs_raspipbx.html et <http://www.raspberry-asterisk.org/>).

C'est sans compter les différentes appliances maintenues ou non que l'on peut trouver sur le marché.

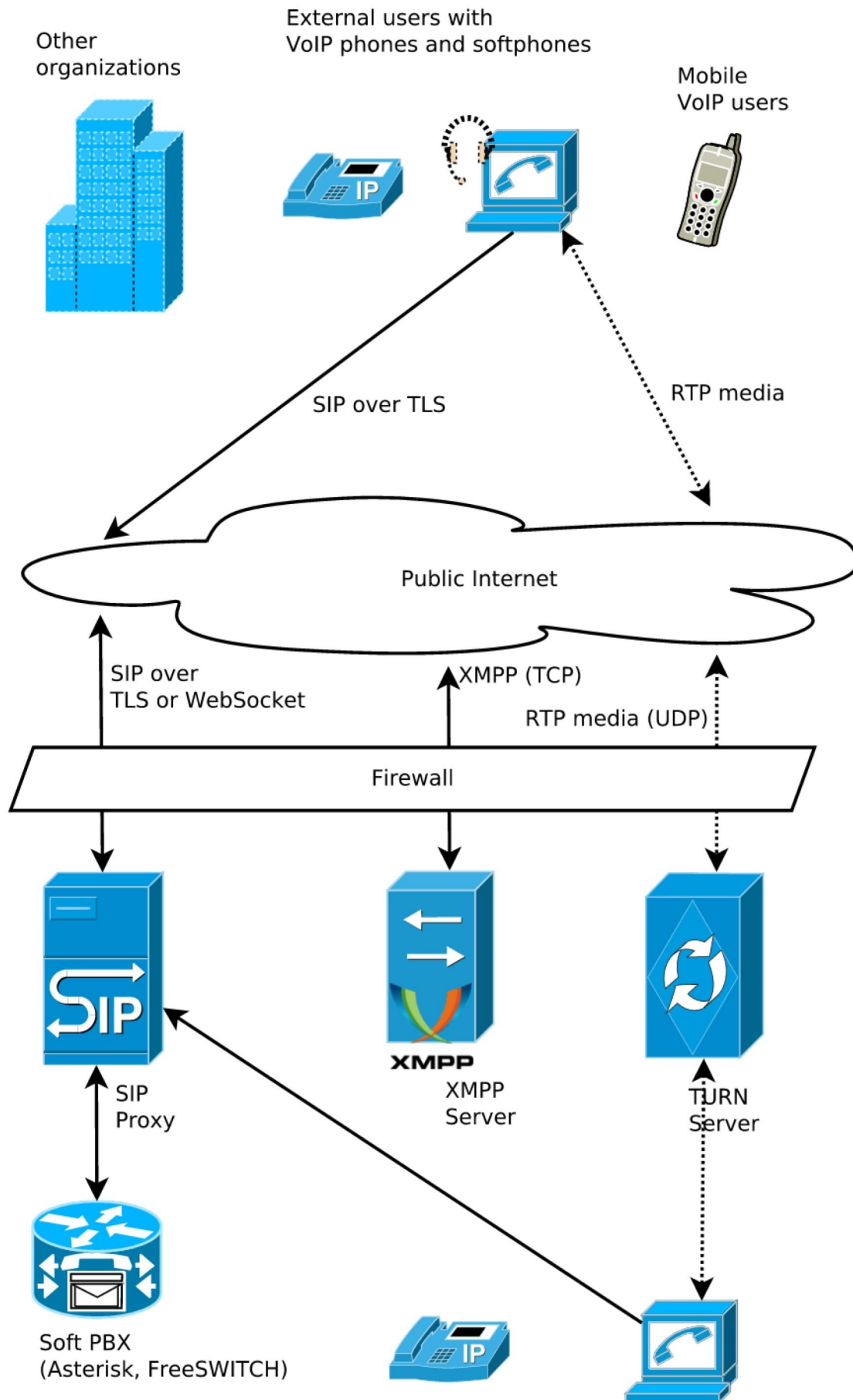
Freeswitch

<https://en.wikipedia.org/wiki/FreeSWITCH>

2.3. Proxys SIP

Ces logiciels vise à exploiter la Federated VoIP : <http://rtcquickstart.org/>.

La "Federated VoIP" est une forme de la téléphonie qui utilise la VoIP entre des domaines autonomes dans l'Internet Public sans aucun déploiement d'un point central d'échange ou de commutateurs pour le routage du trafic. (https://en.wikipedia.org/wiki/Federated_VoIP)



Source : <http://rtcquickstart.org/guide/multi/rtc-architecture-big-picture.html#arch-dia-1>

La "Federated VoIP" peut utiliser notamment ENUM comme système d'adressage, de localisation et d'identification des participants. Elle exploite des entrées DNS SRV à la manière des entrées MX pour le service SMTP. Elle implémente des communications sécurisées grâce à TLS et aux certificats X.509. Elle utilise les protocoles SIP et XMPP/Jabber pour établir les communications.

Kamailio

Kamailio est un serveur SIP libre (anciennement OpenSER)

<https://fr.wikipedia.org/wiki/Kamailio>

<https://www.kamailio.org/w/>

OpenSips

Fork de OpenSER

<http://www.opensips.org/>

Repro

https://www.resiprocate.org/About_Repro

Flexisip

<http://www.linphone.org/technical-corner/flexisip/overview>

2.4. Clients VoIP Open Source

Clients graphiques

Logiciel	Support OS	Licence	remarque
CSipSimple	Android	GNU GPL v3	concurrent avec le téléphone natif
Jitsi	GNU/Linux, Windows, Mac OS X (Java)	Apache v2	:)
Linphone	GNU/Linux, Windows, Mac OS X	GNU GPL 2	ouvert
Zoiper	GNU/Linux, Windows, Mac OS X, Android, Apple Ios	Commercial/gratuit	Robuste
Blink	GNU/Linux, Windows, Mac OS X	Commercial/gratuit	SIPS, ZRTP

Librairies

- PJSIP : <http://www.pjsip.org/docs/book-latest/html/index.html>, <http://www.pjsip.org/pjsua.htm>
- Sipp : <http://sipp.sourceforge.net/>
- Linphonec : <http://www.linphone.org/technical-corner/linphone/documentation>

Connexion SIP

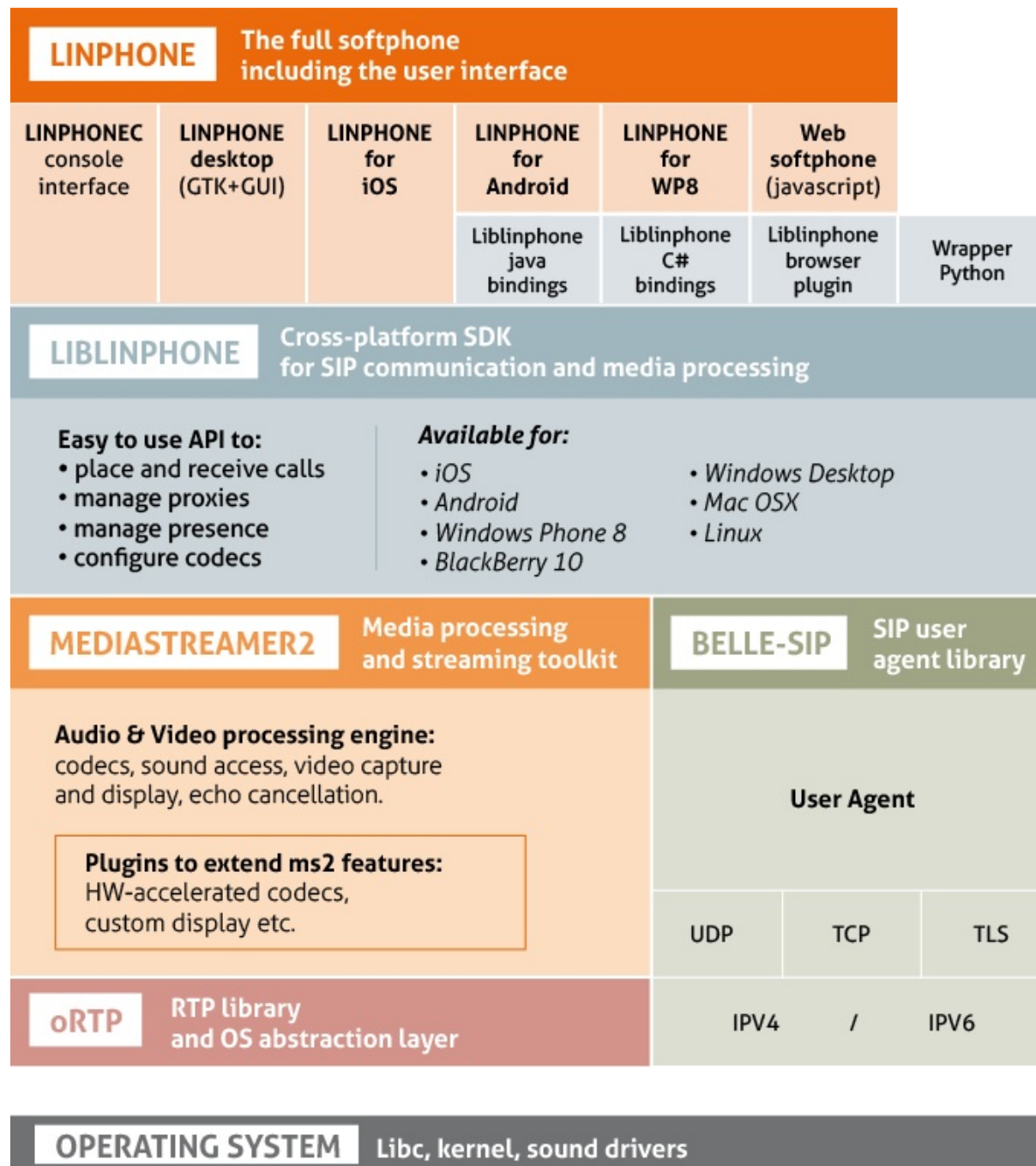
- [Description](#)
- [1. Logiciel Linphone](#)
- [2. Topologie](#)
- [3. Exercice de connexion SIP directe UAC/UAS](#)

Description

Exercice de connexion SIP directe UAC à UAS avec Linphone

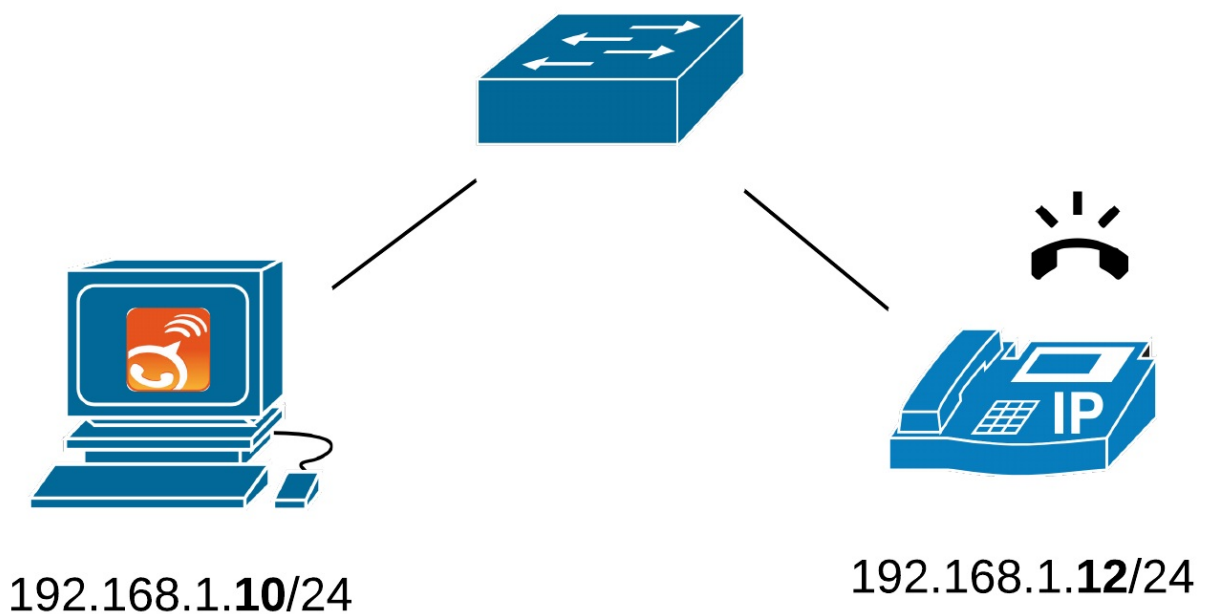
1. Logiciel Linphone

- Pour les besoins du lab, on utilisera la version desktop
- On s'intéressera aussi aux outils de bas niveau comme `linphonec` et `linphonecsh`



Source : <http://www.linphone.org/technical-corner/linphone/overview>

2. Topologie



3. Exercice de connexion SIP directe UAC/UAS

A deux participants disposant de leur machine ou entre une machine physique et un machine virtuelle.

- Télécharger **Linphone pour Desktop** : <http://www.linphone.org/downloads-for-desktop.html>
- Noter l'adresse IP de l'interface connectée au LAN
- Adapter le pare-feu et lancer Linphone avec les droits requis
- Noter l'adresse IP locale et le port d'écoute de Linphone
- Activer/désactiver le support vidéo
- Etablir un appel vers un voisin avec l'URI : `sip:<adresse ip du correspondant>`
- Activer uniquement les codecs PCMU/PCMA
- Lancer une capture Wireshark avec le filtre "sip or rtp"
- Etablir un nouvel appel vers le partenaire

Infrastructure VoIP

- 1. Caractéristiques d'une infrastructure LAN
- 2. Infrastructure WAN
 - Connexion à Internet
 - Réseau WAN privé
- 3. SIP Trunk Provider
- 4. PBX / sofswitch
- 5. Téléphones IP
- 6. Interfaces Trunk analogiques/digitaux
- 7. Fournisseurs SIP

Il existe une offre très large en matière de matériel, de logiciels et de services VoIP. Pour le matériel (hors routeurs, pare-feux et commutateurs), on peut commencer par consulter les sites de ventes en ligne comme :

- <http://www.voipon.co.uk/>
- <http://www.voipsupply.com/>
- <https://www.digium.com/>

1. Caractéristiques d'une infrastructure LAN

L'infrastructure LAN devrait être :

- robuste
- évolutive
- sécurisée
- gérée

L'infrastructure LAN doit répondre à plusieurs critères.

- Elle est documentée et basée sur un modèle de conception.
- Elle est robuste avec une redondance L1, L2 et L3.
- Elle est évolutive avec une possibilité de déployer une architecture VLAN.
- Elle est sécurisée, documentée et dispose de mécanismes d'authentification forte

Voir CCNA Voice Primer 5. Preparing the network to support voice.

2. Infrastructure WAN

Connexion à Internet

- Liaisons Synchrone/Asynchrone
- Débit :
 - Débit garanti
 - SLA
 - Quantité
- Connexions VPN

Réseau WAN privé

- Coeur IP/MPLS
- Accès CPE : FO, SDSL, ADSL, VDSL, ...
- garantie, contrats SLA/QoS

3. SIP Trunk Provider

- Connexion dédiées (SLA/QoS)
- Connexion VPN
- Fournisseur SIP
- Fournisseur IAX

4. PBX / sofswitch

Définition du PBX, PBX/IP +++

Fonctionnalités minimales.

- VoIP Ready
- Voice Messaging
- Mobility (find me/follow me, convergence mobile/fixe)
- Conferencing (diminution des coûts de voyage)
- Reporting

Pour le dimensionnement prendre garde à plusieurs éléments.

- PC, CPU/RAM, machine virtuelle, Image Cloud, SaaS, dimensionnement
- Nombre de téléphones
- Nombre d'appels concurrents
- Charge suscitée par l'encodage des medias
- Solutions déportées (Cloud) gérées ou non.

On trouvera des appliances chez :

- FreePBX : <https://www.freepbx.org/store/freepbx-appliances/>
- Elastix : <http://www.elastix.com/en/elastix-appliances/>
- Asterisk : <http://www.asterisk.org/products>
- VoiPSupply : <http://www.voipsupply.com/ip-pbx-hardware/appliances>
- Varia : Askozia, PC-Engine : http://varia-store.com/Ready-Systems/Telephone-System:::223_748.html
- ...

5. Téléphones IP

- Segment physique dédié ou VLAN dédié
- Problématique de l'alimentation
- Software
- Hardware
- Matériel divers
- interfaces USB etc ATA
- Solution de gestion et de maintenance

6. Interfaces Trunk analogiques/digitaux

Banks USB, cartes PCI, voice gateway, voir catalogues/prix/specs :

- Connexions analogiques FXO/FXS
- Connexions digitales BRI/PRA
- Ascenseurs (bank FXS)

7. Fournisseurs SIP

Offres de fournisseurs SIP :

- Nomado : https://www.nomado.eu/fr_BE/
- 3stars : <http://www.3starsnet.com/fr/prive/accueil.html>
- Anveo : <http://www.anveo.com/>
- Voipbuster : <http://www.voipbuster.com/dashboard>
- Offre OVH : <https://www.ovhtelecom.fr/telephonie/>
- Opérateurs traditionnels : Proximus, Orange Business Services, Base, Mobistar, Voo, Easynet, IP Nexia, etc.

Migration VoIP

- 1. Quand migrer vers la VoIP ?
- 2. Architecture future
- 3. Topologies de déploiement
 - VoIP / ToIP dans une TPE
 - Architectures PME monosite
 - Architectures multisites
 - Architectures critiques

1. Quand migrer vers la VoIP ?

- Quand le central téléphonique de l'entreprise est amorti.
- Quand l'entreprise connaît une croissance en utilisateurs ou en périphérique.
- Quand l'entreprise décide d'investir dans son système d'information.
- Certains marchés verticaux sont plus réceptifs que d'autres.

2. Architecture future

Décider d'implémenter de la VoIP au sein d'une entreprise nécessite de poser les bonnes questions :

- Quelle est l'architecture existante du système d'information de l'entreprise ?
- Quel état de l'infrastructure actuelle du réseau ?
- Téléphonie fixe : contrats, tarifs et technologies.
- Téléphonie mobile : contrats, tarifs et technologies.
- Coûts de maintenance du PBX.
- Coûts et fonctionnalités des périphériques.
- Quels sont les services qui seront disponibles pour les utilisateurs finaux ?
- Quelle sont les fonctionnalités à déployer ?
- Selon quelles phases ?
- Quels sont les projets à intégrer au déploiement de la VoIP ?

Au minimum :

- une définition des besoins en fonctionnalités "Communications unifiées" et un cahier des charges
- un audit de l'infrastructure du réseau ;
- une étude de faisabilité du QoS ;

Des projets d'intégration plus complexes :

- de la communication inter-site, multi-opérateur
- du télétravail
- l'intégration de périphériques mobiles
- un helpdesk
- un système de paiement
- un CRM/ERP, avec suivi de clientèle, un centre d'appel

Encore des questions à se poser :

- Quel est le plan de déploiement à mettre en oeuvre ?
- Des séances d'information et de formation sont-elles prévues pour le management (enjeux) et le personnel (bon

usage) ?

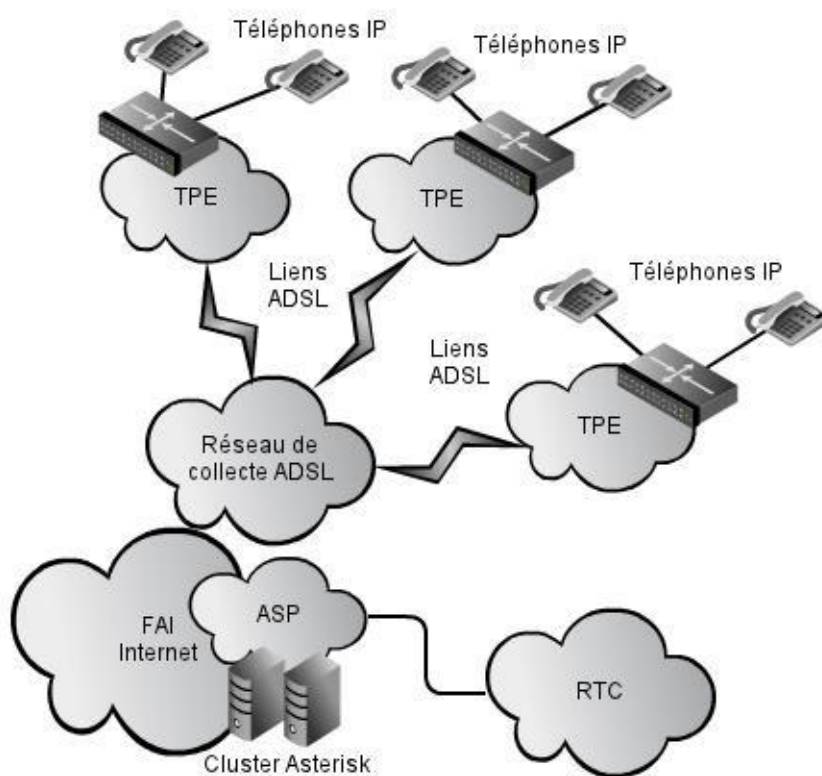
- Quel est le périmètre budgétaire ?
- Quel est le retour sur investissement (ROI pour Return On Investment) escompté ? Est-il qualifiable et quantifiable ?

3. Topologies de déploiement

On peut imaginer une multitude de scénarios de déploiement. En voici quelques exemples.

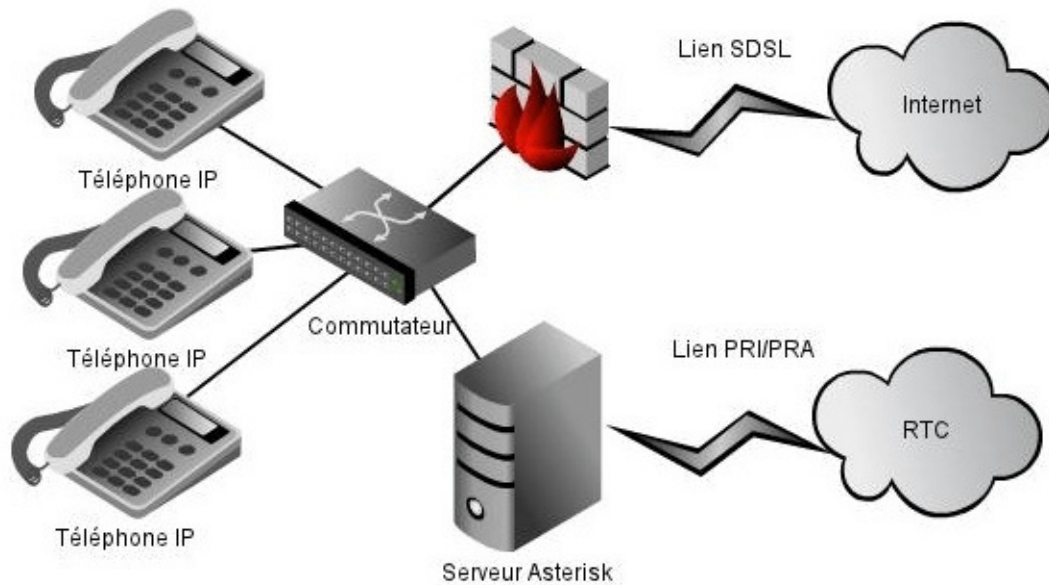
VoIP / ToIP dans une TPE

Le service est assuré par un ASP comme Anveo ou Nomado par exemple.



Architectures PME monosite

Ici avec un PBX local connecté directement au RTC.

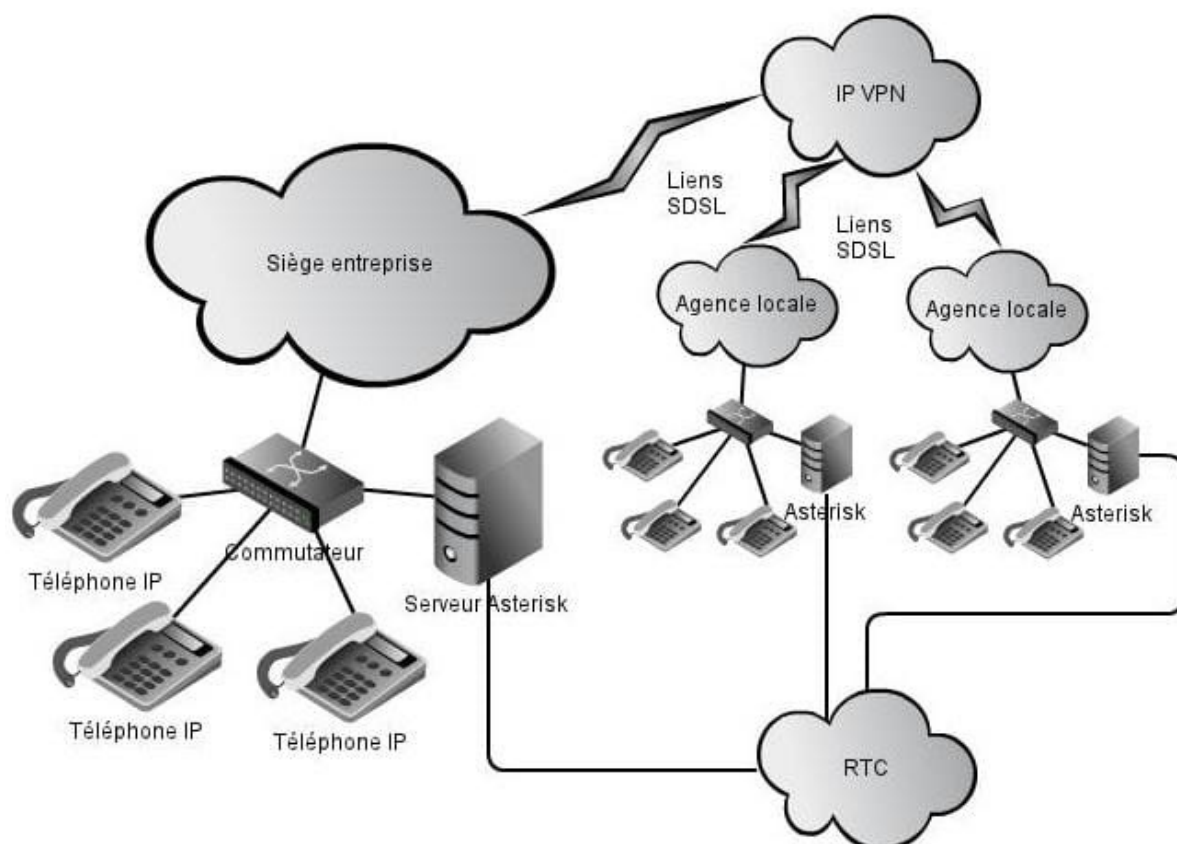


Ici le PBX connecté au RTC est déporté chez un fournisseur de service.

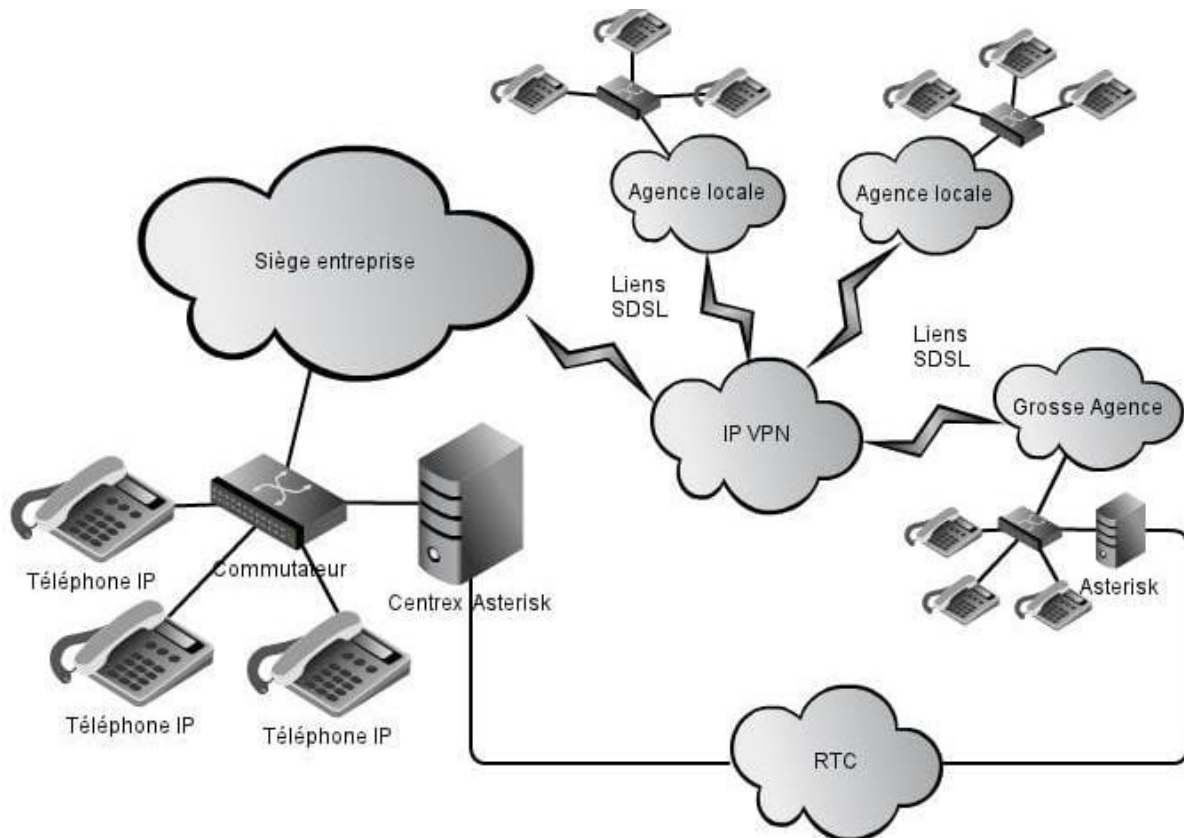


Architectures multisites

Chaque site dispose de son propre PBX connectés entre eux en IP et connectés individuellement au RTC.

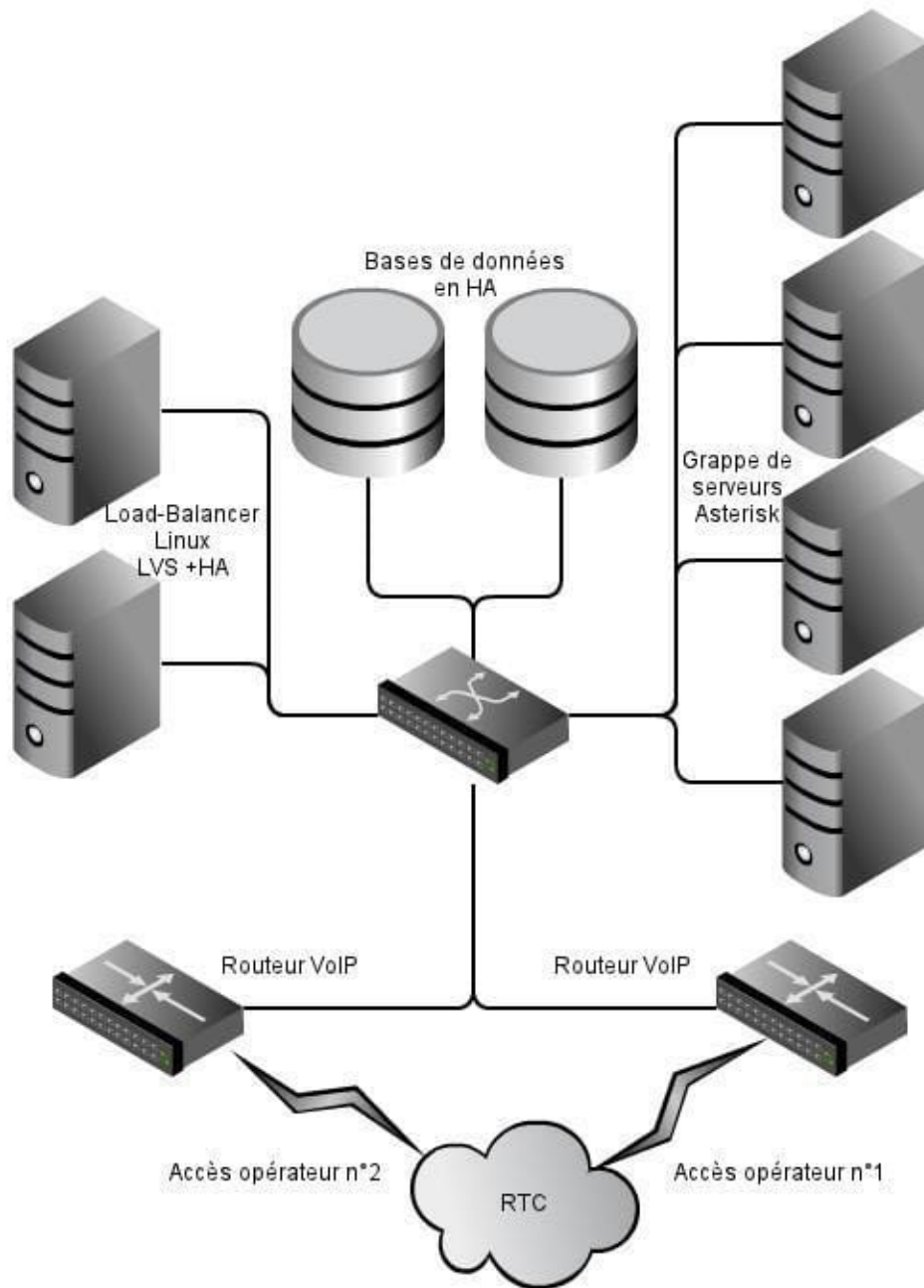


Dans cette topologie, tous les sites distants sont interconnectés en IP mais certains seulement disposent d'un PBX connecté au RTC. Un site secondaire peut prendre le relais vers le RTC en cas de rupture sur le site principal.



Architectures critiques

La charge est distribuée entre plusieurs machines, l'architecture du PBX est éclatée. Des services de haute disponibilité sont mis en œuvre. Des passerelles redondantes sont mises en place.

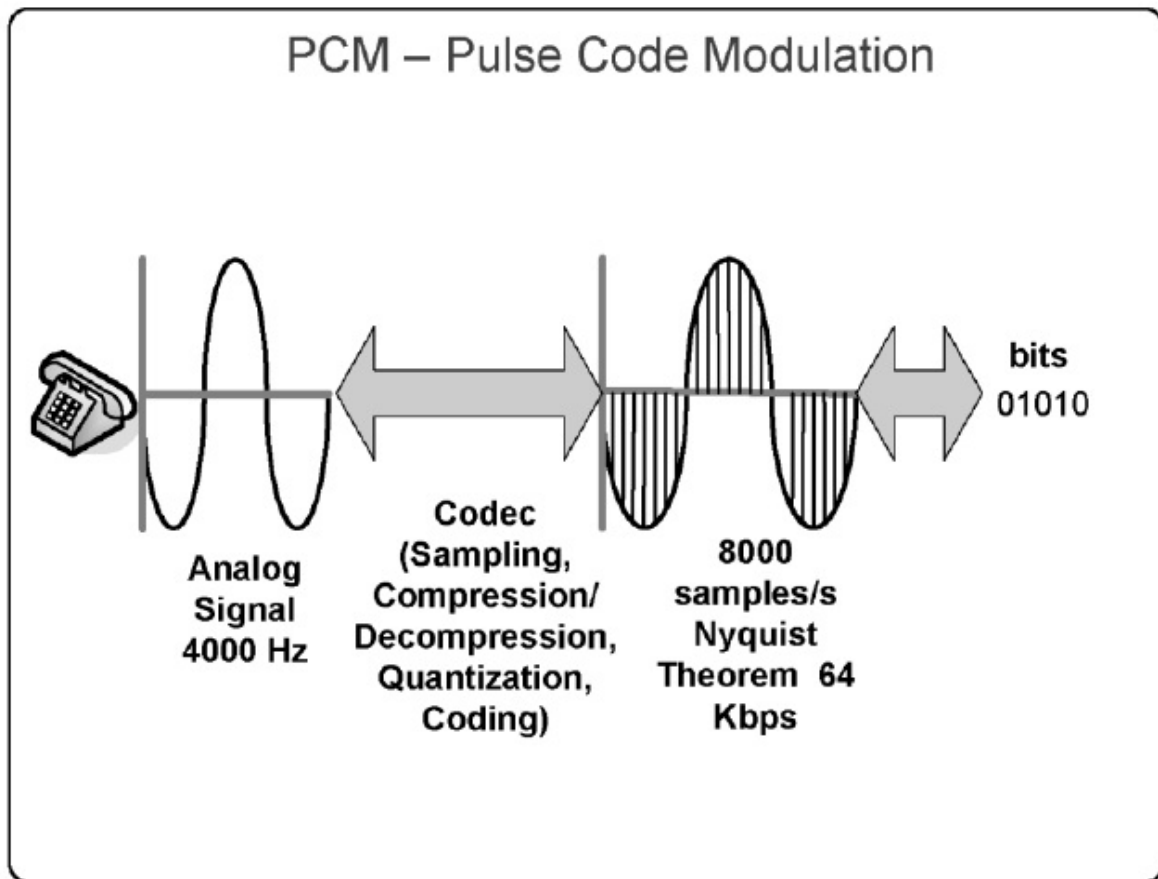


Conception VoIP

- 1. Codecs
 - Définition
 - Exemples de codecs/MOS
 - Comment choisir son codec ?
 - Surcharge d'en-têtes
 - Calcul de bande passante et dimensionnement de lignes
- 2. Réduction de la bande passante nécessaire
- 3. Intégrité et confidentialité des échanges

1. Codecs

Définition



Source : Configuration Guide for Asterisk PBX (Flavio E. Goncalves)

Un codec est un dispositif capable de compresser et/ou de décompresser un signal numérique. Ce dispositif peut être un circuit électronique, un circuit intégré ou un logiciel. Le mot-valise « codec » vient de « codage-décodage » (-COde-DECode en anglais). D'un côté, les codecs encodent des flux ou des signaux pour la transmission, le stockage ou le chiffrement de données. D'un autre côté, ils décodent ces flux ou signaux pour édition ou restitution. Les différents algorithmes de compression et de décompression peuvent correspondre à différents besoins en qualité de restitution, de temps de compression ou de décompression, de limitation en termes de ressource processeur ou

mémoire, de débit du flux après compression ou de taille du fichier résultant. Ils sont utilisés pour des applications comme la téléphonie, les visioconférences, la diffusion de médias sur Internet, le stockage sur CD, DVD, la télé numérique par exemple.

Exemples de codecs/MOS

Le MOS pour *Mean Opinion Score* évalue la qualité audible de la voix sur une échelle de 0 à 5.

Exemples de codecs devant être disponible sur les UA :

- GSM: 13 Kbps, MOS 3.8
- iLBC: 13.3 Kbps, MOS 4.14
- ITU G.711: 64 Kbps, MOS 4.1
- ITU G.723.1: 5.3/6.3 Kbps MOS 3.8
- ITU G.726: 16/24/32/40 Kbps MOS 3.85
- ITU G.729: 8 Kbps MOS 3.92
- ITU G.722: 64 Kbps - High definition MOS 4.13

Comment choisir son codec ?

Codec	g.711	g.729A (20 ms)	iLBC (30 ms)	GSM 06.10 RTE/LTP
Bandwidth (Kbps)	64	8	13.33	13
Costs	Free	~ USD10.00 (per channel)	Free	Free
Resistance to Frame Erasure ¹	No mechanism	3%	5%	3%
Complexity MIPS ²	~0.35	~13	~18	~5

Source : Configuration Guide for Asterisk PBX (Flavio E. Goncalves)

- Qualité du son (MOS – Mean Opinion Score)
- Coût de licence
- Consommation CPU
- Consommation bande passante
- Gestion de la perte de paquets
- Support des périphériques (PBX/Téléphones)

<http://www.voip-info.org/wiki/view/Codecs>

Surcharge d'en-têtes

Un appel G.729 avec un voice payload size of 20 octets (20 ms) :

- 18 octets Ethernet header and trailer +
- 40 octets IP/UDP/RTP headers +

- 20 octets voice payload

= 78 octets

Total packet size (bits) = (78 octets) * 8 bits per octet = 624 bits

PPS = (8 Kbps codec bit rate) / (160 bits) = 50 pps

Note: **160 bits** = 20 octets (default voice payload) * 8 bits per byte

Bande passante par appel = voice packet size (624 bits) * 50 pps = 31.2 Kbps

Exercice avec un appel G.711 avec une charge de 160 octets (20 ms).

Calcul de bande passante et dimensionnement de lignes

Calculateurs de bande passante

- http://www.cisco.com/en/US/tech/tk652/tk698/technologies_tech_note09186a0080094ae2.shtml

2. Réduction de la bande passante nécessaire

Il y a trois méthodes pour réduire la bande passante nécessaire des appels VoIP

- Compression d'en-têtes RTP (cRTP) : à placer sur des liens WAN sur les routeurs voir la documentation Cisco (http://www.cisco.com/en/US/tech/tk543/tk762/technologies_tech_note09186a0080108e2c.shtml).
- Trunk IAX : protocole ouvert ([RFC 5456](#)) de signalisation et de transport fournissant des options de compression d'en-tête et de confidentialité.
- VoIP payload : sur les passerelles et PBX on peut augmenter taille par défaut (par codec) de la charge RTP. En faisant cela, on réduit la surcharge en en-tête, on diminue la bande passante utilisé mais on augmente les délais.

3. Intégrité et confidentialité des échanges

Protocoles sécurisés

- 802.1q
- SIPS
- SRTP
- ZRTP
- IAX

Aperçu des logiciels Open Source

- 1. Solutions serveurs Open Source
- 2. IP PABX
 - Asterisk
 - Freeswitch
- 3. Proxys SIP
 - Kamailio
 - OpenSips
 - Repro
 - Flexisip
- 4. Clients VoIP Open Source
 - Clients graphiques
 - Librairies

En concurrence directe avec des solutions intégrées Cisco, Alcatel ou encore Avaya.

- https://en.wikipedia.org/wiki/List_of_SIP_software
- https://en.wikipedia.org/wiki/Comparison_of_VoIP_software

1. Solutions serveurs Open Source

- IP PABX
 - Asterisk et dérivatifs
 - Freeswitch
- Proxys SIP (Fed VoIP)
 - Kamailio
 - OpenSips
 - Repro
 - Flexisip

2. IP PABX

Ces logiciels font office de B2BUA, de serveurs Multimedia, de transcodeur de medias, d'IVR, etc.

Asterisk

Asterisk Core (<http://www.asterisk.org/>), sous Linux, en installation par paquet ou en installation native par compilation sur des plateformes Intel ou embarquées, voire en VM ou en Container ou encore en VPS.

Il existe aussi des dérivatifs à partir d'Asterisk et de FreePBX (GUI Web) fournis en distribution Linux :

- FreePBX (<https://www.freepbx.org/>) supporté par Schmooze
- Elastix (<http://www.elastix.org/>)
- Trixbox (<http://www.fonality.com/trixbox>)
- AsteriskNow comme demo d'Asterisk (<http://www.asterisk.org/downloads/asterisknow>)

Notons aussi l'endurance du projet **Xivo** (<http://www.xivo.io/>), basé Asterisk avec une interface spécifique.

On trouvera aussi des solutions sur l'embarqué Raspberry PI (voir https://iot.goffinet.org/labs_raspipbx.html et <http://www.raspberry-asterisk.org/>).

C'est sans compter les différentes appliances maintenues ou non que l'on peut trouver sur le marché.

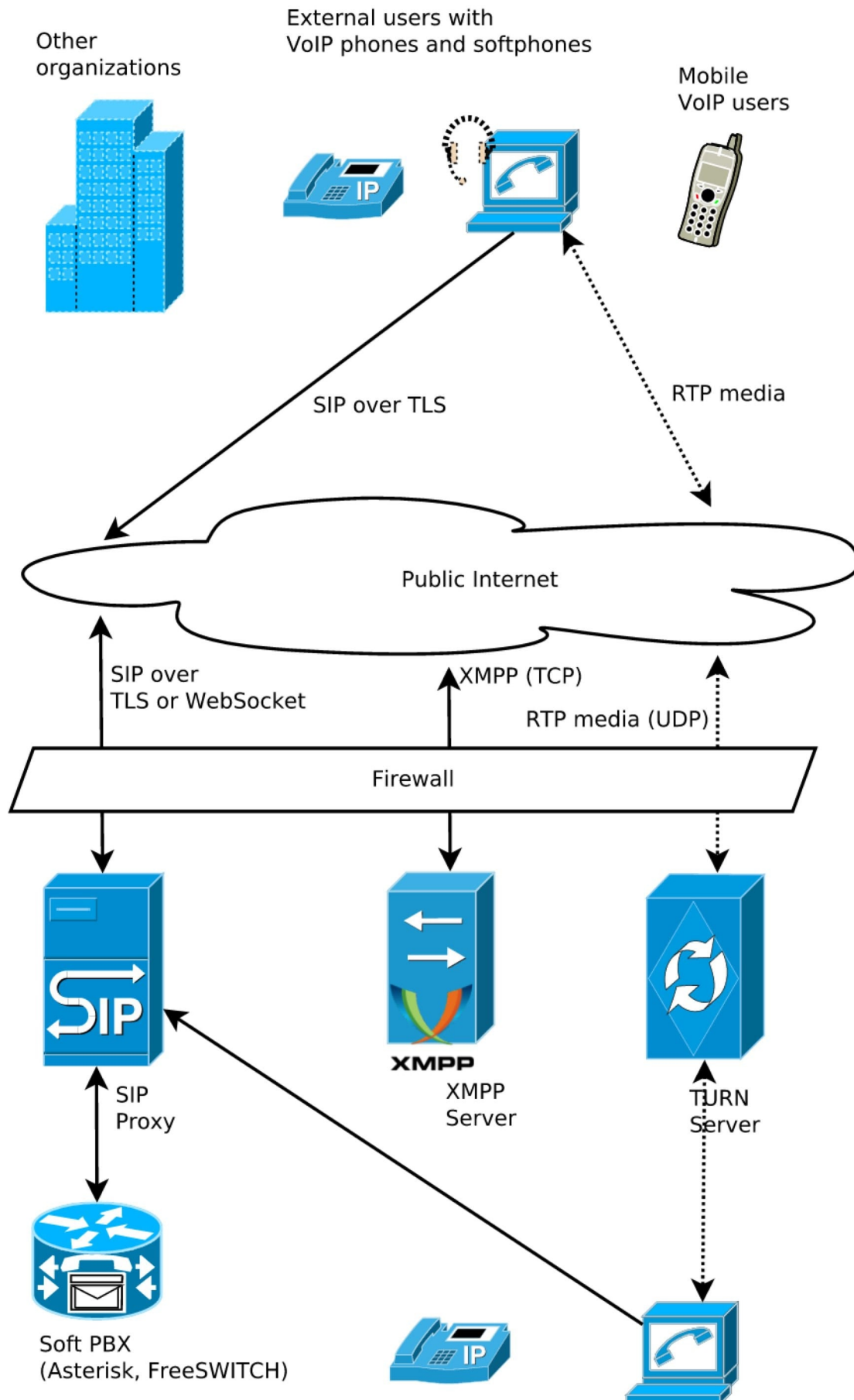
Freeswitch

<https://en.wikipedia.org/wiki/FreeSWITCH>

3. Proxys SIP

Ces logiciels vise à exploiter la Federated VoIP : <http://rtcquickstart.org/>.

La "Federated VoIP" est une forme de la téléphonie qui utilise la VoIP entre des domaines autonomes dans l'Internet Public sans aucun déploiement d'un point central d'échange ou de commutateurs pour le routage du trafic.
(https://en.wikipedia.org/wiki/Federated_VoIP)



Source : <http://rtcquickstart.org/guide/multi/rtc-architecture-big-picture.html#arch-dia-1>

La "Federated VoIP" peut utiliser notamment ENUM comme système d'adressage, de localisation et d'identification des participants. Elle exploite des entrées DNS SRV à la manière des entrées MX pour le service SMTP. Elle implémente des communications sécurisées grâce à TLS et aux certificats X.509. Elle utilise les protocoles SIP et XMPP/Jabber pour établir les communications.

Kamailio

Kamailio est un serveur SIP libre (anciennement OpenSER)

<https://fr.wikipedia.org/wiki/Kamailio>

<https://www.kamailio.org/w/>

OpenSips

Fork de OpenSER

<http://www.opensips.org/>

Repro

https://www.resiprocate.org/About_Repro

Flexisip

<http://www.linphone.org/technical-corner/flexisip/overview>

4. Clients VoIP Open Source

Clients graphiques

Logiciel	Support OS	Licence	remarque
CSipSimple	Android	GNU GPL v3	conccurent avec le téléphone natif
Jitsi	GNU/Linux, Windows, Mac OS X (Java)	Apache v2	: -)
Linphone	GNU/Linux, Windows, Mac OS X	GNU GPL 2	ouvert
Zoiper	GNU/Linux, Windows, Mac OS X, Android, Apple Ios	Commercial/gratuit	Robuste
Blink	GNU/Linux, Windows, Mac OS X	Commercial/gratuit	SIPS, ZRTP

Librairies

- PJSIP : <http://www.pjsip.org/docs/book-latest/html/index.html>, <http://www.pjsip.org/pjsua.htm>
- Sipp : <http://sipp.sourceforge.net/>
- Linphonec : <http://www.linphone.org/technical-corner/linphone/documentation>

Exemples de périphériques SIP

- 1. [Polycom SoundPoint IP 321 \(3\)](#)
 - [Datasheet](#)
 - [Support](#)
- 2. [Cisco SPA508G \(3\)](#)
 - [Datasheet](#)
 - [Userguide \(French\)](#)
- 3. [Cisco SPA504G \(2\)](#)
 - [Datasheet and support](#)
- 5. [ATA Cisco PAP2T \(2\)](#)
 - [Datasheet](#)
- 6. [PoE FS108P \(4\)](#)
 - [Datasheet](#)
 - [Support](#)
- 7. [Cisco PoE SF300-24P \(1\)](#)
 - [Datasheet](#)
 - [Support](#)

1. Polycom SoundPoint IP 321 (3)



- 2 lines enterprise-grade SIP phone
- Excellent sound quality and enterprise-grade features in a simple, reliable IP deskphone.
- The last thing your business needs is another set of technologies to distract your teams from the business at hand. The most commonly requested phone features are the simple ones – often, all your teams need is a simple, reliable deskphone without the distractions created by advanced features.
- The SoundPoint IP 321 and 331 are two-line enterprise-grade IP deskphones with excellent sound quality. They are easy to use and are designed for small and medium-sized businesses and enterprises.

Benefit from:

- Two-line deskphones with excellent sound quality
- A set of enterprise-grade features
- Easy configuration and use
- Interoperability with leading IP PBX and Softswitch platforms

- SIP features

Datasheet

- <http://www.polycom.com/content/dam/polycom/www/documents/data-sheets/soundpoint-ip321-331-ds-enus.pdf>

Support

- http://support.polycom.com/PolycomService/support/us/support/voice/soundpoint_ip/soundpoint_ip321_331.html

2. Cisco SPA508G (3)



- 8-Line IP Phone with 2-Port Switch, PoE and LCD Display
- Full-featured 8-line business-class IP phone supporting Power over Ethernet (PoE)
- Monochrome backlit display for ease of use, aesthetics, and on-screen applications
- Connects directly to an Internet telephone service provider or to an IP private branch exchange (PBX)
- Wideband audio for unsurpassed voice clarity and enhanced speaker quality
- Easy installation and highly secure remote provisioning, as well as menu-based and web-based configuration

Datasheet

- http://www.cisco.com/en/US/prod/collateral/voicesw/ps6788/phones/ps10499/data_sheet_c78-548569.html

Userguide (French)

- http://www.cisco.com/en/US/docs/voice_ip_comm/csbpipp/ip_phones/user/guide/localized/50X/OL-24293.pdf

3. Cisco SPA504G (2)

4-Line IP Phone with 2-Port Switch, PoE and LCD Display

4. Cisco SPA922 (6)

1-line IP Phone with 2-port Switch



Datasheet and support

- <http://www.cisco.com/en/US/products/ps10036/index.html>

5. ATA Cisco PAP2T (2)

Internet Phone Adapter with 2 VoIP Ports



Datasheet

- http://www.cisco.com/en/US/prod/collateral/voicesw/ps6790/gatecont/ps10024/ps10029/data_sheet_c78-502441.html Support
- http://www.cisco.com/cisco/web/support/model/tsd_hardware_internetphoneadapter_model_pap2t.html

6. PoE FS108P (4)



- Unmanaged switch 4 ports PoE
- 8-PORT 10/100 SWITCH WITH 4-PORT POWER OVER ETHERNET
- Supply power to wireless access points and surveillance cameras over Cat-5 cabling
- IEEE 802.3af power on up to 4 ports
- 8 10/100 Mbps Ports with AutoUplink™

Datasheet

- <http://www.netgear.com/business/products/switches/unmanaged-switches/unmanaged-desktop-switches/FS108P.aspx>

Support

- FS108P Installation Guide : http://www.downloads.netgear.com/files/GDC/FS108P/FS108P_IGprt_11oct2.pdf
- FS108P Product Data Sheet : http://www.downloads.netgear.com/files/GDC/FS108P/FS108P_ds_1Dec04.pdf
- FS108P Quick Installation Guide (ZIP) : http://www.downloads.netgear.com/files/fs108p_quick_install_guide.zip

7. Cisco PoE SF300-24P (1)



Managed switch 24 PoE

- 24 10/100 PoE ports
- 2 10/100/1000 ports
- 2 combo mini-GBIC ports

Datasheet

- http://www.cisco.com/en/US/prod/collateral/switches/ps5718/ps10898/data_sheet_c78-610061.html

Support

- <http://www.google.com/search?q=SF300-24P+user+guide>

Exercices de mise en œuvre de l'infrastructure physique

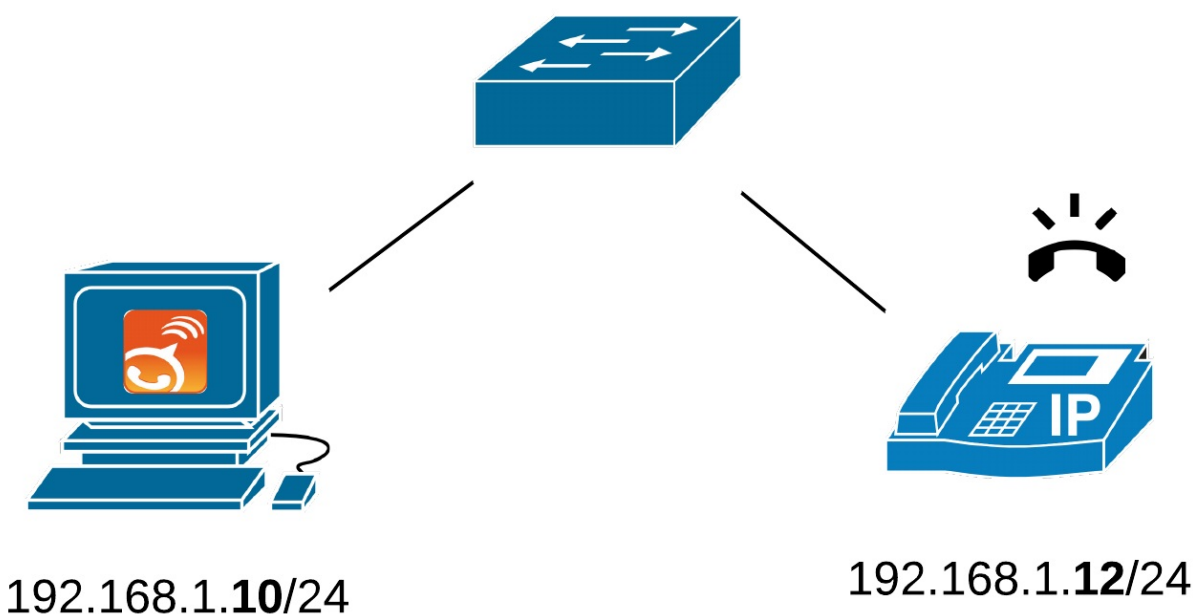
- 1. Mise en place de l'infrastructure
- 2. Connexion interne
- 3. Fournisseur Anveo
- 4. Connexion externe avec un softphone

1. Mise en place de l'infrastructure

- Prendre connaissance du manuel d'utilisation de chaque téléphone
- Trouver une solution d'alimentation du téléphone (PoE)
- Connecter le téléphone au réseau
- Remettre à zéro les paramètres du téléphone (DHCP)
- Trouver l'adresse de l'interface Web

2. Connexion interne

- Etablir une connexion SIP interne (softphone vers hardphone)



3. Fournisseur Anveo

- Prendre connaissance des offres du fournisseur de services VoIP Anveo
 - <http://anveo.com/business/service.asp>
 - <http://anveo.com/business/mainphonenumber.asp>
- Prendre connaissance des codes de connexions au fournisseur SIP Anveo
- Se connecter sur <https://anveo.com/logina.asp>

utilisateur	mot de passe	numéro	compte SIP
user@domain	-	+3225xxxxxx	453-----

- Accepter les conditions d'utilisation
- Configurer le compte Anveo

Sip Registration Details	
Username:	453 [REDACTED]
Password:	[REDACTED] Generate New
SIP Proxy Servers:	sip.anveo.com (USA) sip.ca.anveo.com (CANADA) sip.de.anveo.com (GERMANY) sip.hk.anveo.com (HONG KONG) DNS_SRV is supported Proxy port: 5010 (only use if your SIP device does not support DNS_SRV)
Audio codecs supported:	All Subscription Plans: G.711 ulaw (PCMu), GSM Premium Plans (Starter and up): G.722, G.729
Protocol:	UDP, TCP (only available on premium subscription plans: Starter and up)

Additional information which might be helpful

SIP Options

new SIP Status Alerts

SIP Options	
Dialing Format:	Localized Area code + number
	Default calling country: Belgium 32
	International dialing prefix: 00
Outbound CallerID ? :	3225 [REDACTED] - BELGIUM -
Internal CallerID/Default Extension ? :	
☒ Always include Country Code in Outbound CallerID ?	
☒ Enable MWI (Message Waiting Indicator) ?	
☒ Generate artificial ringback tone ?	
UPDATE	

Dial Features

***67XXXXXXXXXX** - Block CallerID for the call where XXXXXXXXXXXX is a phone number in the regular dialing format
***97** - Access ALL Voicemail without Voicemail box authentication
***99** - Access Voicemail box
***123** - Record Voicemail Greeting
***91** - Echo Test
****** - Record a call
#* - Attended transfer
***01** - Park a call (press Transfer key on IP phone and then enter *01)
***11** - Pickup parked call

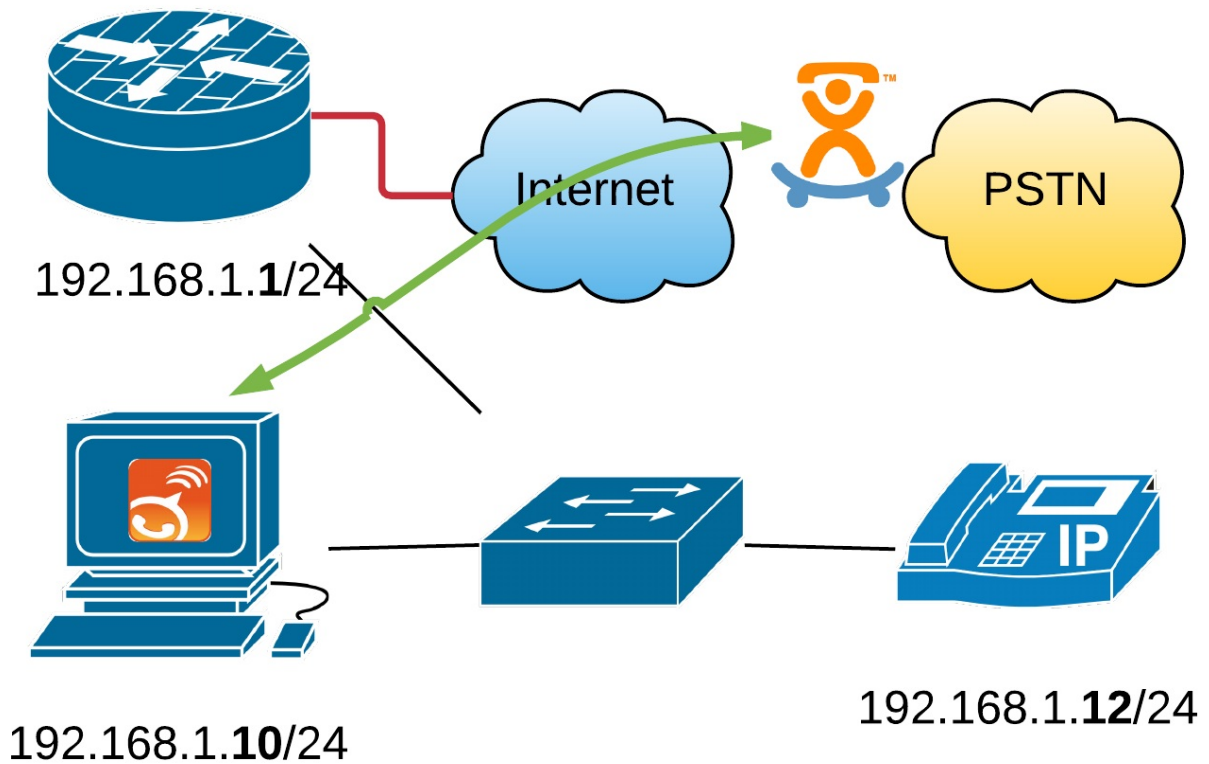
SIP Status ONLINE

Registration from: [REDACTED] 168.5000
 SIP UA: Linphone/3.9.1%20(belle-sip/1.4.2)
 Registered on: 12/11/2016 12:22:51
 Registration Expires: 12/11/2016 13:22:46
 Registration Server: Germany
 MWI Subscription on:

4. Connexion externe avec un softphone

Établir une connexion SIP vers l'extérieur

- Identité SIP : `sip:453-----@anveo.com`
- Adresse du proxy SIP : `sip:sip.de.anveo.com`



Programme de formation Wireshark

L'objectif de ce chapitre est de fournir une aide aux utilisateurs qui veulent capturer et analyser des communications L2 et TCP/IP avec Wireshark.

En voici le sommaire :

- [1. Analyseurs de paquets](#)
- [2. Analyse de paquets](#)
- [3. Placement de l'analyseur de paquet](#)
- [4. Introduction à Wireshark](#)
- [5. Menus de Wireshark](#)
- [6. Capture de paquets](#)
- [7. Travailler avec des captures Wireshark](#)
- [8. Statistiques Wireshark](#)
- [9. Analyse VoIP Wireshark](#)
- [10. Wireshark en ligne de commande](#)

Analyseurs de paquets

- [Définition](#)
- [Utilité](#)
- [Compétences à développer](#)

Définition

Un analyseur de paquets peut aussi être appelé en anglais : *packet analyzer*, *network analyzer*, *protocol analyzer* ou encore *packet sniffer*.

Wireshark est certainement le plus connu mais il en existe bien d'autres.

On citera en logiciels Open Source : `tcpdump` , `ngrep` , `winDump` , `tshark` , `dumpcap` , `capinfos` , `rawshark` , `editcap` , `mergcap` , `text2cap` , `reordercap` , ...

Par ailleurs, on remarquera le logiciel et le service en ligne CloudShark qui permet de présenter des captures en version Web (partages, commentaires, wireshark-like). Celui-ci ne remplit pas la fonction de capture qui est laissée à des logiciels de bas niveau.

Utilité

Analyser des paquets permet :

- de comprendre et d'apprendre les protocoles
- de reproduire leur comportement et de valider ces comportements
- de réaliser un audit de performance du réseau, d'identifier des problèmes dans une phase de diagnostic, d'implémenter du QoS *dans le cadre de la gestion de la bande passante*
- en cybersécurité, dans une *phase de reconnaissance passive* ou *active*, le *sniffing* permet d'interpréter les résultats d'une *prise d'empreinte par le réseau*
- dans un cadre plus défensif, les pots de miel (honeypots) et les systèmes de détection/prévention d'intrusions (IDS/IPS) utilisent la capture de trafic à des fins de journalisation ou de prise de décision
- En téléphonie, la capture de paquets aide à surveiller et à recomposer les conversations (dans un cadre légal strict : salles de marchés, services de centre d'appels, enquête légale, ...)

Compétences à développer

Les compétences à développer sont :

- L'identification des précises des hôtes et des utilisateurs d'une conversation au sein plongeée au sein d'un trafic dense.
- Faire inter-agir les conversations accessoires pour comprendre une conversation utile.
- Etre capable d'identifier la charge d'un conversation voire la restituer.

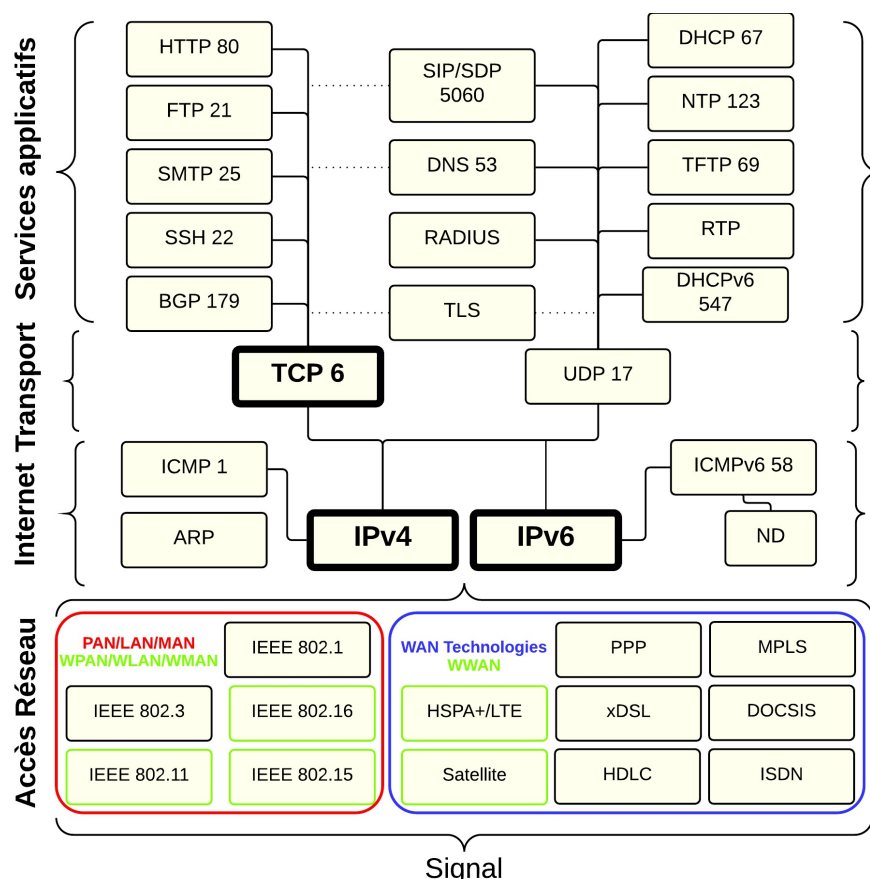
Analyse de paquets avec Cloudshark

- [Introduction](#)
- [Dialogues DNS, TCP et HTTP](#)
- [Scénario](#)
 - [Outils d'analyse](#)
 - [Points d'extrémité](#)
 - [Protocoles](#)
 - [Diagramme](#)
- [Trafic S](#)
 - [Query DNS](#)
 - [Query Response DNS](#)

Introduction

Dès que l'on dispose d'une vue formelle des modèles TCP/IP et OSI, une activité d'observation du trafic réseau permet de s'initier à des pratiques plus avancées grâce à un analyseur de paquets. Un analyseur de paquets est un logiciel qui se met à l'écoute d'une des interfaces de l'ordinateur et qui met en mémoire le trafic qui passe par elle. L'analyseur de paquets est capable de décoder, sauvegarder, traiter, analyser et présenter la capture.

- **Application**
 - Elle est la couche de communication qui s'interface avec les utilisateurs.
 - S'exécute sur les machines hôtes terminales.
- **Transport**
 - Elle est responsable du dialogue entre les hôtes terminaux d'une communication.
 - Les applications utiliseront TCP pour un transport fiable et UDP sans ce service.
 - Les routeurs NAT et les pare-feu opèrent un filtrage au niveau de la couche transport.
- **Internet**
 - Elle permet de déterminer les meilleurs chemins à travers les réseaux
 - Identifie globalement les interfaces
 - Les routeurs transfèrent le trafic IP qui ne leur est pas destiné.
- **Accès réseau**
 - Elle organise le flux binaire
 - identifie physiquement les interfaces
 - Elle règle la méthode d'accès au support
 - Elle place le flux binaire sur les support physique
 - Commutateurs, câbles, NIC,



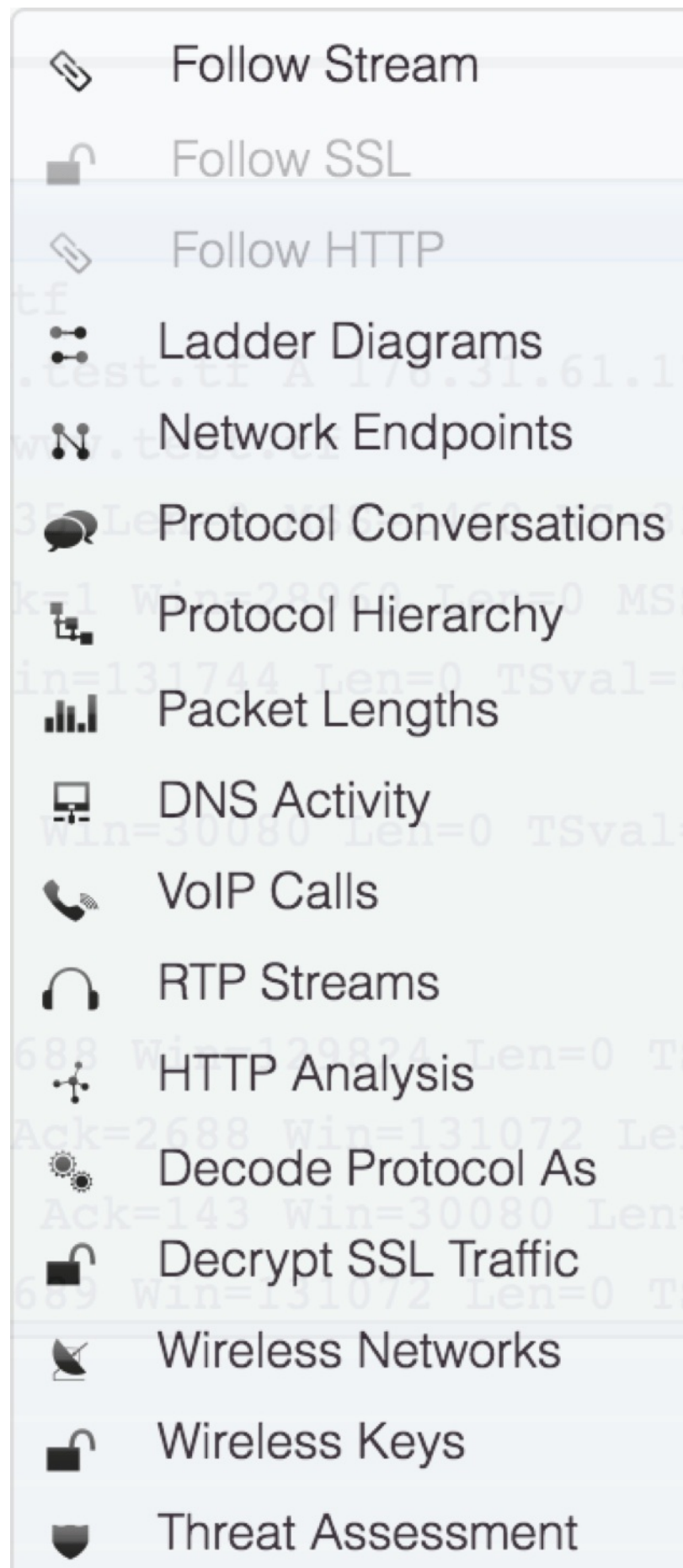
Dialogues DNS, TCP et HTTP

- Dans un navigateur Web, ouvrir la page Cloudshark <https://www.cloudshark.org/captures/26c43039ccd6>.

Scénario

DNS + GET HTTP /

Outils d'analyse



Points d'extrémité

"Network Endpoints"

Viewing 3 ipv4 Endpoints for [get http + dns](#)

Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
192.168.88.189	15	3874	8	691	7	3183
176.31.61.170	11	3574	5	3025	6	549
192.168.88.1	4	300	2	158	2	142

Protocoles

"Protocol Hierarchy"

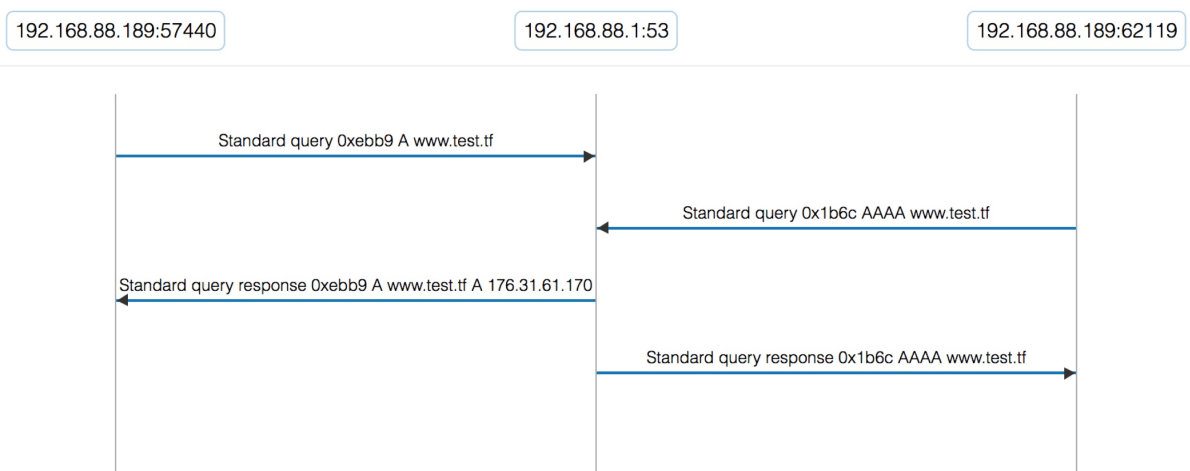
Protocol Hierarchy Stats for [get http + dns](#)

Click on the protocol name to apply a Display Filter for only that traffic.

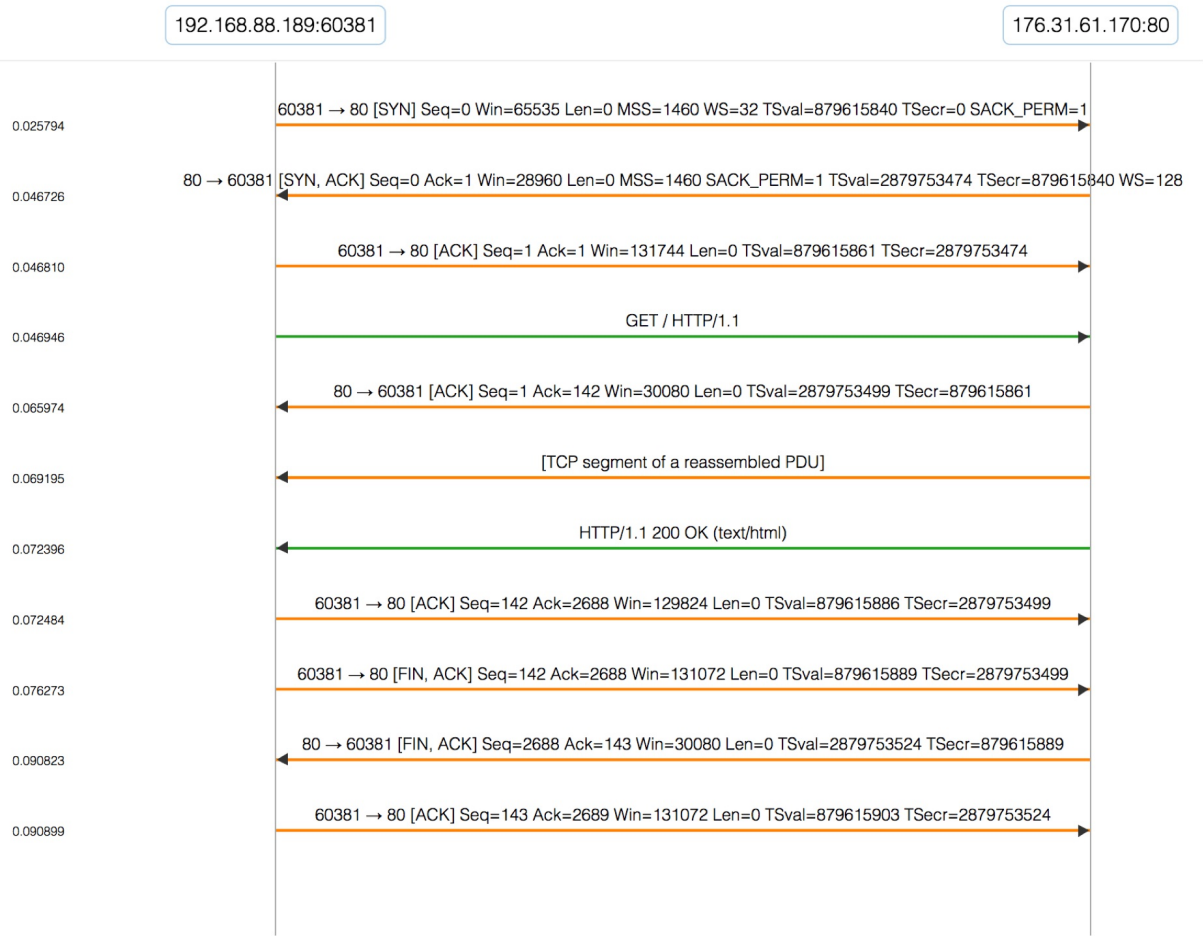
Protocol	Frames	Frames %	Bytes	Bytes %
frame	15	100.00	3874	100.00
eth	15	100.00	3874	100.00
ip	15	100.00	3874	100.00
udp	4	26.67	300	7.74
dns	4	26.67	300	7.74
tcp	11	73.33	3574	92.26
http	2	13.33	1520	39.24
data-text-lines	1	6.67	1313	33.89
tcp.segments	1	6.67	1313	33.89

Diagramme

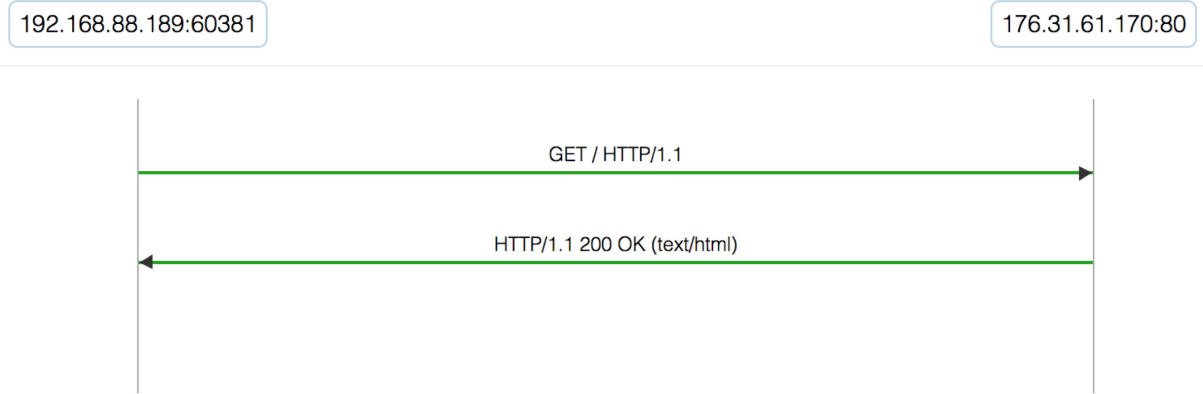
- Filtrer selon les protocoles `dns`, `tcp` et `http`.
- Noter dans un schéma les phases d'une connexion UDP.



- Noter le schéma d'une connexion TCP.



- Noter le schéma d'une connexion HTTP.



Trafic S

Query DNS

No.		Time	Source	Destination	Protocol	Length	Info
1	o	0.000000	192.168.88.189	192.168.88.1	DNS	71	Standard query 0xebb9 A www.test.tf
2	o	0.000092	192.168.88.189	192.168.88.1	DNS	71	Standard query 0x1b6c AAAA www.test.tf
3	o	0.024923	192.168.88.1	192.168.88.189	DNS	87	Standard query response 0xebb9 A www.test.tf A 176.31.61.170
4	o	0.025204	192.168.88.1	192.168.88.189	DNS	71	Standard query response 0x1b6c AAAA www.test.tf

▸ Frame 1: 71 bytes on wire (568 bits), 71 bytes captured (568 bits)

▼ Ethernet II, Src: Apple_bf:7c:37 (70:56:81:bf:7c:37), Dst: Tp-LinkT_e5:a2:58 (10:fe:ed:e5:a2:58)

▸ Destination: Tp-LinkT_e5:a2:58 (10:fe:ed:e5:a2:58)

▸ Source: Apple_bf:7c:37 (70:56:81:bf:7c:37)

Type: IPv4 (0x0800)

▸ Internet Protocol Version 4, Src: 192.168.88.189, Dst: 192.168.88.1

▼ User Datagram Protocol, Src Port: 57440 (57440), Dst Port: domain (53)

Source Port: 57440 (57440)

Destination Port: domain (53)

Length: 37

Checksum: 0x3105 [unverified]

Checksum Status: Unverified

Stream index: 0

▼ Domain Name System (query)

Transaction ID: 0xebb9

▸ Flags: 0x0100 Standard query

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 0

▼ Queries

▸ www.test.tf: type A, class IN

0000	10 fe ed e5 a2 58 70 56 81 bf 7c 37 08 00 45 00	XpV.. 7..E.
0010	00 39 d6 d1 00 00 ff 11 b2 d2 c0 a8 58 bd c0 a8	.9.....X...
0020	58 01 e0 60 00 35 00 25 31 05 eb b9 01 00 00 01	X..`.5.&1.....
0030	00 00 00 00 00 00 03 77 77 77 04 74 65 73 74 02	www.test.
0040	74 66 00 00 01 00 01	tf.....

Query Response DNS

No.		Time	Source	Destination	Protocol	Length	Info
1	o	0.000000	192.168.88.189	192.168.88.1	DNS	71	Standard query 0xebb9 A www.test.tf
2	o	0.000092	192.168.88.189	192.168.88.1	DNS	71	Standard query 0x1b6c AAAA www.test.tf
3	o	0.024923	192.168.88.1	192.168.88.189	DNS	87	Standard query response 0xebb9 A www.test.tf A 176.31.61.170
4	o	0.025204	192.168.88.1	192.168.88.189	DNS	71	Standard query response 0x1b6c AAAA www.test.tf

▸ Frame 3: 87 bytes on wire (696 bits), 87 bytes captured (696 bits)

▸ Ethernet II, Src: Tp-LinkT_e5:a2:58 (10:fe:ed:e5:a2:58), Dst: Apple_bf:7c:37 (70:56:81:bf:7c:37)

▸ Internet Protocol Version 4, Src: 192.168.88.1, Dst: 192.168.88.189

▼ User Datagram Protocol, Src Port: domain (53), Dst Port: 57440 (57440)

Source Port: domain (53)

Destination Port: 57440 (57440)

Length: 53

Checksum: 0xe6a7 [unverified]

Checksum Status: Unverified

Stream index: 0

▼ Domain Name System (response)

Request In: 1

Time: 0.024923000 seconds

Transaction ID: 0xebb9

▸ Flags: 0x8180 Standard query response, No error

Questions: 1

Answer RRs: 1

Authority RRs: 0

Additional RRs: 0

▼ Queries

▼ www.test.tf: type A, class IN

Name: www.test.tf

Name Length: 11

Label Count: 3

Type: A (Host Address) (1)

Class: IN (0x0001)

▼ Answers

▼ www.test.tf: type A, class IN, addr 176.31.61.170

Name: www.test.tf

Type: A (Host Address) (1)

Class: IN (0x0001)

Time to live: 3565

Data length: 4

Address: 176.31.61.170

Placement de l'analyseur de paquets

- Emplacement du trafic
 - Le support
 - L'interface de connexion
 - Mode "promiscuous"
 - Adresses de livraison
 - Concentrateur ou Hub
 - Commutateur
 - Routeurs
 - VLANs
- Méthodes d'écoute
 - Port mirroring
 - Hubbing out
 - Aggregated Taps
 - Attaques de redirection
 - Installation directe sur l'hôte à surveiller

Emplacement du trafic

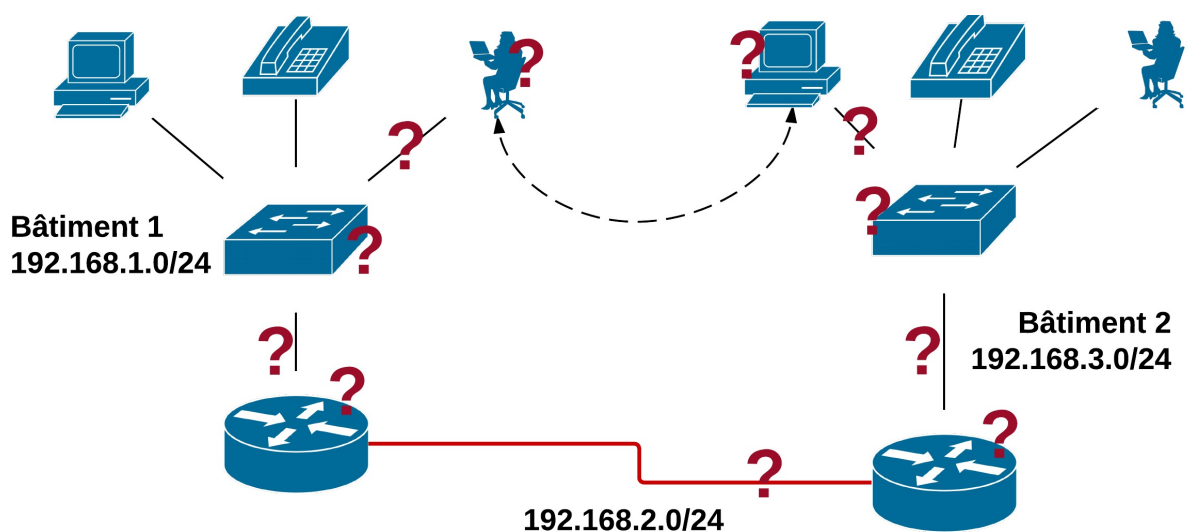
On pose ici la question du placement idéal du "sniffer" dans le réseau.

Le trafic passe par des supports auxquels des interfaces sont attachées. Ces interfaces permettent à des ordinateurs de communiquer entre eux.

D'une origine à une destination, le trafic peut passer par différents types de matériel :

- un commutateur
- un routeur

On représentera ces emplacements dans un diagramme :



Si on considère qu'il faille surveiller le trafic entre deux stations de travail, on pourrait placer le "sniffer" à chaque endroit marqué d'un point d'interrogation.

Mais avant d'envisager de désigner l'emplacement idéal, il est important de rappeler le rôle de ces différents composants.

Le support

- Cuivre : paire torsadée, câble coaxial (électricité, ondes électro-magnétiques)
- Fibre : fibre optique (ondes lumineuses)
- Air : ondes radios, ondes lumineuses

L'interface de connexion

C'est l'interface de communication sous un certain format qui connecte le support de transmission.

Une interface de connexion est liée à une puce électronique qui place les données sur le support et qui les transmet au traitement de l'ordinateur. C'est le rôle de la carte réseau (NIC)

Dans la technologie Ethernet, on connaît des interfaces de type RJ-45 pour la paire torsadée. Pour la fibre, il y a un grand nombre de type de connecteurs.

Mode "promiscuous"

La carte réseau (NIC) de l'analyseur de paquets va décider des paquets à traiter. Dans un mode "normal", la carte réseau de l'ordinateur qui exécute le logiciel de l'analyseur de paquet ne transmettra que les paquets destinés à l'ordinateur.

Placer la carte réseau en mode "promiscuous" (*promiscuité*), permet de transmettre à l'ordinateur et son logiciel de capture tous les paquets qui traversent le "fil" connecté, notamment ceux qui auraient été éliminés en mode "normal".

Cette capacité de promiscuité dépend de la nature de la carte réseau utilisée pour la capture mais surtout des pilotes de périphériques utilisés par le système d'exploitation et le logiciel. A cet égard, on trouvera une contrainte commerciale pour les captures avec Windows (pilotes payant Airpcap-NG) alors que les autres systèmes d'exploitation (Linux et MacOS X) l'autorise gratuitement.

Adresses de livraison

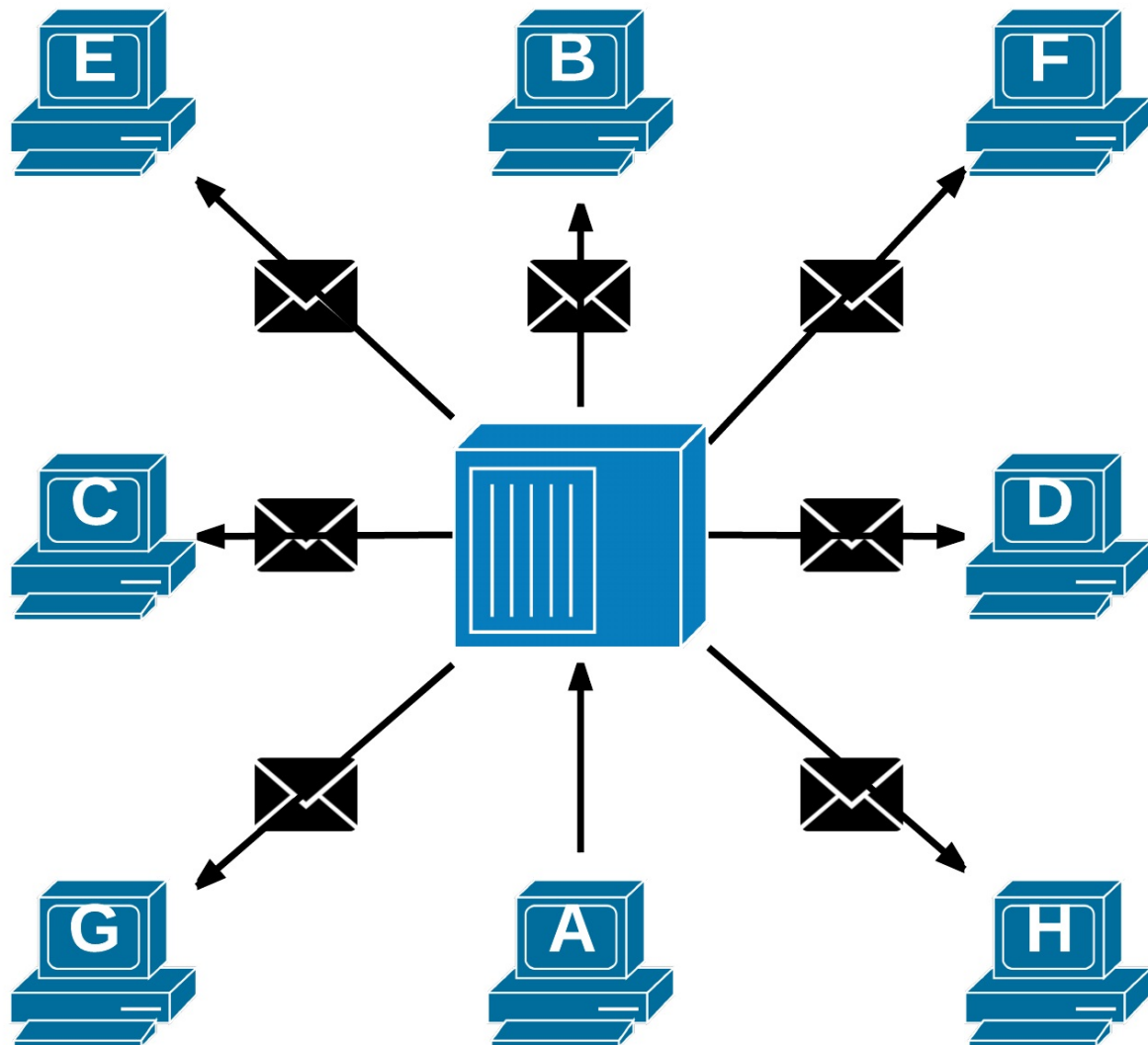
Concentrateur ou Hub

Un concentrateur Ethernet ou Hub est un périphérique du réseau ancestral dont l'objectif est de **concentrer** les connexions physiques. Le trafic est partagé par le *Hub* : toutes les interfaces attachées au concentrateur reçoivent le trafic que chacune place.

En technologie Ethernet, on dit qu'un "Hub" étend le domaine de collision (dans lequel les collisions Ethernet interviennent).

Cet type de périphérique a disparu de nos réseaux commutés. Il est franchement déconseillé de placer ce type de périphérique sur le réseau.

Mais sur le plan purement technique, étant donnée que ce matériel étend le fil, il fut intéressant de placer le "sniffer" à cet endroit.



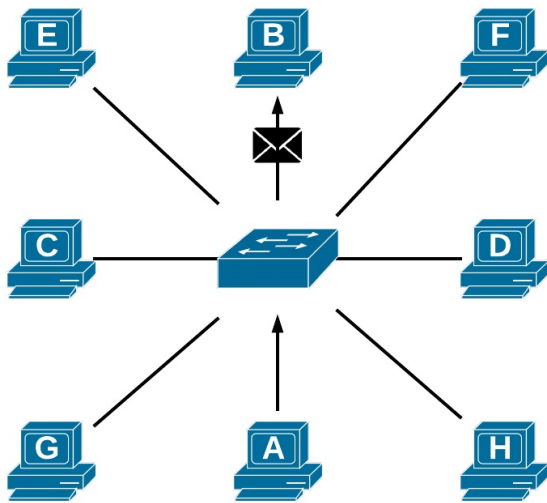
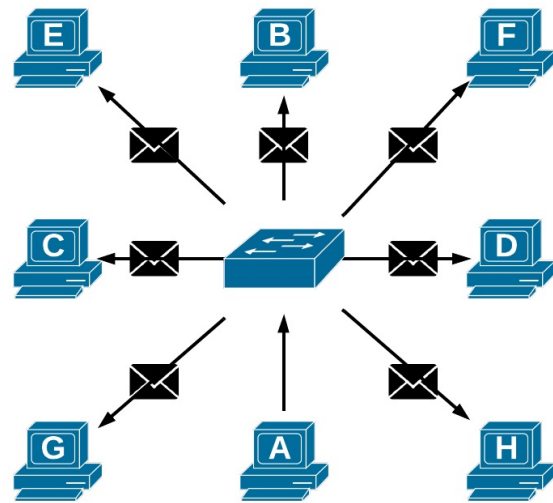
L'hôte A émet du trafic *unicast* ou *multicast*

Commutateur

Avec un commutateur, la livraison des paquets n'est plus partagé mais dédiée d'une interface à l'autre.

- Le commutateur prend ses décisions de transfert du trafic sur base des adresses MAC de destination (appries dynamiquement). Ce sont des composants matériel comme des puces spécialisées comme des ASIC qui prennent en charge la décision de transfert.
- Apprentissage dynamique : le commutateur apprend la ou les adresses MAC attachées à chacun de ses ports par écoute de trafic.
- Table CAM (Content Addressable Memory) : Le commutateur entretient une table de correspondance entre les ports et leurs adresses attachées. On peut vérifier la table de commutation d'un commutateur Cisco avec la commande suivante :

```
# show mac-address-table
```

L'hôte A émet du trafic *unicast* à destination de BL'hôte A émet du trafic *broadcast* ou *multicast*

- Si l'adresse MAC de destination du trafic est connue du commutateur, il le transfère sur le bon port de sortie aussi rapidement que si les deux partenaires de communication étaient directement connectés au même fil.
- Si le trafic dispose d'une destination
 - inconnue (trafic unicast inconnu du commutateur),
 - broadcast (FF:FF:FF:FF:FF:FF)
 - multicast
 - **il le transfère par tous les ports sauf le port d'origine**

En dédiant le transfert du trafic d'une interface à l'autre du LAN, le commutateur empêche l'interception des communications et donc la possibilité aisée de réaliser des captures. On proposera plus bas les méthodes adéquates pour réaliser ce genre de tâche dans un environnement commuté.

Routeurs

Le routeur transfère le trafic en fonction des adresses IP (IPv4 ou IPv6) pour des destinations distantes. Ils arrêtent le trafic de broadcast alors que les commutateurs et les "Hubs" le propagent.

VLANs

Pour diminuer la taille des domaines de broadcast, il est possible de diviser logiquement une seule infrastructure commutée en plusieurs LAN logiques : des LANs virtuels (VLANs). Ces LANs virtuels sont identifiés par un numéro.

Méthodes d'écoute

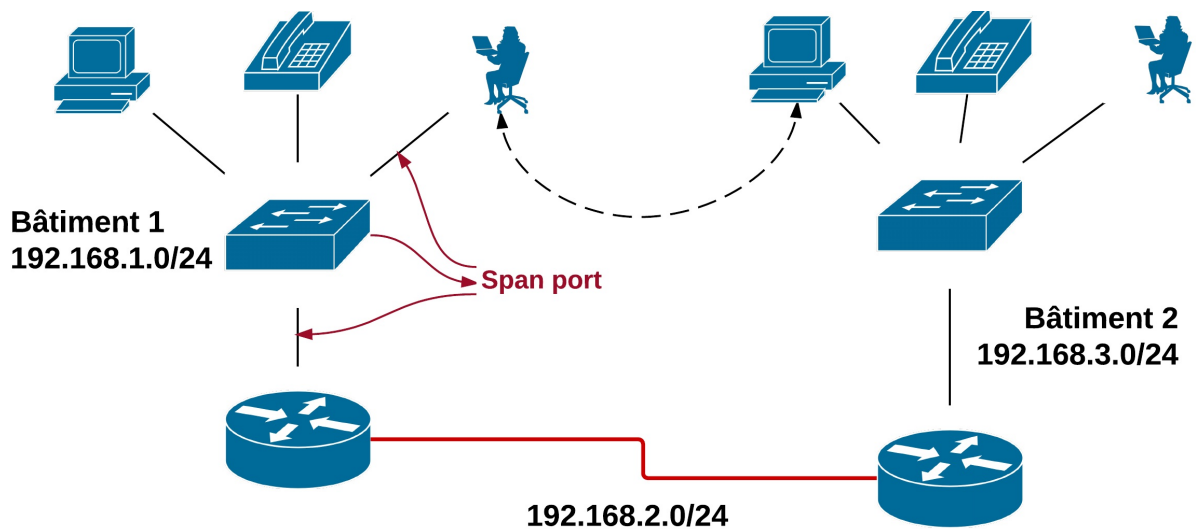
La méthode d'écoute dépendra :

- du type de dialogue à surveiller
- de l'environnement de cette communication (technologie supportée)
- de la maîtrise de cet environnement
- du coût à consentir pour réaliser la capture

Port mirroring

https://fr.wikipedia.org/wiki/Miroir_de_port

Le port mirroring, port monitoring, span port, est une fonction que l'on trouve sur un commutateur et qui consiste à transférer sur un port appelé par ce nom le trafic normalement dédié entre d'autres ports du commutateur.



- Cette méthode est habituellement préférée car elle n'ajoute aucune empreinte de trafic ou paquets supplémentaires.
- Elle est la seule qui permet de surveiller le trafic distant entre des commutateurs.
- Elle ne nécessite pas de déconnecter le client à surveiller.
- Elle est toutefois gourmande en ressources pour le commutateur.

Configuration Cisco :

```
! Monitor Session 1
monitor session 1 source interface Gi1/0/11 - 12 rx
monitor session 1 destination interface Gi1/0/21
!
! Monitor Session 2 - the default direction is both, so both does not appear in the
! command.
monitor session 2 source vlan 11
monitor session 2 destination interface Gi1/0/22
```

```
SW11# show monitor session all
Session 1
```

```
Type : Local Session
Source Ports :
RX Only : Gi1/0/1112
Destination Ports : Gi1/0/21
Encapsulation : Native
Ingress : Disabled
Session 2
```

```
Type : Local Session
Source VLANs :
Both : 11
Destination Ports : Gi1/0/22
Encapsulation : Native
Ingress : Disabled
```

```
SW11# show monitor detail
Session 1
```

```
Type : Local Session
Description :
Source Ports :
RX Only : Gi1/0/1112
TX Only : None
```

```
Both : None
Source VLANs :
RX Only : None
TX Only : None
Both : None
! Lines omitted for brevity
```

Hubbing out

Cette méthode est vue pour le principe. En effet, si l'on peut débrancher des hôtes à surveiller et que l'on se connecte au même "Hub", le sniffer verra tout son trafic.

- L'interface du client à surveiller peut être déconnectée.
- La méthode peut devenir inefficace pour surveiller du trafic entre plusieurs hôtes (collisions)
- La capacité d'un Hub étant à 10 Mbps, on pourrait connaître des pertes de paquets sur un réseau commuté à 100/1000 Mbps.

Aggregated Taps

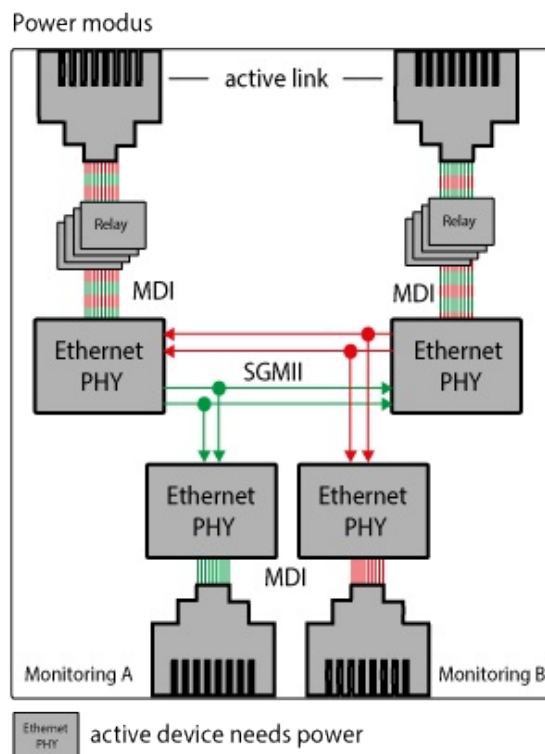
https://fr.wikipedia.org/wiki/TAP_r%C3%A9seau

- Aggregated Taps
- Nonaggregated Taps

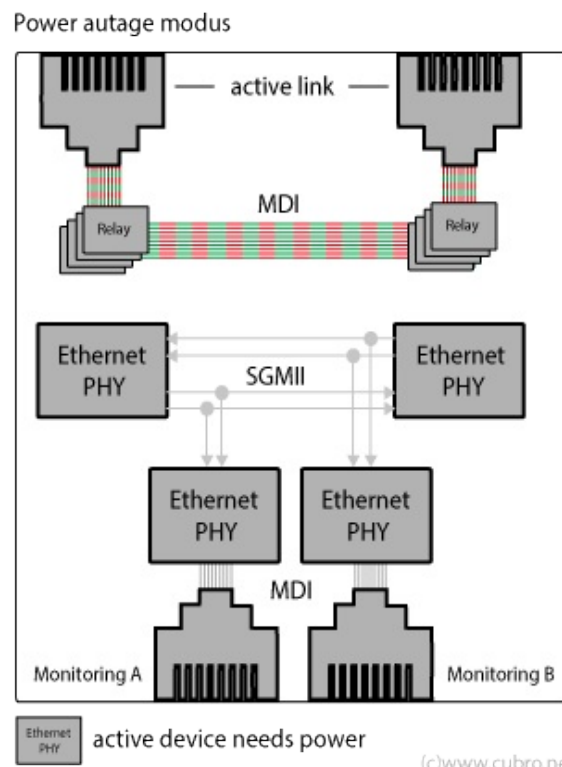
Ce périphérique matériel fournit un moyen d'accéder aux données circulant sur un réseau informatique. Le TAP réseau a (au moins) trois ports: un port A, un port B et un port de moniteur. Un tap réseau inséré entre A et B transmet tout le trafic sans obstacle, mais copie également ces mêmes données sur son port de moniteur, ce qui permet à un tiers d'écouter ledit trafic.

Il y a des TAPs pour chaque type de réseau : en cuivre (RJ45), en fibre optique. Mais le type de TAP peut varier, certains peuvent dupliquer le signal réseau sur plusieurs sorties (appelés regeneration TAPs) tandis que d'autres font de l'agrégation en rassemblant en un seul flux les données d'un réseau half duplex ou full duplex.

Common Gbit Copper TAP



CUBRO(c)



- L'interface du client à surveiller peut être déconnectée.
- Permet de capturer du trafic sur des **interfaces fibres** qui deviennent de plus en plus communes dans les installations LAN/WAN.
- Supporte le transfert du trafic commuté à ses vitesses (10/1000 Mbps Full Duplex).
- Matériel est assez coûteux.

Attaques de redirection

- APR
- Attaques 802.1
- DHCP, DNS, Routing : SuddenSix
- Méthode de capture la plus lente, elle nécessite la création de paquets supplémentaires pour rediriger le trafic vers l'analyseur de paquets.
- Il s'agit plutôt d'une méthode offensive qui s'impose lorsque l'on ne peut pas prendre la main sur l'infrastructure.
- Elle n'est donc transparente.
- Elle peut susciter des problèmes de performance sur le réseau (lenteur constatable).

Installation directe sur l'hôte à surveiller

Sauf pour des tests, des constitutions de "baseline", cette méthode n'est pas recommandée car le trafic pourrait être perdu ou transformé.

Introduction à Wireshark

- 1. Introduction à Wireshark
 - 1.1. Fonctionnalités de Wireshark
 - 1.2. Prérequis système
 - Microsoft Windows
 - UNIX / Linux
- 2. Obtenir Wireshark
- 3. Installation de Wireshark
 - 3.1. Installation de Wireshark sous Windows
 - Choix des composants d'installation
 - Tâches supplémentaires
 - Choix de l'emplacement d'installation
 - Installation de WinPcap
 - Options en ligne de commande de l'installateur Windows
 - Installation manuelle de WinPcap
 - Mettre à jour Wireshark
 - Mettre à jour WinPcap
 - Désinstaller Wireshark
 - Désinstaller WinPcap
 - 3.2. Installation de Wireshark sous MacOS
 - 3.3. Installation des fichiers binaires sous UNIX
 - Installation à partir de RPM sous Red Hat et similaire
 - Installation Dpkg sous Debian, Ubuntu et autres dérivés Debian
 - Installation sous Gentoo Linux avec portage
 - Installation sous FreeBSD avec packages
- Sources

Version : Wireshark 2.1

1. Introduction à Wireshark

Wireshark est un analyseur de paquets open source (GNU) populaire. Ses "dissectors" ou décodeurs de protocoles permettent d'interpréter le trafic du réseau.

Conçu en 1997-1998 par Gerald Combs sous le nom historique de "Ethereal", il est repris en 2006 sous le nom moderne de "Wireshark". En 2008, Wireshark sort en version 1.0 et en 2015 en version 2.0 avec une nouvelle interface graphique.

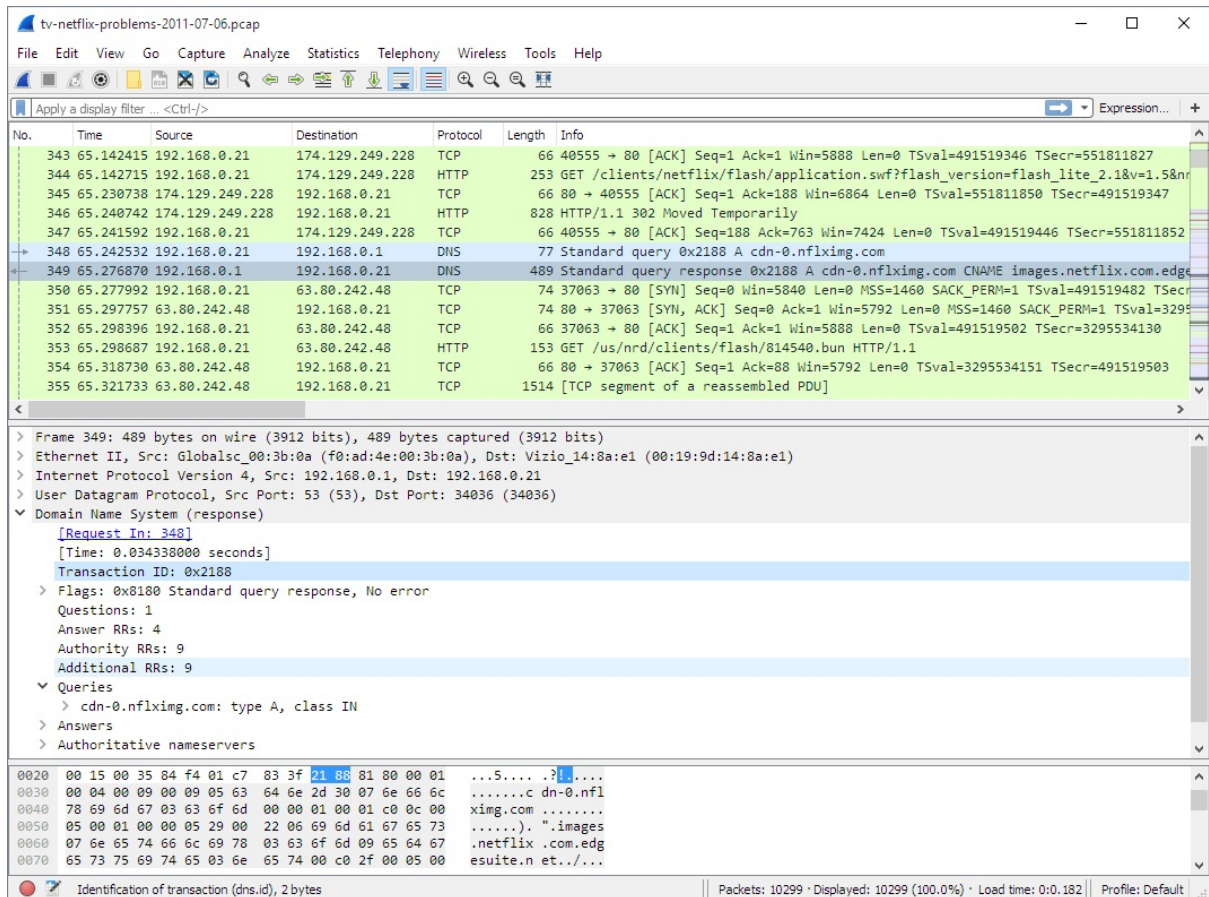
1.1. Fonctionnalités de Wireshark

Les fonctionnalités principales de Wireshark sont :

- Disponibles pour les systèmes *UNIX* et *Windows*.
- *Capter* les packets de données en "live" qui passent en live sur les interfaces à partir de n'importe quel type de supports : Ethernet, Wi-Fi, Bluetooth, Frame-Relay, ATM, HDLC, USB, ...
(<https://wiki.wireshark.org/CaptureSetup/NetworkMedia>).
- *Ouvrir* des fichiers de captures de paquets réalisés avec tcpdump/WinDump, Wireshark et bien d'autres programmes.
- *Importer* des paquets venant de fichiers texte contenant les charges en hexa de paquets de données.

- Display packets with *very detailed protocol information*.
- *Enregistrer* des paquets de données capturés.
- *Exporter* certains ou tous les paquets capturés dans différents formats.
- *Filtrer les paquets* sur base de différents critères.
- *Rechercher* des paquets sur base de différents critères.
- *Coloriser* des paquets sur base de différents critères.
- Créer différentes *statistiques*
- ...

Figure 1.1. Wireshark captures packets and lets you examine their contents.



Wireshark n'est ni un IDS/IPS qui réagirait à des emprunts de trafic, ni un forgeur de paquet qui manipule le trafic.

1.2. Prérequis système

La quantité de ressources requise par Wireshark dépend de votre environnement et de la taille du fichier de capture que vous analysez. Les valeurs ci-dessous devraient être adaptées pour les fichiers de capture de petite à moyenne taille de quelques centaines de Mo. Les fichiers de capture plus importants nécessiteront plus de mémoire et d'espace disque.

Les réseaux occupés signifient de grandes captures

Travailler avec un réseau occupé peut facilement produire d'énormes fichiers de capture. La capture sur un réseau gigabit ou même 100 mégabits peut produire des centaines de mégaoctets de données de capture en peu de temps. Un processeur rapide, beaucoup de mémoire et d'espace disque est toujours une bonne idée.

Bien que Wireshark capture des paquets en utilisant un processus distinct, l'interface principale est simple et ne bénéficiera pas beaucoup des systèmes multi-core.

Microsoft Windows

- Les version de Wireshark supporte les versions de Windows avec [extended support lifetime](#) : Windows 10, 8, 7, Vista, Server 2016, Server 2012 R2, Server 2012, Server 2008 R2, and Server 2008.
- Processeur 64-bit AMD64/x86-64 or 32-bit x86 processor.
- 400 MB RAM disponible. Des fichiers de captures important nécessitent plus de RAM.
- 300 MB d'espace disque pour l'installation.
- Graphique 1024×768 (1280×1024 or ou plus recommandé) resolution en couleur 16 bits.
- Une carte réseau (NIC) supportée.
 - Ethernet. N'importe quelle carte devrait être fonctionnelle. Voir <https://wiki.wireshark.org/CaptureSetup/Ethernet> et <https://wiki.wireshark.org/CaptureSetup/Offloading> en cas problèmes.
 - 802.11. Capturer du trafic réseau Wi-Fi sans matériel adapté peut être difficile sous Windows : [Wireshark wiki page](#). Capturing raw 802.11 information may be difficult without special equipment.
 - Autres supports. Voir <https://wiki.wireshark.org/CaptureSetup/NetworkMedia>.

UNIX / Linux

Wireshark fonctionne sur la plupart des systèmes UNIX, UNIX-like comme par exemple macOS ou Linux. Les pré-requis systèmes sont identiques à ceux d'une installation Windows.

Les binaires d'installation sont mis à disposition pour les systèmes suivants :

- Apple macOS
- Debian GNU/Linux
- FreeBSD
- Gentoo Linux
- HP-UX
- Mandriva Linux
- NetBSD
- OpenPKG
- Red Hat Enterprise/Fedora Linux
- Sun Solaris/i386
- Sun Solaris/SPARC
- Canonical Ubuntu

2. Obtenir Wireshark

On peut obtenir Wireshark à partir de son site Web : <https://www.wireshark.org/download.html>. La page de téléchargement vous redirige automatiquement sur le bon téléchargement pour votre système en utilisant le miroir Internet le plus proche de chez vous. Les installateurs officiels Windows et macOS sont signés par la **Wireshark Foundation**.

Une nouvelle version de Wireshark est publiée tous les mois ou tous les deux mois.

3. Installation de Wireshark

3.1. Installation de Wireshark sous Windows

L'installateur Wireshark contient le type de système et sa version : Wireshark-win64-2.3.0.exe installe Wireshark 2.3.0 pour Windows 64-bit. L'installateur Wireshark installe le pilote WinPcap indispensable aux captures.

Choix des composants d'installation

On the *Choose Components* page of the installer you can select from the following:

- **Wireshark** - Le logiciel lui-même.
- **TShark** - Un outil en ligne de commande.
- **Wireshark 1 Legacy** - L'ancienne interface graphique.
- **Plugins & Extensions** - Des extras pour Wireshark et Tshark en matière de "dissecteurs".
 - **Dissector Plugins** - Plug-ins "dissecteurs".
 - **Tree Statistics Plugins** - Statistiques étendues.
 - **Mate - Meta Analysis and Tracing Engine** - Extensions utilisateurs d'affichage, de filtrage, voir <https://wiki.wireshark.org/Mate>](<https://wiki.wireshark.org/Mate>).
 - **SNMP MIBs** - dissection des MIBs SNMP.
- **Tools** - Outils supplémentaires de traitement des fichiers de capture.
 - **Editcap** - Lit un fichier de capture et écrit certains ou tous les paquets dans un autre fichier de capture.
 - **Text2Pcap** - Lecture dans un vaisseau hexadécimal ASCII et écrit les données dans un fichier de capture de pcap.
 - **Reordercap** - Réorganise un fichier de capture par timestamp.
 - **Mergecap** - Combine plusieurs fichiers de capture enregistrés dans un seul fichier de sortie.
 - **Capinfos** - Fournit des informations sur les fichiers de capture.
 - **Rawshark** - Filtre en paquet brut.
 - **Guide de l'utilisateur** - Installation locale du Guide de l'utilisateur. Les boutons d'aide de la plupart des dialogues nécessiteront une connexion Internet.

Tâches supplémentaires

- **Raccourcis du menu Démarrer** - Ajouter des raccourcis du menu de démarrage.
- **Icône de bureau** - Ajouter une icône Wireshark au bureau.
- **Icône de lancement rapide** - Ajouter une icône Wireshark à la barre d'outils de lancement rapide de l'Explorer.
- Associer des extensions de fichiers à Wireshark.

Choix de l'emplacement d'installation

Par défaut, Wireshark s'installe dans `%ProgramFiles%\Wireshark` sur Windows 32 bits et `%ProgramFiles64%\Wireshark` sur Windows 64 bits. (`C:\Program Files\Wireshark` sur la plupart des systèmes).

Installation de WinPcap

L'installation de Wireshark contient le dernier programme d'installation de WinPcap.

Si vous n'avez pas installé WinPcap, vous ne pourrez pas capturer le trafic réseau en direct, mais vous pourrez toujours ouvrir des fichiers de capture enregistrés. Par défaut, la dernière version de WinPcap sera installée.

Pour plus d'informations sur WinPcap, consultez <https://www.winpcap.org/> et <https://wiki.wireshark.org/WinPcap>.

Options en ligne de commande de l'installateur Windows

- `/s` exécute le programme d'installation ou de désinstallation en silence avec des valeurs par défaut. L'installateur silencieux **n'installera pas WinPcap**.
- `/desktopicon` installation de l'icône du bureau, `=yes` - force l'installation, `=no` - ne pas installer, sinon utilisez les paramètres par défaut. Cette option peut être utile pour un installateur silencieux.
- `/quicklaunchicon` installation de l'icône de lancement rapide, `=yes` - forcer l'installation, `=no` - ne pas installer, sinon utiliser les paramètres par défaut.
- `/D` éfinit le répertoire d'installation par défaut (`\$INSTDIR`), en remplaçant `InstallDir` et `InstallDirRegKey` . Il doit être le dernier paramètre utilisé dans la ligne de commande et ne doit pas contenir de guillemets même si le chemin contient des espaces.
- `/NCRC` désactive la vérification CRC. Nous vous recommandons de ne pas utiliser ce drapeau.

Exemple:

```
> Wireshark-win64-wireshark-2.0.5.exe /NCRC /S /desktopicon=yes /quicklaunchicon=no /D=C:\Program Files\Foo
```

L'exécution du programme d'installation sans paramètres montre l'installateur interactif normal.

Installation manuelle de WinPcap

Comme mentionné ci-dessus, l'installateur Wireshark prend soin d'installer WinPcap. Ce qui suit n'est nécessaire que si vous souhaitez utiliser une version différente de celle incluse dans l'installateur Wireshark, par ex. Car une nouvelle version de WinPcap a été diffusée.

D'autres versions de WinPcap (y compris les versions alpha ou bêta plus récentes) peuvent être téléchargées depuis le site principal de WinPcap à <https://www.winpcap.org/>.

Mettre à jour Wireshark

Par défaut, le paquet Windows officiel vérifiera les nouvelles versions et vous informera quand elles seront disponibles. Si vous avez désactivé la requête *Vérifier les mises à jour* ou si vous exécutez Wireshark dans un environnement isolé, vous devez vous inscrire à la liste de diffusion *wireshark-announce*.

Les nouvelles versions de Wireshark sont généralement diffusées toutes les quatre à six semaines. La mise à jour de Wireshark se fait de la même manière que l'installation. Il suffit de télécharger et de démarrer l'installateur exe. Un redémarrage n'est généralement pas nécessaire et tous vos paramètres personnels restent inchangés.

Mettre à jour WinPcap

Les nouvelles versions de WinPcap sont moins fréquentes. Vous trouverez les instructions de mise à jour de WinPcap dans le site Web de WinPcap à <https://www.winpcap.org/>. Vous devrez peut-être redémarrer votre machine après avoir installé une nouvelle version de WinPcap.

Désinstaller Wireshark

Vous pouvez désinstaller Wireshark à l'aide du panneau de configuration *Programmes et fonctionnalités*. Sélectionnez l'entrée "Wireshark" pour lancer la procédure de désinstallation.

Le désinstallateur Wireshark offre plusieurs options de suppression. La valeur par défaut consiste à supprimer les composants principaux, mais à garder vos paramètres personnels et WinPcap. WinPcap est installé par défaut dans le cas où d'autres programmes en ont besoin.

Désinstaller WinPcap

Vous pouvez désinstaller WinPcap indépendamment de Wireshark en utilisant l'entrée *WinPcap* dans le panneau de configuration *Programmes et fonctionnalités*. N'oubliez pas que si vous désinstallez WinPcap, vous ne pourrez rien capturer avec Wireshark.

3.2. Installation de Wireshark sous MacOS

Les paquets MacOS officiels sont distribués sous forme d'images de disque (.dmg) contenant l'installateur de l'application. Pour installer Wireshark, ouvrez simplement l'image du disque et exécutez le programme d'installation ci-joint.

Le package d'installation comprend Wireshark, ses utilitaires de ligne de commande associés et un démon de lancement qui ajuste les autorisations de capture lors du démarrage du système. Consultez le fichier inclus *Read me first* pour plus de détails.

3.3. Installation des fichiers binaires sous UNIX

En général, l'installation de la binaire sous votre version d'UNIX sera spécifique aux méthodes d'installation utilisées avec votre version d'UNIX. Par exemple, sous AIX, vous utiliserez *smft* pour installer le package binaire Wireshark, alors que sous Tru64 UNIX (anciennement Digital UNIX), vous utiliserez *setld*.

Installation à partir de RPM sous Red Hat et similaire

La création de RPM à partir du code source de Wireshark génère plusieurs paquets (la plupart des distributions suivent le même système) :

- Le paquet `wireshark` contient les bibliothèques Wireshark principales et les outils de ligne de commande.
- Le paquet `wireshark-qt` contient l'interface graphique basée sur Qt.
- Le paquet `wireshark-gtk` (anciennement `wireshark-gnome`) contient l'interface graphique basée sur Gtk +.

De nombreuses distributions utilisent `yum` ou un outil de gestion de package similaire pour faciliter l'installation de logiciels (y compris ses dépendances). Si votre distribution utilise `yum`, utilisez la commande suivante pour installer Wireshark avec l'interface graphique Qt :

```
yum install wireshark wireshark-qt
```

Si vous avez créé vos propres RPM à partir des sources Wireshark, vous pouvez les installer en exécutant, par exemple :

```
rpm -ivh wireshark-2.0.0-1.x86_64.rpm wireshark-qt-2.0.0-1.x86_64.rpm
```

Si la commande ci-dessus échoue en raison des dépendances manquantes, installez d'abord les dépendances, puis réessayez l'étape ci-dessus.

Installation Dpkg sous Debian, Ubuntu et autres dérivés Debian

Si vous pouvez simplement installer à partir du dépôt, utilisez :

```
$ aptitude install wireshark
```

Aptitude devrait s'occuper de tous les problèmes de dépendance pour vous.

Utilisez la commande suivante pour installer le fichier deb de Wireshark sous Debian :

```
$ dpkg -i wireshark-common_2.0.5.0-1_i386.deb wireshark-wireshark-2.0.5.0-1_i386.deb
```

Dans ce dernier cas, Dpkg ne prend pas en charge toutes les dépendances, mais indique ce qui manque.

La capture nécessite des privilèges

En installant des packages Wireshark, les utilisateurs non root n'obtiennent pas de droits automatiquement pour capturer des paquets. Pour permettre aux utilisateurs non-root de capturer des paquets, suivez la procédure décrite dans `/usr/share/doc/wireshark-common/README.Debian`.

Installation sous Gentoo Linux avec portage

Utilisez la commande suivante pour installer Wireshark sous Gentoo Linux avec toutes les fonctionnalités supplémentaires :

```
$ USE="c-ares gtk ipv6 portaudio snmp ssl kerberos threads selinux" emerge wireshark
```

Installation sous FreeBSD avec packages

Utilisez la commande suivante pour installer Wireshark sous FreeBSD :

```
$ pkg_add -r wireshark
```

La commande `pkg_add` devrait prendre en charge tous les problèmes de dépendance pour vous.

Sources

- https://www.wireshark.org/docs/wsug_html/#ChapterIntroduction
- https://www.wireshark.org/docs/wsug_html/#ChapterBuildInstall
- https://www.wireshark.org/docs/wsug_html/#ChapterUsing

Menus Wireshark

- 1. Démarrer Wireshark
- 2. The Main window
 - Main Window Navigation
- 3. The Menu
- 3. The “File” menu
 - The “Edit” menu
 - The “View” menu
 - The “Go” menu
 - The “Capture” menu
 - The “Analyze” menu
 - The “Statistics” menu
 - The “Telephony” menu
 - The “Tools” menu
 - The “Internals” menu
 - The “Help” menu
- 4. Toolbars
 - The “Main” toolbar
 - The “Filter” toolbar
- 5. Panes
 - The “Packet List” pane
 - The “Packet Details” pane
 - The “Packet Bytes” pane
- 6. The Statusbar
- Sources

Texte en anglais

1. Démarrer Wireshark

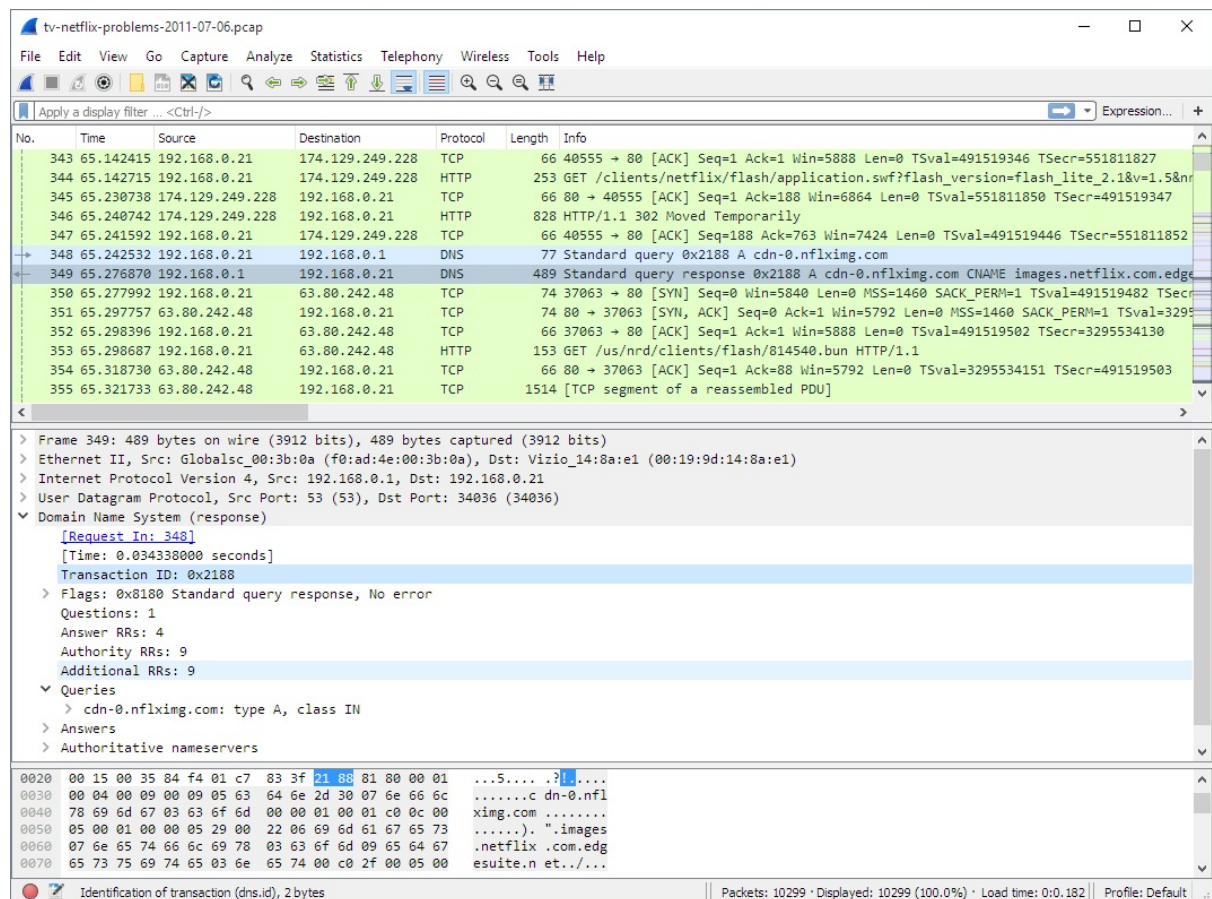
Vous pouvez démarrer Wireshark depuis votre gestionnaire de fenêtre ou terminal.

Dans les chapitres suivants, beaucoup de captures d'écran de Wireshark seront affichées. Comme Wireshark fonctionne sur de nombreuses plates-formes différentes avec différents gestionnaires de fenêtres, différents styles appliqués et il existe différentes versions de la boîte à outils GUI sous-jacente utilisée, votre écran peut être différent des captures d'écran fournies. Mais ceux-ci ne sont toujours pas compréhensibles.

2. The Main window

Regardons l'interface utilisateur de Wireshark. La Figure 3.1, "La fenêtre principale" montre Wireshark comme vous le verriez habituellement après que certains paquets sont capturés ou chargés (comment cela sera décrit plus loin).

Figure 3.1. La fenêtre principale



La fenêtre principale de Wireshark se compose de parties connues de plusieurs autres programmes GUI.

1. La *menu* sert à lancer des actions.
2. La *barre d'outils principale* permet d'accéder rapidement aux éléments fréquemment utilisés dans le menu.
3. La barre d'outils *filter*.
4. Le volet *liste des paquets* affiche un résumé de chaque paquet capturé. En cliquant sur les paquets dans ce volet, vous contrôlez ce qui s'affiche dans les deux autres panneaux.
5. Le volet *des détails du paquet* affiche le paquet sélectionné dans le volet de la liste des paquets plus en détail.
6. Le volet *paquets bytes* affiche les données du paquet sélectionné dans le volet de la liste des paquets et met en surbrillance le champ sélectionné dans le volet des détails du paquet.
7. La *barre d'état* montre des informations détaillées sur l'état actuel du programme et les données capturées.

The layout of the main window can be customized by changing preference settings.

Main Window Navigation

Packet list and detail navigation can be done entirely from the keyboard. [Table 3.1, “Keyboard Navigation”](#) shows a list of keystrokes that will let you quickly move around a capture file. for additional navigation keystrokes.

Table 3.1. Keyboard Navigation

Accelerator	Description
Tab, Shift+Tab	Move between screen elements, e.g. from the toolbars to the packet list to the packet detail.
Down	Move to the next packet or detail item.

Up	Move to the previous packet or detail item.
Ctrl+Down, F8	Move to the next packet, even if the packet list isn't focused.
Ctrl+Up, F7	Move to the previous packet, even if the packet list isn't focused.
Ctrl+.	Move to the next packet of the conversation (TCP, UDP or IP).
Ctrl+,	Move to the previous packet of the conversation (TCP, UDP or IP).
Alt+Right or Option+Right (macOS)	Move to the next packet in the selection history.
Alt+Left or Option+Right (macOS)	Move to the previous packet in the selection history.
Left	In the packet detail, closes the selected tree item. If it's already closed, jumps to the parent node.
Right	In the packet detail, opens the selected tree item.
Shift+Right	In the packet detail, opens the selected tree item and all of its subtrees.
Ctrl+Right	In the packet detail, opens all tree items.
Ctrl+Left	In the packet detail, closes all tree items.
Backspace	In the packet detail, jumps to the parent node.
Return, Enter	In the packet detail, toggles the selected tree item.

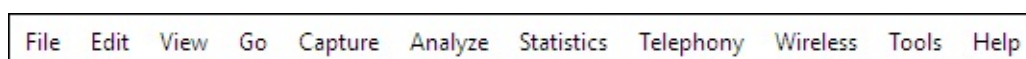
Help → About Wireshark → Keyboard Shortcuts will show a list of all shortcuts in the main window. Additionally, typing anywhere in the main window will start filling in a display filter.

3. The Menu

Wireshark's main menu is located either at the top of the main window (Windows, Linux) or at the top of your main screen (macOS). An example is shown in [Figure 3.2, "The Menu"](#).

Some menu items will be disabled (greyed out) if the corresponding feature isn't available. For example, you cannot save a capture file if you haven't captured or loaded any packets.

Figure 3.2. The Menu



The main menu contains the following items:

File

This menu contains items to open and merge capture files, save, print, or export capture files in whole or in part, and to quit the Wireshark application.

Edit

This menu contains items to find a packet, time reference or mark one or more packets, handle configuration profiles, and set your preferences; (cut, copy, and paste are not presently implemented).

View

This menu controls the display of the captured data, including colorization of packets, zooming the font, showing a packet in a separate window, expanding and collapsing trees in packet details,

Go

This menu contains items to go to a specific packet.

Capture

This menu allows you to start and stop captures and to edit capture filters.

Analyze

This menu contains items to manipulate display filters, enable or disable the dissection of protocols, configure user specified decodes and follow a TCP stream.

Statistics

This menu contains items to display various statistic windows, including a summary of the packets that have been captured, display protocol hierarchy statistics and much more.

Telephony

This menu contains items to display various telephony related statistic windows, including a media analysis, flow diagrams, display protocol hierarchy statistics and much more.

Wireless

The items in this menu show Bluetooth and IEEE 802.11 wireless statistics.

Tools

This menu contains various tools available in Wireshark, such as creating Firewall ACL Rules.

Help

This menu contains items to help the user, e.g. access to some basic help, manual pages of the various command line tools, online access to some of the webpages, and the usual about dialog.

Each of these menu items is described in more detail in the sections that follow.

Shortcuts make life easier

Most common menu items have keyboard shortcuts. For example, you can press the Control (or Strg in German) and the K keys together to open the “Capture Options” dialog.

3. The “File” menu

The Wireshark file menu contains the fields shown in [Table 3.2, “File menu items”](#).

Figure 3.3. The “File” Menu

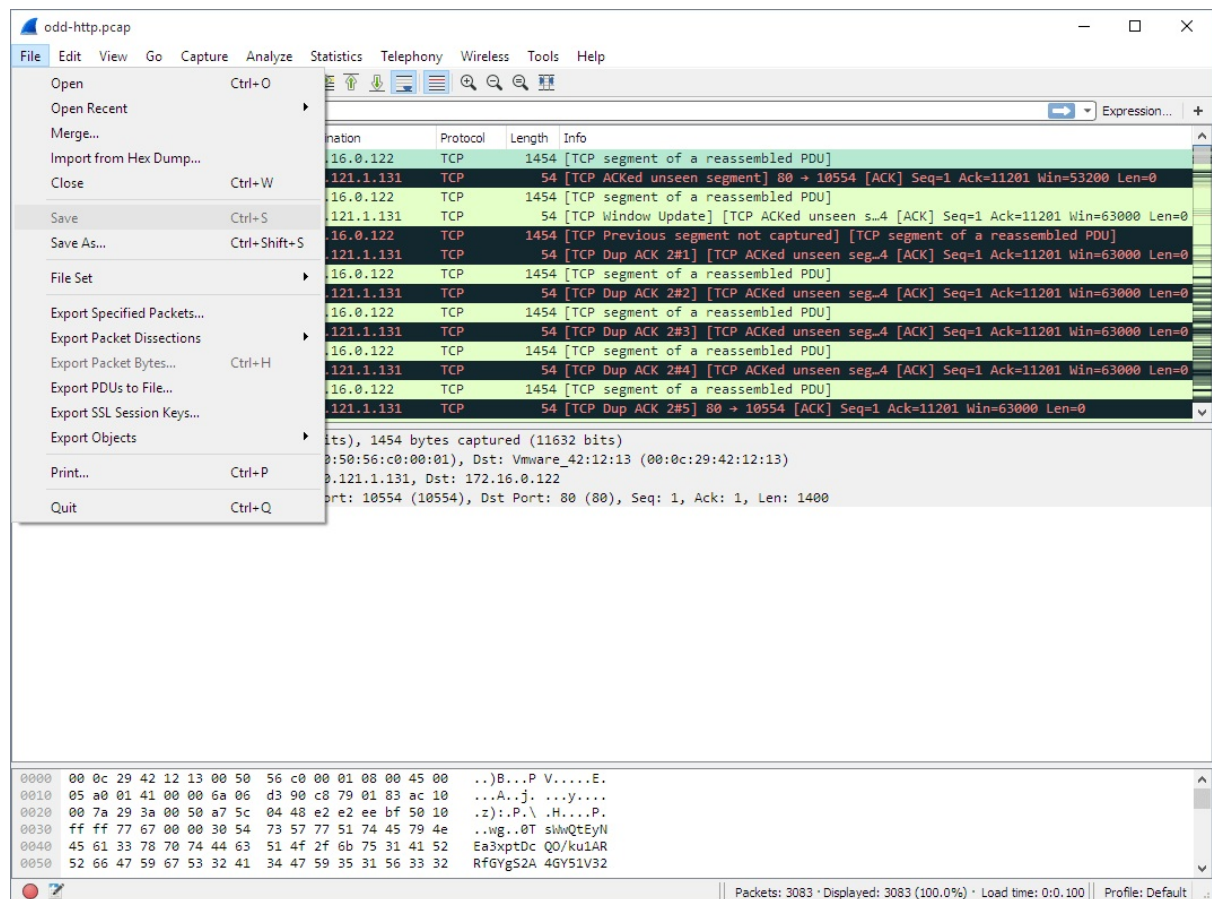


Table 3.2. File menu items

Menu Item	Accelerator	Description
Open...	Ctrl+O	This shows the file open dialog box that allows you to load a capture file for viewing. It is discussed in more detail in Section 5.2.1, “The “Open Capture File” dialog box” .
Open Recent		This lets you open recently opened capture files. Clicking on one of the submenu items will open the corresponding capture file directly.
Merge...		This menu item lets you merge a capture file into the currently loaded one. It is discussed in more detail in Section 5.4, “Merging capture files” .
Import from Hex Dump...		This menu item brings up the import file dialog box that allows you to import a text file containing a hex dump into a new temporary capture. It is discussed in more detail in Section 5.5, “Import hex dump” .
Close	Ctrl+W	This menu item closes the current capture. If you haven’t saved the capture, you will be asked to do so first (this can be disabled by a preference setting).
Save	Ctrl+S	This menu item saves the current capture. If you have not set a default capture file name (perhaps with the <code>-w \</code> option), Wireshark pops up the Save Capture File As dialog box (which is discussed further in Section 5.3.1, “The “Save Capture File As” dialog box”). If you have already saved the current capture, this menu item will be greyed out. You cannot save a live capture while the capture is in progress. You must stop the capture in order to save.
Save As...	Shift+Ctrl+S	This menu item allows you to save the current capture file to whatever file you would like. It pops up the Save Capture File As dialog box (which is discussed further in Section 5.3.1, “The “Save Capture File As” dialog box”).

File Set → List Files		This menu item allows you to show a list of files in a file set. It pops up the Wireshark List File Set dialog box (which is discussed further in Section 5.6, “File Sets”).
File Set → Next File		If the currently loaded file is part of a file set, jump to the next file in the set. If it isn't part of a file set or just the last file in that set, this item is greyed out.
File Set → Previous File		If the currently loaded file is part of a file set, jump to the previous file in the set. If it isn't part of a file set or just the first file in that set, this item is greyed out.
Export Specified Packets...		This menu item allows you to export all (or some) of the packets in the capture file to file. It pops up the Wireshark Export dialog box (which is discussed further in Section 5.7, “Exporting data”).
Export Packet Dissections...	Ctrl+H	These menu items allow you to export the currently selected bytes in the packet bytes pane to a text file in a number of formats including plain, CSV, and XML. It is discussed further in Section 5.7.7, “The “Export selected packet bytes” dialog box” .
Export Objects		These menu items allow you to export captured DICOM, HTTP, SMB, or TFTP objects into local files. It pops up a corresponding object list (which is discussed further in Section 5.7.8, “The “Export Objects” dialog box”).
Print...	Ctrl+P	This menu item allows you to print all (or some) of the packets in the capture file. It pops up the Wireshark Print dialog box (which is discussed further in Section 5.8, “Printing packets”).
Quit	Ctrl+Q	This menu item allows you to quit from Wireshark. Wireshark will ask to save your capture file if you haven't previously saved it (this can be disabled by a preference setting).

The “Edit” menu

The Wireshark Edit menu contains the fields shown in [Table 3.3, “Edit menu items”](#).

Figure 3.4. The “Edit” Menu

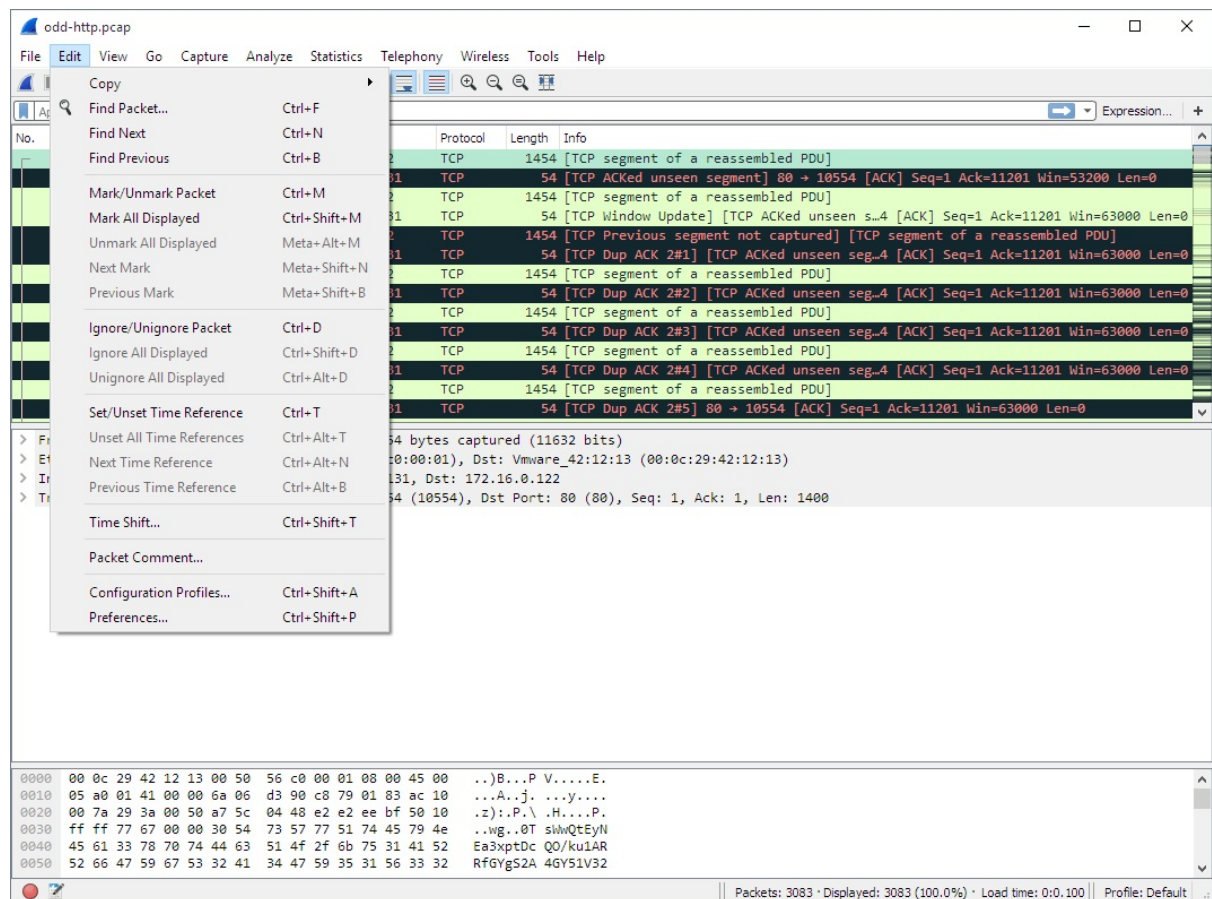


Table 3.3. Edit menu items

Menu Item	Accelerator	Description
Copy		These menu items will copy the packet list, packet detail, or properties of the currently selected packet to the clipboard.
Find Packet...	Ctrl+F	This menu item brings up a toolbar that allows you to find a packet by many criteria. There is further information on finding packets in Section 6.8, "Finding packets" .
Find Next	Ctrl+N	This menu item tries to find the next packet matching the settings from "Find Packet...".
Find Previous	Ctrl+B	This menu item tries to find the previous packet matching the settings from "Find Packet...".
Mark/Unmark Packet	Ctrl+M	This menu item marks the currently selected packet. for details.
Mark All Displayed Packets	Shift+Ctrl+M	This menu item marks all displayed packets.
Unmark All Displayed Packets	Ctrl+Alt+M	This menu item unmarks all displayed packets.
Next Mark	Shift+Alt+N	Find the next marked packet.
Previous Mark	Shift+Alt+B	Find the previous marked packet.
Ignore/Unignore Packet	Ctrl+D	This menu item marks the currently selected packet as ignored. for details.

Ignore All Displayed	Shift+Ctrl+D	This menu item marks all displayed packets as ignored.
Unignore All Displayed	Ctrl+Alt+D	This menu item unmarks all ignored packets.
Set/Unset Time Reference	Ctrl+T	This menu item set a time reference on the currently selected packet. for more information about the time referenced packets.
Unset All Time References	Ctrl+Alt+T	This menu item removes all time references on the packets.
Next Time Reference	Ctrl+Alt+N	This menu item tries to find the next time referenced packet.
Previous Time Reference	Ctrl+Alt+B	This menu item tries to find the previous time referenced packet.
Time Shift	Ctrl+Shift+T	This will show the Time Shift dialog, which allows you to adjust the timestamps of some or all packets.
Packet Comment...		This will let you add a comment to a single packet. Note that the ability to save packet comments depends on your file format. E.g. pcapng supports comments, pcap does not.
Capture Comment...		This will let you add a capture comment. Note that the ability to save capture comments depends on your file format. E.g. pcapng supports comments, pcap does not.
Configuration Profiles...	Shift+Ctrl+A	This menu item brings up a dialog box for handling configuration profiles. More detail is provided in Section 10.6, “Configuration Profiles” .
Preferences...	Shift+Ctrl+P or Cmd+** (macOS)	This menu item brings up a dialog box that allows you to set preferences for many parameters that control Wireshark. You can also save your preferences so Wireshark will use them the next time you start it. More detail is provided in Section 10.5, “Preferences” .

The “View” menu

The Wireshark View menu contains the fields shown in [Table 3.4, “View menu items”](#).

Figure 3.5. The “View” Menu

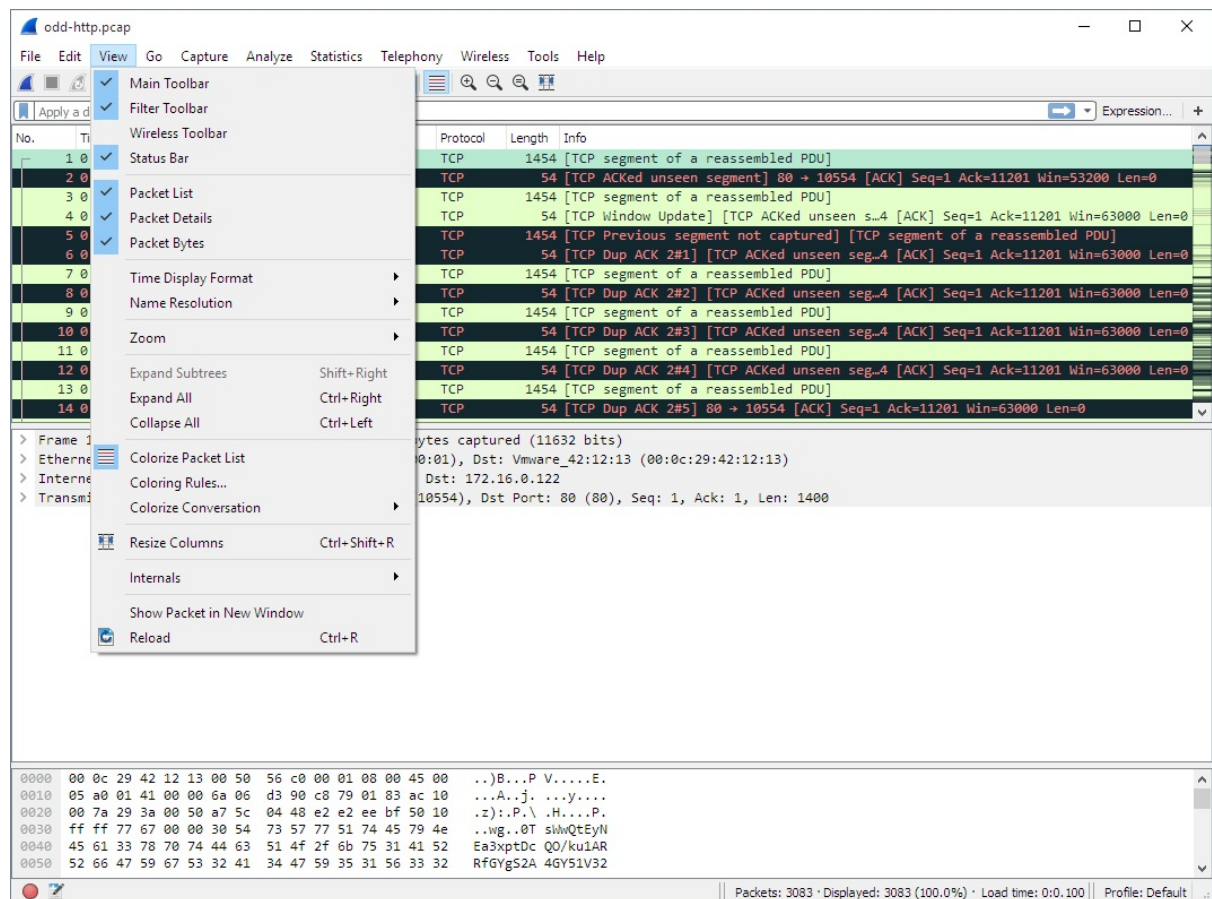


Table 3.4. View menu items

Menu Item	Accelerator	Description
Main Toolbar		This menu item hides or shows the main toolbar, .
Filter Toolbar		This menu item hides or shows the filter toolbar, .
Wireless Toolbar		This menu item hides or shows the wireless toolbar. May not be present on some platforms.
Statusbar		This menu item hides or shows the statusbar, .
Packet List		This menu item hides or shows the packet list pane, .
Packet Details		This menu item hides or shows the packet details pane, .
Packet Bytes		This menu item hides or shows the packet bytes pane, .
Time Display Format → Date and Time of Day: 1970-01-01 01:02:03.123456		Selecting this tells Wireshark to display the time stamps in date and time of day format, . The fields "Time of Day", "Date and Time of Day", "Seconds Since Beginning of Capture", "Seconds Since Previous Captured Packet" and "Seconds Since Previous Displayed Packet" are mutually exclusive.
Time Display Format → Time of Day: 01:02:03.123456		Selecting this tells Wireshark to display time stamps in time of day format, .
Time Display Format → Seconds Since Epoch (1970-01-01): 1234567890.123456		Selecting this tells Wireshark to display time stamps in seconds since 1970-01-01 00:00:00, .
Time Display Format		

→ Seconds Since Beginning of Capture: 123.123456		Selecting this tells Wireshark to display time stamps in seconds since beginning of capture format, .
Time Display Format → Seconds Since Previous Captured Packet: 1.123456		Selecting this tells Wireshark to display time stamps in seconds since previous captured packet format, .
Time Display Format → Seconds Since Previous Displayed Packet: 1.123456		Selecting this tells Wireshark to display time stamps in seconds since previous displayed packet format, .
Time Display Format → Automatic (File Format Precision)		Selecting this tells Wireshark to display time stamps with the precision given by the capture file format used, . The fields "Automatic", "Seconds" and "...seconds" are mutually exclusive.
Time Display Format → Seconds: 0		Selecting this tells Wireshark to display time stamps with a precision of one second, .
Time Display Format → ...seconds: 0....		Selecting this tells Wireshark to display time stamps with a precision of one second, decisecond, centisecond, millisecond, microsecond or nanosecond, .
Time Display Format → Display Seconds with hours and minutes		Selecting this tells Wireshark to display time stamps in seconds, with hours and minutes.
Name Resolution → Resolve Name		This item allows you to trigger a name resolve of the current packet only, .
Name Resolution → Enable for MAC Layer		This item allows you to control whether or not Wireshark translates MAC addresses into names, .
Name Resolution → Enable for Network Layer		This item allows you to control whether or not Wireshark translates network addresses into names, .
Name Resolution → Enable for Transport Layer		This item allows you to control whether or not Wireshark translates transport addresses into names, .
Colorize Packet List		This item allows you to control whether or not Wireshark should colorize the packet list. Enabling colorization will slow down the display of new packets while capturing / loading capture files.
Auto Scroll in Live Capture		This item allows you to specify that Wireshark should scroll the packet list pane as new packets come in, so you are always looking at the last packet. If you do not specify this, Wireshark simply adds new packets onto the end of the list, but does not scroll the packet list pane.
Zoom In	Ctrl++	Zoom into the packet data (increase the font size).
Zoom Out	Ctrl+-	Zoom out of the packet data (decrease the font size).
Normal Size	Ctrl+=	Set zoom level back to 100% (set font size back to normal).
Resize All Columns	Shift+Ctrl+R	Resize all column widths so the content will fit into it. Resizing may take a significant amount of time, especially if a large capture file is loaded.
Displayed Columns		This menu items folds out with a list of all configured columns. These columns can now be shown or hidden in the packet list.
		This menu item expands the currently selected subtree in the packet

		details tree.
Collapse Subtrees	Shift+←	This menu item collapses the currently selected subtree in the packet details tree.
Expand All	Ctrl+→	Wireshark keeps a list of all the protocol subtrees that are expanded, and uses it to ensure that the correct subtrees are expanded when you display a packet. This menu item expands all subtrees in all packets in the capture.
Collapse All	Ctrl+←	This menu item collapses the tree view of all packets in the capture list.
Colorize Conversation		This menu item brings up a submenu that allows you to color packets in the packet list pane based on the addresses of the currently selected packet. This makes it easy to distinguish packets belonging to different conversations. Section 10.3, “Packet coloring” .
Colorize Conversation → Color 1-10		These menu items enable one of the ten temporary color filters based on the currently selected conversation.
Colorize Conversation → Reset coloring		This menu item clears all temporary coloring rules.
Colorize Conversation → New Coloring Rule...		This menu item opens a dialog window in which a new permanent coloring rule can be created based on the currently selected conversation.
Coloring Rules...		This menu item brings up a dialog box that allows you to color packets in the packet list pane according to filter expressions you choose. It can be very useful for spotting certain types of packets, .
Show Packet in New Window		This menu item brings up the selected packet in a separate window. The separate window shows only the tree view and byte view panes.
Reload	Ctrl+R	This menu item allows you to reload the current capture file.

The “Go” menu

The Wireshark Go menu contains the fields shown in [Table 3.5, “Go menu items”](#).

Figure 3.6. The “Go” Menu

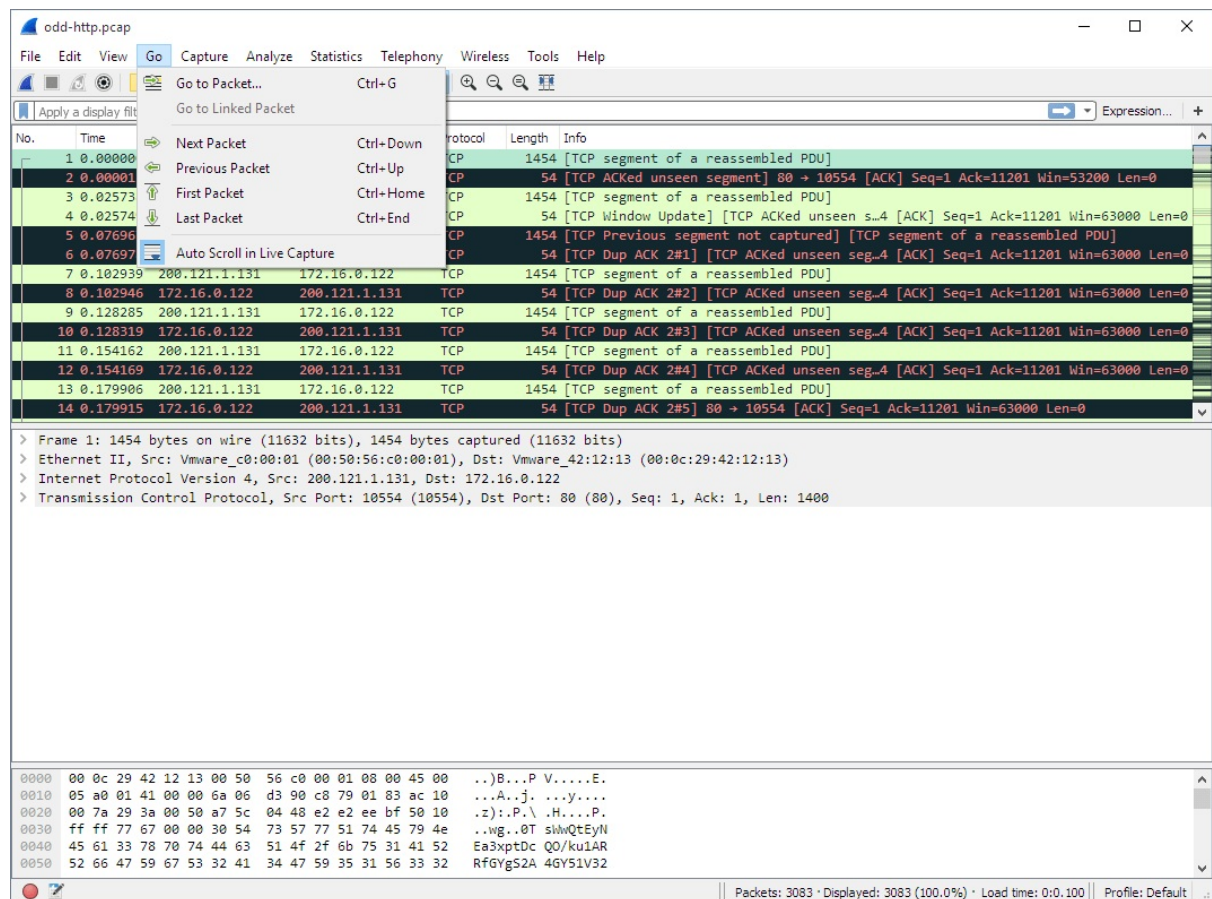


Table 3.5. Go menu items

Menu Item	Accelerator	Description
Back	Alt+←	Jump to the recently visited packet in the packet history, much like the page history in a web browser.
Forward	Alt+→	Jump to the next visited packet in the packet history, much like the page history in a web browser.
Go to Packet...	Ctrl+G	Bring up a window frame that allows you to specify a packet number, and then goes to that packet. for details.
Go to Corresponding Packet		Go to the corresponding packet of the currently selected protocol field. If the selected field doesn't correspond to a packet, this item is greyed out.
Previous Packet	Ctrl+↑	Move to the previous packet in the list. This can be used to move to the previous packet even if the packet list doesn't have keyboard focus.
Next Packet	Ctrl+↓	Move to the next packet in the list. This can be used to move to the previous packet even if the packet list doesn't have keyboard focus.
First Packet	Ctrl+Home	Jump to the first packet of the capture file.
Last Packet	Ctrl+End	Jump to the last packet of the capture file.
Previous Packet In Conversation	Ctrl+,	Move to the previous packet in the current conversation. This can be used to move to the previous packet even if the packet list doesn't have keyboard focus.
Next Packet In Conversation	Ctrl+.	Move to the next packet in the current conversation. This can be used to move to the previous packet even if the packet list doesn't have keyboard focus.

The “Capture” menu

The Wireshark Capture menu contains the fields shown in [Table 3.6, “Capture menu items”](#).

Figure 3.7. The “Capture” Menu

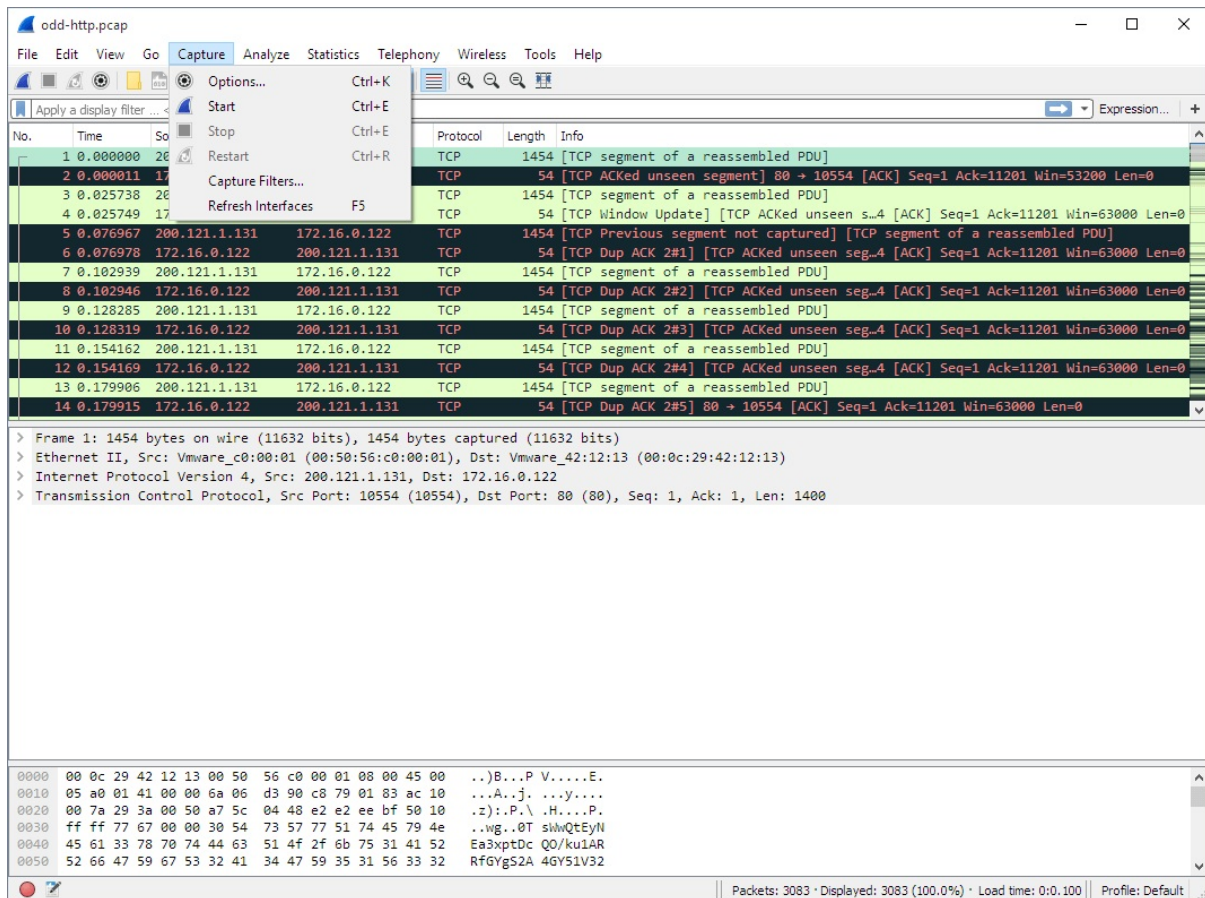


Table 3.6. Capture menu items

Menu Item	Accelerator	Description
Interfaces...	Ctrl+I	This menu item brings up a dialog box that shows what's going on at the network interfaces Wireshark knows of,).
Options...	Ctrl+K	This menu item brings up the Capture Options dialog box (discussed further in Section 4.5, “The “Capture Options” dialog box”) and allows you to start capturing packets.
Start	Ctrl+E	Immediately start capturing packets with the same settings than the last time.
Stop	Ctrl+E	This menu item stops the currently running capture,).
Restart	Ctrl+R	This menu item stops the currently running capture and starts again with the same options, this is just for convenience.
Capture Filters...		This menu item brings up a dialog box that allows you to create and edit capture filters. You can name filters, and you can save them for future use. More detail on this subject is provided in Section 6.6, “Defining and saving filters”

The “Analyze” menu

The Wireshark Analyze menu contains the fields shown in [Table 3.7, “Analyze menu items”](#).

Figure 3.8. The “Analyze” Menu

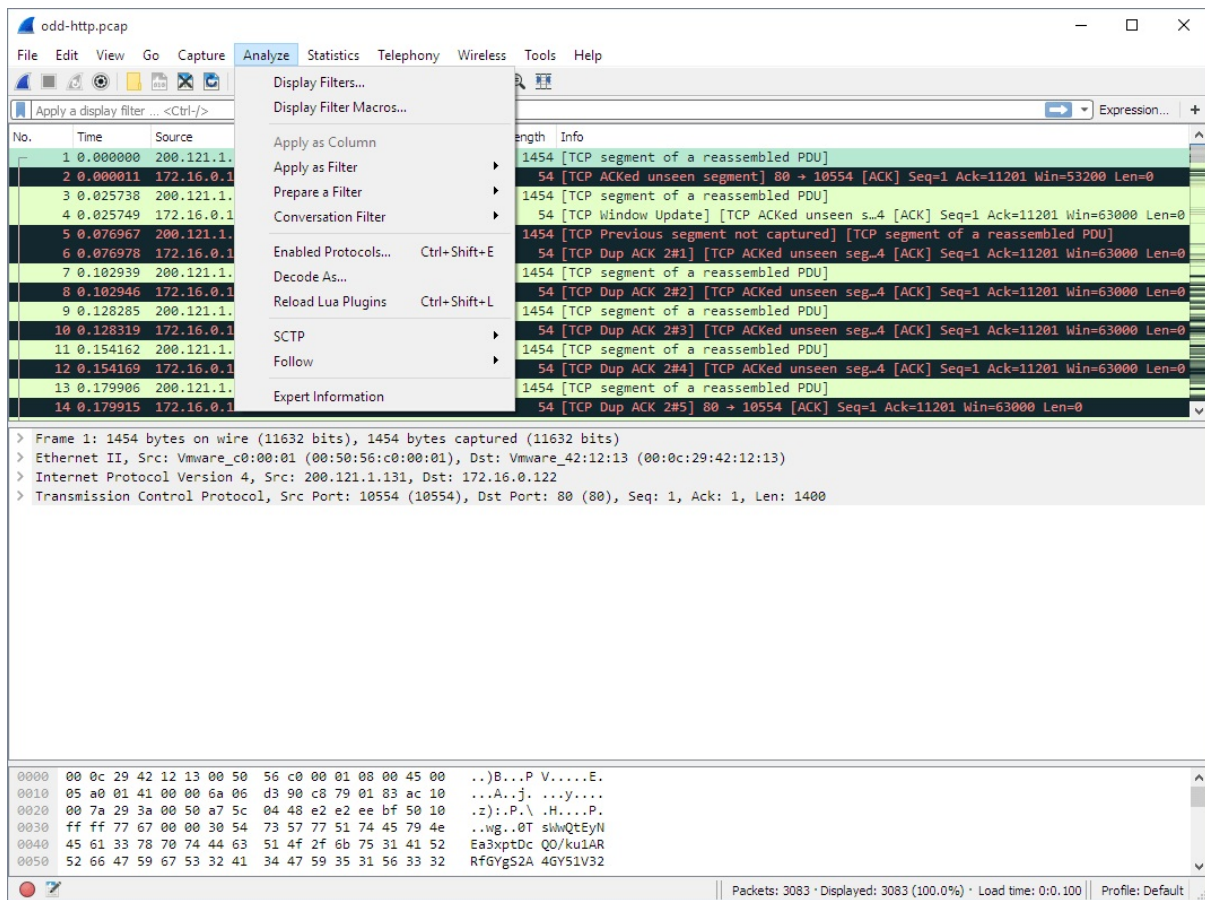


Table 3.7. Analyze menu items

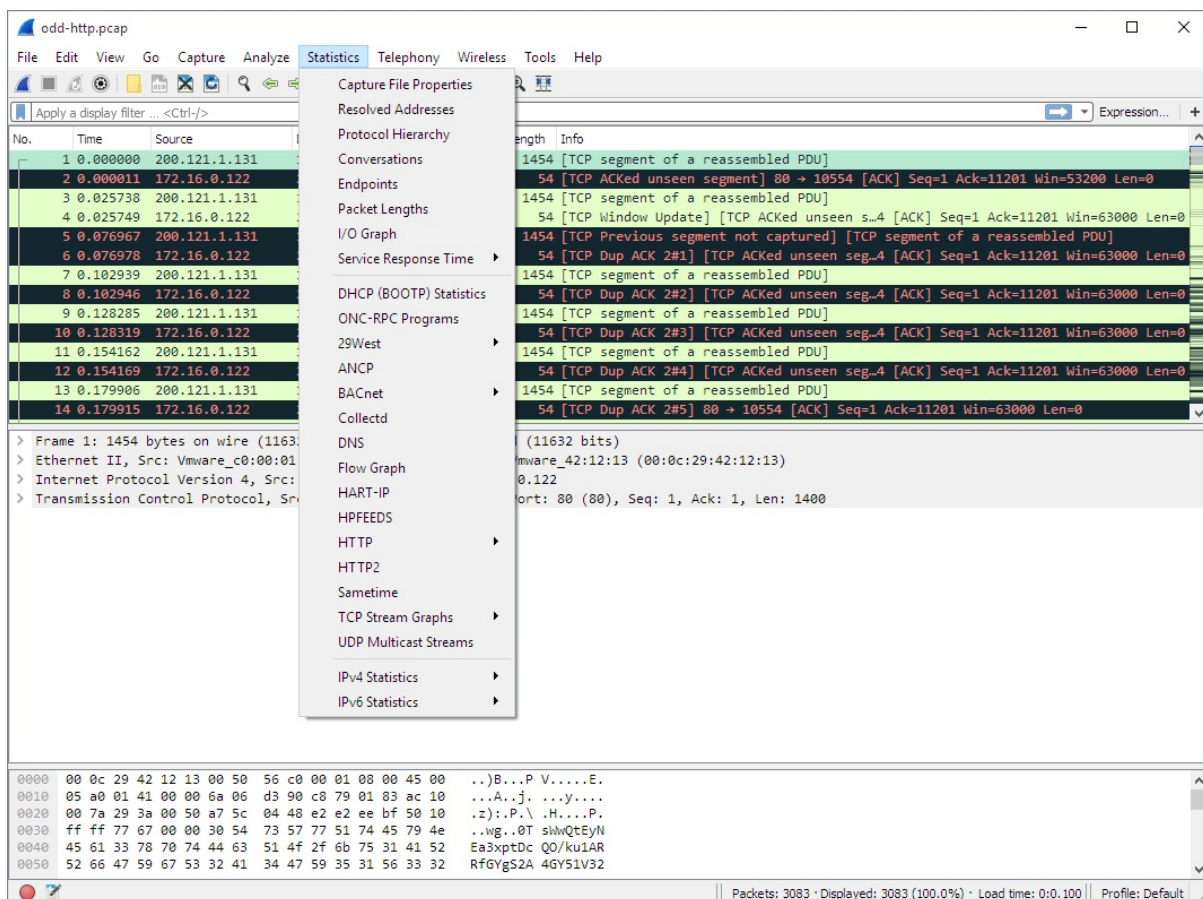
Menu Item	Accelerator	Description
Display Filters...		This menu item brings up a dialog box that allows you to create and edit display filters. You can name filters, and you can save them for future use. More detail on this subject is provided in Section 6.6, “Defining and saving filters”
Display Filter Macros...		This menu item brings up a dialog box that allows you to create and edit display filter macros. You can name filter macros, and you can save them for future use. More detail on this subject is provided in Section 6.7, “Defining and saving filter macros”
Apply as Column		This menu item adds the selected protocol item in the packet details pane as a column to the packet list.
Apply as Filter → ...		These menu items will change the current display filter and apply the changed filter immediately. Depending on the chosen menu item, the current display filter string will be replaced or appended to by the selected protocol field in the packet details pane.
Prepare a Filter → ...		These menu items will change the current display filter but won't apply the changed filter. Depending on the chosen menu item, the current display filter string will be replaced or appended to by the selected protocol field in the packet details pane.
Enabled Protocols...	Shift+Ctrl+E	This menu item allows the user to enable/disable protocol dissectors,
Decode As...		This menu item allows the user to force Wireshark to decode certain packets as a particular protocol,

User Specified Decodes...		This menu item allows the user to force Wireshark to decode certain packets as a particular protocol,
Follow TCP Stream		This menu item brings up a separate window and displays all the TCP segments captured that are on the same TCP connection as a selected packet,
Follow UDP Stream		Same functionality as “Follow TCP Stream” but for UDP streams.
Follow SSL Stream		Same functionality as “Follow TCP Stream” but for SSL streams. See the wiki page on SSL for instructions on providing SSL keys.
Expert Info		Open a dialog showing some expert information about the captured packets. The amount of information will depend on the protocol and varies from very detailed to non-existent. XXX - add a new section about this and link from here
Conversation Filter → ...		In this menu you will find conversation filter for various protocols.

The “Statistics” menu

The Wireshark Statistics menu contains the fields shown in [Table 3.8, “Statistics menu items”](#).

Figure 3.9. The “Statistics” Menu



All menu items will bring up a new window showing specific statistical information.

Table 3.8. Statistics menu items

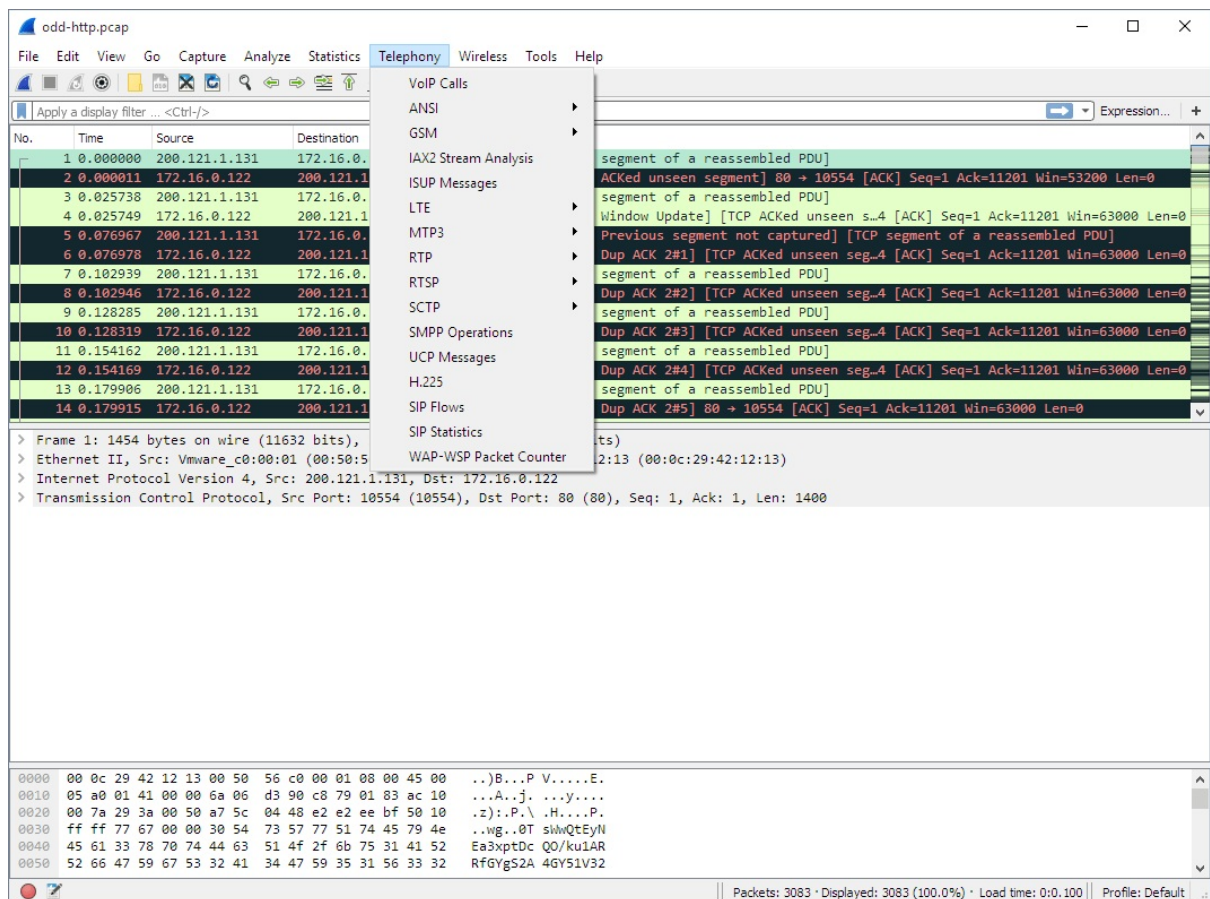
Menu Item	Accelerator	Description

Summary		Show information about the data captured, .
Protocol Hierarchy		Display a hierarchical tree of protocol statistics, .
Conversations		Display a list of conversations (traffic between two endpoints), .
Endpoints		Display a list of endpoints (traffic to/from an address), .
Packet Lengths...		
IO Graphs		Display user specified graphs (e.g. the number of packets in the course of time), .
Service Response Time		Display the time between a request and the corresponding response, .
ANCP		
Collected...		
Compare...		
Flow Graph...		
HTTP		HTTP request/response statistics,
IP Addresses...		
IP Destinations...		
IP Protocol Types...		
ONC-RPC Programs		
Sametime		
TCP Stream Graph		
UDP Multicast Streams		
WLAN Traffic		
BOOTP-DHCP		

The “Telephony” menu

The Wireshark Telephony menu contains the fields shown in [Table 3.9, “Telephony menu items”](#).

Figure 3.10. The “Telephony” Menu



All menu items will bring up a new window showing specific telephony related statistical information.

Table 3.9. Telephony menu items

Menu Item	Accelerator	Description
IAX2		
SMPP Operations...		
SCTP		
ANSI		
GSM		
H.225...		
ISUP Messages...		
LTE		
MTP3		
RTP		
SIP...		
UCP Messages...		
VoIP Calls...		
WAP-WSP...		

The “Tools” menu

The Wireshark Tools menu contains the fields shown in [Table 3.10, “Tools menu items”](#).

Figure 3.11. The “Tools” Menu

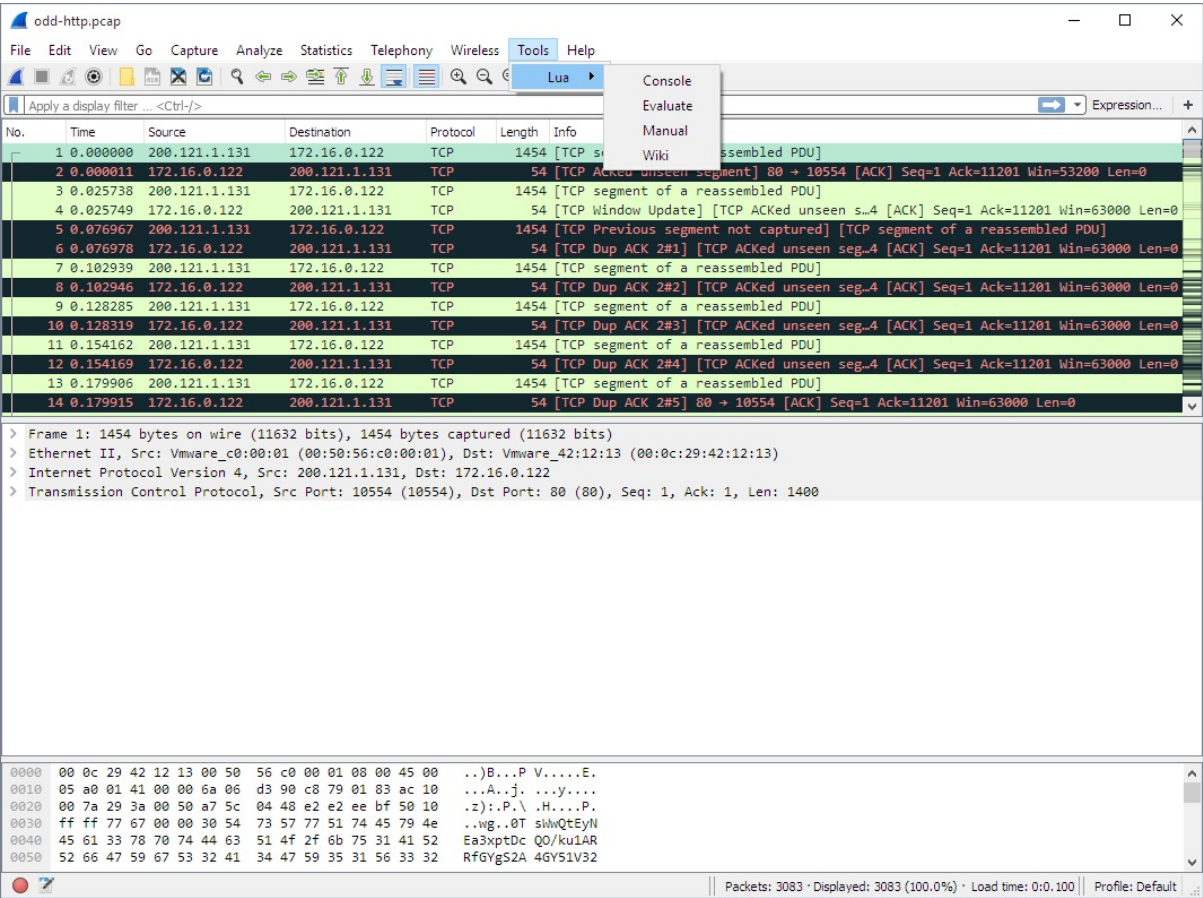


Table 3.10. Tools menu items

Menu Item	Accelerator	Description
Firewall ACL Rules		This allows you to create command-line ACL rules for many different firewall products, including Cisco IOS, Linux Netfilter (iptables), OpenBSD pf and Windows Firewall (via netsh). Rules for MAC addresses, IPv4 addresses, TCP and UDP ports, and IPv4+port combinations are supported. It is assumed that the rules will be applied to an outside interface.
Lua		These options allow you to work with the Lua interpreter optionally build into Wireshark. See the “Lua Support in Wireshark” in the Wireshark Developer’s Guide.

The “Internals” menu

The Wireshark Internals menu contains the fields shown in [Table 3.11, “Internals menu items”](#).

Figure 3.12. The “Internals” Menu

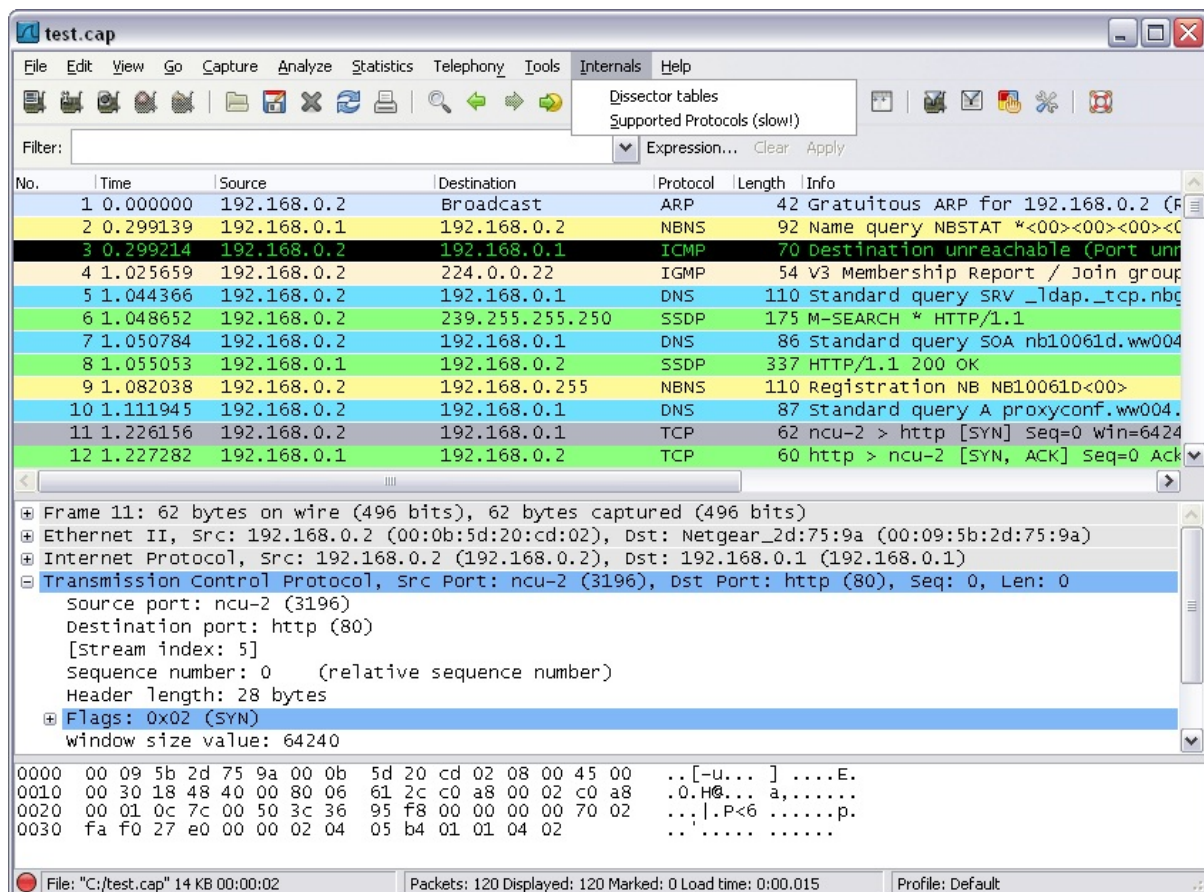


Table 3.11. Internals menu items

Menu Item	Accelerator	Description
Dissector tables		This menu item brings up a dialog box showing the tables with subdissector relationships.
Supported Protocols (slow!)		This menu item brings up a dialog box showing the supported protocols and protocol fields.

The “Help” menu

The Wireshark Help menu contains the fields shown in Table 3.12, “Help menu items”.

Figure 3.13. The “Help” Menu

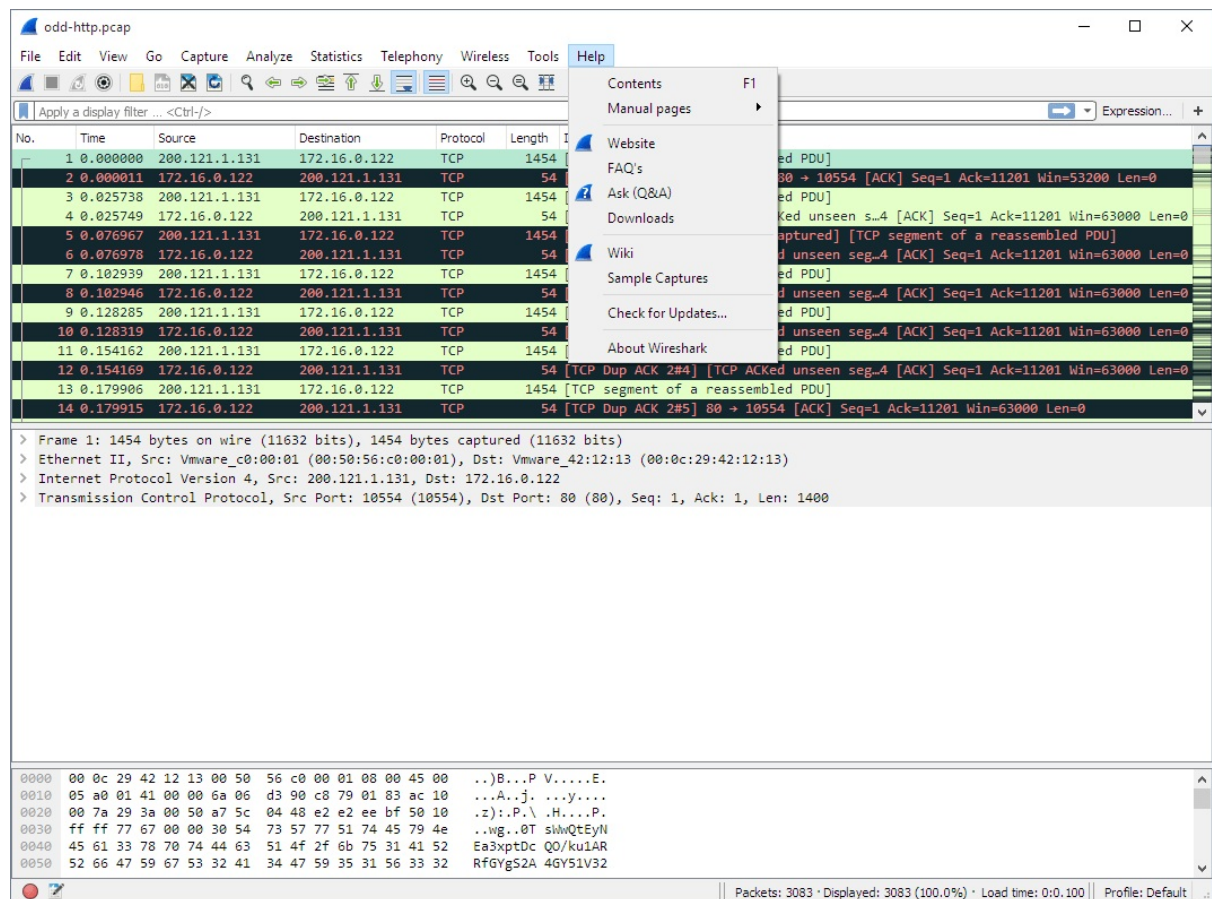


Table 3.12. Help menu items

Menu Item	Accelerator	Description
Contents	F1	This menu item brings up a basic help system.
Manual Pages → ...		This menu item starts a Web browser showing one of the locally installed html manual pages.
Website		This menu item starts a Web browser showing the webpage from: https://www.wireshark.org/ .
FAQ's		This menu item starts a Web browser showing various FAQ's.
Downloads		This menu item starts a Web browser showing the downloads from: https://www.wireshark.org/download.html .
Wiki		This menu item starts a Web browser showing the front page from: https://wiki.wireshark.org/ .
Sample Captures		This menu item starts a Web browser showing the sample captures from: https://wiki.wireshark.org/SampleCaptures .
About Wireshark		This menu item brings up an information window that provides various detailed information items on Wireshark, such as how it's build, the plugins loaded, the used folders, ...

Opening a Web browser might be unsupported in your version of Wireshark. If this is the case the corresponding menu items will be hidden.

If calling a Web browser fails on your machine, nothing happens, or the browser starts but no page is shown, have a look at the web browser setting in the preferences dialog.

4. Toolbars

The “Main” toolbar

The main toolbar provides quick access to frequently used items from the menu. This toolbar cannot be customized by the user, but it can be hidden using the View menu, if the space on the screen is needed to show even more packet data.

As in the menu, only the items useful in the current program state will be available. The others will be greyed out (e.g. you cannot save a capture file if you haven't loaded one).

Figure 3.14. The “Main” toolbar



Table 3.13. Main toolbar items

Toolbar Icon	Toolbar Item	Menu Item	Description
 Interfaces ...	Capture → Interfaces...	This item brings up the Capture Interfaces List dialog box (discussed further in Section 4.3, “Start Capturing”).	
 Options ...	Capture → Options...	This item brings up the Capture Options dialog box (discussed further in Section 4.3, “Start Capturing”) and allows you to start capturing packets.	
 Start	Capture → Start	This item starts capturing packets with the options from the last time.	
 Stop	Capture → Stop	This item stops the currently running live capture process Section 4.3, “Start Capturing”).	
 Restart	Capture → Restart	This item stops the currently running live capture process and restarts it again, for convenience.	
 Open...	File → Open...	This item brings up the file open dialog box that allows you to load a capture file for viewing. It is discussed in more detail in Section 5.2.1, “The “Open Capture File” dialog box” .	
 Save As...	File → Save As...	This item allows you to save the current capture file to whatever file you would like. It pops up the Save Capture File As dialog box (which is discussed further in Section 5.3.1, “The “Save Capture File As” dialog box”). If you currently have a temporary capture file, the Save icon will be shown instead.	
 Close	File → Close	This item closes the current capture. If you have not saved the capture, you will be asked to save it first.	
 Reload	View → Reload	This item allows you to reload the current capture file.	
 Print...	File → Print...	This item allows you to print all (or some of) the packets in the capture file. It pops up the Wireshark Print dialog box (which is discussed further in Section 5.8, “Printing packets”).	
 Find	Edit → Find	This item brings up a dialog box that allows you to find a packet. There is further information on finding packets in	

Packet...		Section 6.8, “Finding packets” .
 Go Back	Go → Go Back	This item jumps back in the packet history. Hold down the Alt key (Option on macOS) to go back in the selection history.
 Go Forward	Go → Go Forward	This item jumps forward in the packet history. Hold down the Alt key (Option on macOS) to go forward in the selection history.
 Go to Packet...	Go → Go to Packet...	This item brings up a dialog box that allows you to specify a packet number to go to that packet.
 Go To First Packet	Go → First Packet	This item jumps to the first packet of the capture file.
 Go To Last Packet	Go → Last Packet	This item jumps to the last packet of the capture file.
 Colorize	View → Colorize	Colorize the packet list (or not).
 Auto Scroll in Live Capture	View → Auto Scroll in Live Capture	Auto scroll packet list while doing a live capture (or not).
 Zoom In	View → Zoom In	Zoom into the packet data (increase the font size).
 Zoom Out	View → Zoom Out	Zoom out of the packet data (decrease the font size).
 Normal Size	View → Normal Size	Set zoom level back to 100%.
 Resize Columns	View → Resize Columns	Resize columns, so the content fits into them.
 Capture Filters...	Capture → Capture Filters...	This item brings up a dialog box that allows you to create and edit capture filters. You can name filters, and you can save them for future use. More detail on this subject is provided in Section 6.6, “Defining and saving filters” .
 Display Filters...	Analyze → Display Filters...	This item brings up a dialog box that allows you to create and edit display filters. You can name filters, and you can save them for future use. More detail on this subject is provided in Section 6.6, “Defining and saving filters” .
 Coloring Rules...	View → Coloring Rules...	This item brings up a dialog box that allows you to color packets in the packet list pane according to filter expressions you choose. It can be very useful for spotting certain types of packets. More detail on this subject is provided in Section 10.3, “Packet colorization” .
 Preferences...	Edit → Preferences	This item brings up a dialog box that allows you to set preferences for many parameters that control Wireshark. You can also save your preferences so Wireshark will use them the next time you start it. More detail is provided in Section 10.5, “Preferences”
 Help	Help → Contents	This item brings up help dialog box.

The “Filter” toolbar

The filter toolbar lets you quickly edit and apply display filters. More information on display filters is available in [Section 6.3, “Filtering packets while viewing”](#).

Figure 3.15. The “Filter” toolbar



Table 3.14. Filter toolbar items

Toolbar Icon	Toolbar Item	Description
 Filter:	Brings up the filter construction dialog, described in Figure 6.8, “The “Capture Filters” and “Display Filters” dialog boxes” .	
	<i>Filter input</i>	The area to enter or edit a display filter string, . A syntax check of your filter string is done while you are typing. The background will turn red if you enter an incomplete or invalid string, and will become green when you enter a valid string. You can click on the pull down arrow to select a previously-entered filter string from a list. The entries in the pull down list will remain available even after a program restart. After you've changed something in this field, don't forget to press the Apply button (or the Enter/Return key), to apply this filter string to the display. This field is also where the current filter in effect is displayed.
 Expressi on...	The middle button labeled "Add Expression..." opens a dialog box that lets you edit a display filter from a list of protocol fields, described in Section 6.5, “The “Filter Expression” dialog box”	
 Clear	Reset the current display filter and clears the edit area.	
 Apply	Apply the current value in the edit area as the new display filter. Applying a display filter on large capture files might take quite a long time.	

5. Panes

The “Packet List” pane

The packet list pane displays all the packets in the current capture file.

Figure 3.16. The “Packet List” pane



Each line in the packet list corresponds to one packet in the capture file. If you select a line in this pane, more details will be displayed in the “Packet Details” and “Packet Bytes” panes.

While dissecting a packet, Wireshark will place information from the protocol dissectors into the columns. As higher level protocols might overwrite information from lower levels, you will typically see the information from the highest possible level only.

For example, let’s look at a packet containing TCP inside IP inside an Ethernet packet. The Ethernet dissector will write its data (such as the Ethernet addresses), the IP dissector will overwrite this by its own (such as the IP addresses), the TCP dissector will overwrite the IP information, and so on.

There are a lot of different columns available. Which columns are displayed can be selected by preference settings, .

The default columns will show:

- No. The number of the packet in the capture file. This number won’t change, even if a display filter is used.
- Time The timestamp of the packet. The presentation format of this timestamp can be changed, .
- Source The address where this packet is coming from.
- Destination The address where this packet is going to.
- Protocol The protocol name in a short (perhaps abbreviated) version.
- Length The length of each packet.
- Info Additional information about the packet content.

The first column shows how each packet is related to the selected packet. For example, in the image above the first packet is selected, which is a DNS request. Wireshark shows a rightward arrow for the request itself, followed by a leftward arrow for the response in packet 2. Why is there a dashed line? There are more DNS packets further down that use the same port numbers. Wireshark treats them as belonging to the same conversation and draws a line connecting them.

Table 3.15. Related packet symbols

- | | |
|---|--|
|  | • First packet in a conversation. |
|  | • Part of the selected conversation. |
|  | • <i>Not</i> part of the selected conversation. |
|  | • Last packet in a conversation. |
|  | • Request. |
|  | • Response. |
|  | • The selected packet acknowledges this packet. |
|  | • The selected packet is a duplicate acknowledgement of this packet. |

-  The selected packet is related to this packet in some other way, e.g. as part of reassembly.

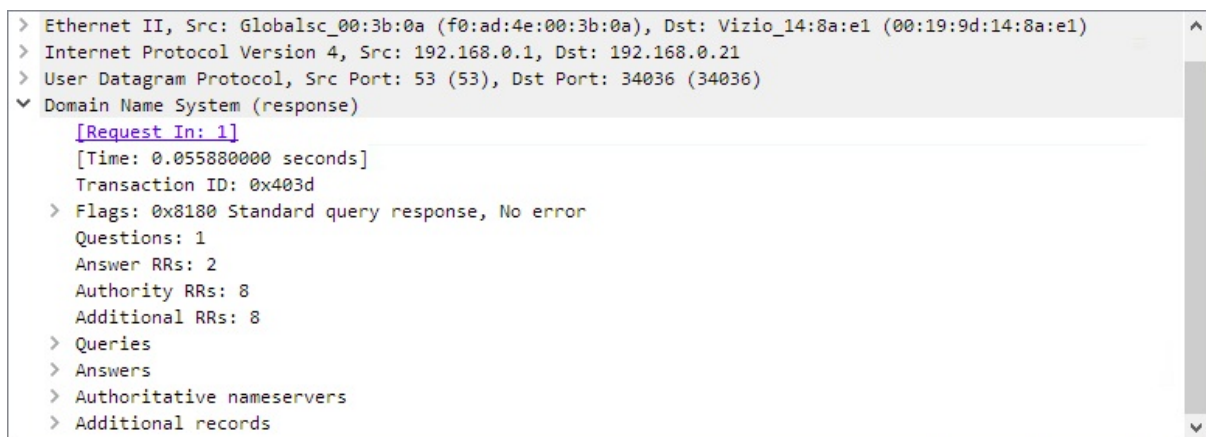
The packet list has an *Intelligent Scrollbar* which shows a miniature map of nearby packets. Each [raster line](#) of the scrollbar corresponds to a single packet, so the number of packets shown in the map depends on your physical display and the height of the packet list. A tall packet list on a high-resolution (“Retina”) display will show you quite a few packets. In the image above the scrollbar shows the status of more than 500 packets along with the 15 shown in the packet list itself.

Right clicking will show a context menu, described in [Figure 6.4, “Pop-up menu of the “Packet List” pane”](#).

The “Packet Details” pane

The packet details pane shows the current packet (selected in the “Packet List” pane) in a more detailed form.

Figure 3.17. The “Packet Details” pane



This pane shows the protocols and protocol fields of the packet selected in the “Packet List” pane. The protocols and fields of the packet shown in a tree which can be expanded and collapsed.

There is a context menu (right mouse click) available.

Some protocol fields have special meanings.

- **Generated fields.** Wireshark itself will generate additional protocol information which isn’t present in the captured data. This information is enclosed in square brackets (‘[’ and ‘]’). Generated information includes response times, TCP analysis, GeoIP information, and checksum validation.
- **Links.** If Wireshark detects a relationship to another packet in the capture file it will generate a link to that packet. Links are underlined and displayed in blue. If you double-clicked on a link Wireshark will jump to the corresponding packet.

The “Packet Bytes” pane

The packet bytes pane shows the data of the current packet (selected in the “Packet List” pane) in a hexdump style.

Figure 3.18. The “Packet Bytes” pane

0000	00 19 9d 14 8a e1 f0 ad 4e 00 3b 0a 08 00 45 00	 N.;...E.
0010	01 d1 00 00 40 00 40 11 b7 b5 c0 a8 00 01 c0 a8	...@. @.
0020	00 15 00 35 84 f4 01 bd 83 35 40 3d 81 80 00 01	...5... .5@=...
0030	00 02 00 08 00 08 0c 6d 6f 76 69 65 63 6f 6e 74	m oviecont
0040	72 6f 6c 07 6e 65 74 66 6c 69 78 03 63 6f 6d 00	rol.netf lix.com.
0050	00 01 00 01 c0 0c 00 05 00 01 00 00 00 2d 00 40	-.@
0060	25 6e 63 63 70 2d 6d 6f 76 69 65 63 6f 6e 74 72	%nccp-mo viecontr
0070	6f 6c 2d 66 72 6f 6e 74 65 6e 64 2d 31 37 31 32	ol-front end-1712
0080	31 38 38 39 32 31 09 75 73 2d 65 61 73 74 2d 31	188921.u s-east-1
0090	03 65 6c 62 09 61 6d 61 7a 6f 6e 61 77 73 c0 21	.elb.ama zonaws.!

The “Packet Bytes” pane shows a canonical [hex dump](#) of the packet data. Each line contains the data offset, sixteen hexadecimal bytes, and sixteen ASCII bytes. Non-printable bytes are replaced with a period ('.').

Depending on the packet data, sometimes more than one page is available, e.g. when Wireshark has reassembled some packets into a single chunk of data. (for details). In this case you can see each data source by clicking its corresponding tab at the bottom of the pane.

Figure 3.19. The “Packet Bytes” pane with tabs

0000	00 19 9d 14 8a e1 f0 ad 4e 00 3b 0a 08 00 45 00	 N.;...E.
0010	01 4f 0b 04 40 00 2e 06 54 c0 32 11 f9 16 c0 a8	.O..@... T.2.....
0020	00 15 01 bb 91 c4 14 dd 57 0b a4 03 62 21 80 18	 W...b!..
0030	02 d4 0e 37 00 00 01 01 08 0a 7d 58 40 bc 1d 4b	...7.... ..}X@..K
0040	3b 0a 06 09 2a 86 48 86 f7 0d 01 01 05 05 00 03	;...*.H.
0050	82 01 01 00 71 49 a0 e4 9e 26 d0 d8 00 4b a1 b9	qI.. .&...K..
0060	5c 37 7e 99 5a 70 cb db ab b7 c7 80 6c 8b 75 c1	\7~.Zp..l.u.
0070	84 77 3c 47 29 f9 e0 f0 d6 4e 61 16 34 1b 4f 75	.w<G)... .Na.4.Ou
0080	c6 5e 64 02 01 65 4d a0 21 8f 7f 8b fd dc 53 85	^d...eM. !.....S.

Frame (349 bytes) Reassembled TCP (3091 bytes)

Additional pages typically contain data reassembled from multiple packets or decrypted data.

The context menu (right mouse click) of the tab labels will show a list of all available pages. This can be helpful if the size in the pane is too small for all the tab labels.

6. The Statusbar

The statusbar displays informational messages.

In general, the left side will show context related information, the middle part will show information about the current capture file, and the right side will show the selected configuration profile. Drag the handles between the text areas to change the size.

Figure 3.20. The initial Statusbar

 Ready to load or capture	No Packets	Profile: Default
--	------------	------------------

This statusbar is shown while no capture file is loaded, e.g. when Wireshark is started.

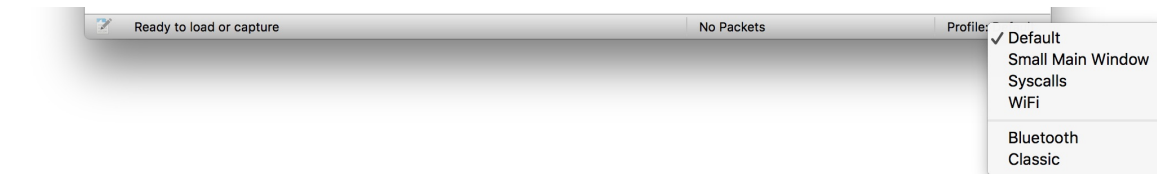
Figure 3.21. The Statusbar with a loaded capture file

 6to4-reqs	Packets: 500 · Displayed: 500 (100.0%) · Load time: 0:0.21	Profile: Default
---	--	------------------

- **The colored bullet** on the left shows the highest expert info level found in the currently loaded capture file. Hovering the mouse over this icon will show a textual description of the expert info level, and clicking the icon will bring up the Expert Infos dialog box. For a detailed description of expert info, .
- **The left side** shows information about the capture file, its name, its size and the elapsed time while it was being captured. Hovering over a file name will show its full path and size.
- **The middle part** shows the current number of packets in the capture file. The following values are displayed:
 - *Packets*: The number of captured packets.
 - *Displayed*: The number of packets currently being displayed.

- *Marked*: The number of marked packets (only displayed if packets are marked).
- *Dropped*: The number of dropped packets (only displayed if Wireshark was unable to capture all packets).
- *Ignored*: The number of ignored packets (only displayed if packets are ignored).
- *Load time*: The time it took to load the capture (wall clock time).
- **The right side** shows the selected configuration profile. Clicking in this part of the statusbar will bring up a menu with all available configuration profiles, and selecting from this list will change the configuration profile.

Figure 3.22. The Statusbar with a configuration profile menu



For a detailed description of configuration profiles, .

Figure 3.23. The Statusbar with a selected protocol field



This is displayed if you have selected a protocol field from the “Packet Details” pane.

The value between the parentheses (in this example ‘ipv6.src’) can be used as a display filter, representing the selected protocol field.

Figure 3.24. The Statusbar with a display filter message



This is displayed if you are trying to use a display filter which may have unexpected results. For a detailed description, .

Sources

- https://www.wireshark.org/docs/wsug_html/#ChapterUsing

Capture de paquets

- [1. Introduction](#)
- [2. Prerequisites](#)
- [3. Start Capturing](#)
- [4. The “Capture Interfaces” dialog box](#)
- [5. The “Capture Options” dialog box](#)
 - [Capture frame](#)
 - [Capture File\(s\) frame](#)
 - [Stop Capture... frame](#)
 - [Display Options frame](#)
 - [Name Resolution frame](#)
 - [Buttons](#)
- [6. The “Edit Interface Settings” dialog box](#)
- [7. The “Compile Results” dialog box](#)
- [8. The “Add New Interfaces” dialog box](#)
 - [Add or remove pipes](#)
 - [Add or hide local interfaces](#)
 - [Add or hide remote interfaces](#)
- [9. The “Remote Capture Interfaces” dialog box](#)
 - [Remote Capture Interfaces](#)
 - [Remote Capture Settings](#)
- [10. The “Interface Details” dialog box](#)
 - [11. Capture files and file modes](#)
- [12. Link-layer header type](#)
- [13. Filtering while capturing](#)
 - [Automatic Remote Traffic Filtering](#)
 - [Stop the running capture](#)
 - [Restart a running capture](#)
- [Sources](#)

Texte en anglais

1. Introduction

Capturing live network data is one of the major features of Wireshark.

The Wireshark capture engine provides the following features:

- Capture from different kinds of network hardware such as Ethernet or 802.11.
- Stop the capture on different triggers such as the amount of captured data, elapsed time, or the number of packets.
- Simultaneously show decoded packets while Wireshark is capturing.
- Filter packets, reducing the amount of data to be captured. See [Section 4.13, “Filtering while capturing”](#).
- Save packets in multiple files while doing a long term capture, optionally rotating through a fixed number of files (a “ringbuffer”). See [Section 4.11, “Capture files and file modes”](#).
- Simultaneously capture from multiple network interfaces.

The capture engine still lacks the following features:

- Stop capturing (or perform some other action) depending on the captured data.

2. Prerequisites

Setting up Wireshark to capture packets for the first time can be tricky. A comprehensive guide “How To setup a Capture” is available at <https://wiki.wireshark.org/CaptureSetup>.

Here are some common pitfalls:

- You may need special privileges to start a live capture.
- You need to choose the right network interface to capture packet data from.
- You need to capture at the right place in the network to see the traffic you want to see.

If you have any problems setting up your capture environment you should have a look at the guide mentioned above.

3. Start Capturing

The following methods can be used to start capturing packets with Wireshark:

- You can double-click on an interface in the main window.
- You can get an overview of the available interfaces using the “Capture Interfaces” dialog box (Capture → Options...). See [Figure 4.1, “The “Capture Interfaces” dialog box on Microsoft Windows”](#) or [Figure 4.2, “The “Capture Interfaces” dialog box on Unix/Linux”](#) for more information. You can start a capture from this dialog box using the Start button.
- You can immediately start a capture using your current settings by selecting Capture → Start or by clicking the first toolbar button.
- If you already know the name of the capture interface you can start Wireshark from the command line:

```
$ wireshark -i eth0 -k
```

This will start Wireshark capturing on interface eth0. More details can be found at [Section 10.2, “Start Wireshark from the command line”](#).

4. The “Capture Interfaces” dialog box

When you select Capture → Options... from the main menu Wireshark pops up the “Capture Interfaces” dialog box as shown in [Figure 4.1, “The “Capture Interfaces” dialog box on Microsoft Windows”](#) or [Figure 4.2, “The “Capture Interfaces” dialog box on Unix/Linux”](#).

Both you and your OS can hide interfaces

This dialog box will only show the local interfaces Wireshark can access. It will also hide interfaces marked as hidden in [Section 10.5.1, “Interface Options”](#). As Wireshark might not be able to detect all local interfaces and it cannot detect the remote interfaces available there could be more capture interfaces available than listed.

It is possible to select more than one interface and capture from them simultaneously.

Figure 4.1. The “Capture Interfaces” dialog box on Microsoft Windows

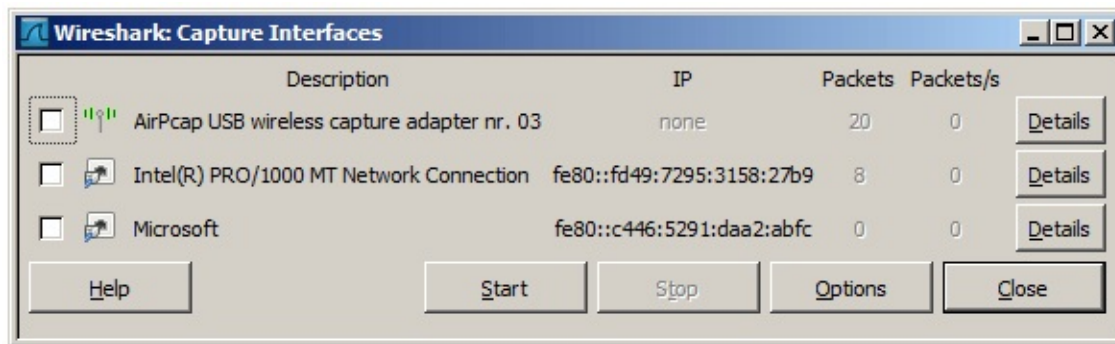
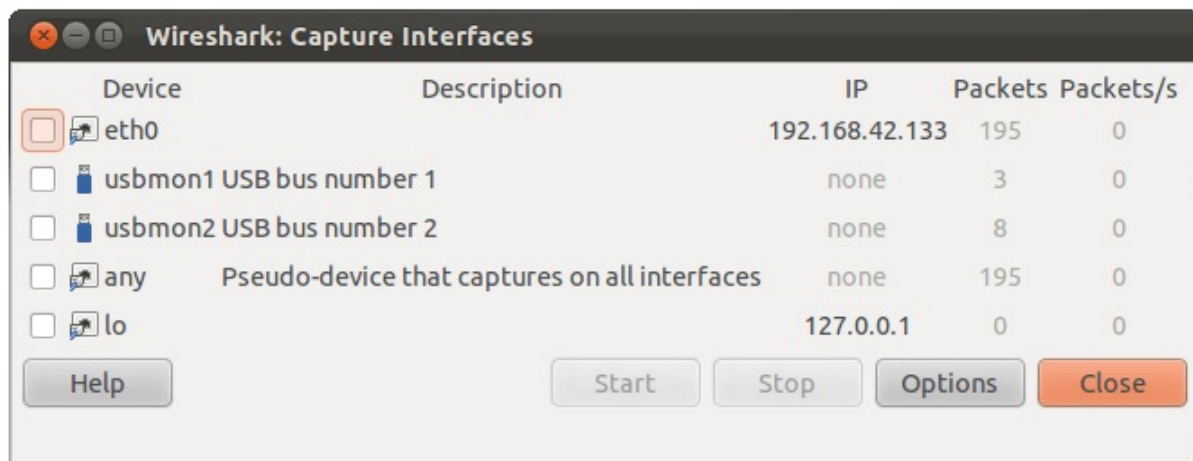


Figure 4.2. The “Capture Interfaces” dialog box on Unix/Linux

**Device (Unix/Linux only)**

The interface device name.

Description

The interface description provided by the operating system, or the user defined comment added in [Section 10.5.1](#), “Interface Options”.

IP

The first IP address Wireshark could find for this interface. You can click on the address to cycle through other addresses assigned to it, if available. If no address could be found “none” will be displayed.

Packets

The number of packets captured from this interface, since this dialog was opened. Will be greyed out, if no packet was captured in the last second.

Packets/s

Number of packets captured in the last second. Will be greyed out, if no packet was captured in the last second.

Stop

Stop a currently running capture.

Start

Start a capture on all selected interfaces immediately, using the settings from the last capture or the default settings, if no options have been set.

Options

Open the Capture Options dialog with the marked interfaces selected. See [Section 4.5](#), “The “Capture Options” dialog box”.

Details (Microsoft Windows only)

Open a dialog with detailed information about the interface. See [Section 4.10, “The “Interface Details” dialog box”](#).

Help

Show this help page.

Close

Close this dialog box.

5. The “Capture Options” dialog box

When you select Capture → Options... (or use the corresponding item in the main toolbar), Wireshark pops up the “Capture Options” dialog box as shown in [Figure 4.3, “The “Capture Options” dialog box”](#).

Figure 4.3. The “Capture Options” dialog box



If you are unsure which options to choose in this dialog box just try keeping the defaults as this should work well in many cases.

Capture frame

The table shows the settings for all available interfaces:

- The name of the interface and its IP addresses. If no address could be resolved from the system, “none” will be shown.

Loopback interfaces are not available on Windows platforms.

- The link-layer header type.
- The information whether promiscuous mode is enabled or disabled.
- The maximum amount of data that will be captured for each packet. The default value is set to the 65535 bytes.
- The size of the kernel buffer that is reserved to keep the captured packets.
- The information whether packets will be captured in monitor mode (Unix/Linux only).
- The chosen capture filter.

By marking the checkboxes in the first column the interfaces are selected to be captured from. By double-clicking on an interface the “Edit Interface Settings” dialog box as shown in [Figure 4.4, “The “Edit Interface Settings” dialog box”](#) will be opened.

Capture on all interfaces

As Wireshark can capture on multiple interfaces it is possible to choose to capture on all available interfaces.

Capture all packets in promiscuous mode

This checkbox allows you to specify that Wireshark should put all interfaces in promiscuous mode when capturing.

Capture Filter

This field allows you to specify a capture filter for all interfaces that are currently selected. Once a filter has been entered in this field, the newly selected interfaces will inherit the filter. Capture filters are discussed in more details in [Section 4.13, “Filtering while capturing”](#). It defaults to empty, or no filter.

You can also click on the Capture Filter button and Wireshark will bring up the Capture Filters dialog box and allow you to create and/or select a filter. Please see [Section 6.6, “Defining and saving filters”](#)

Compile selected BPFs

This button allows you to compile the capture filter into BPF code and pop up a window showing you the resulting pseudo code. This can help in understanding the working of the capture filter you created. The Compile Selected BPFs button leads you to [Figure 4.5, “The “Compile Results” dialog box”](#).

Linux power user tip

The execution of BPFs can be sped up on Linux by turning on BPF JIT by executing

```
$ echo 1 >/proc/sys/net/core/bpf_jit_enable
```

if it is not enabled already. To make the change persistent you can use [sysfsutils](#).

Manage Interfaces

The Manage Interfaces button opens the [Figure 4.6, “The “Add New Interfaces” dialog box”](#) where pipes can be defined, local interfaces scanned or hidden, or remote interfaces added (Windows only).

Capture File(s) frame

An explanation about capture file usage can be found in [Section 4.11, “Capture files and file modes”](#).

File

This field allows you to specify the file name that will be used for the capture file. This field is left blank by default. If the field is left blank, the capture data will be stored in a temporary file. See [Section 4.11, “Capture files and file modes”](#) for details.

You can also click on the button to the right of this field to browse through the filesystem.

Use multiple files

Instead of using a single file Wireshark will automatically switch to a new one if a specific trigger condition is reached.

Use pcap-ng format

This checkbox allows you to specify that Wireshark saves the captured packets in pcap-ng format. This next generation capture file format is currently in development. If more than one interface is chosen for capturing, this checkbox is set by default. See <https://wiki.wireshark.org/Development/PcapNg> for more details on pcap-ng.

Next file every n megabyte(s)

Multiple files only. Switch to the next file after the given number of byte(s)/kilobyte(s)/megabyte(s)/gigabyte(s) have been captured.

Next file every n minute(s)

Multiple files only: Switch to the next file after the given number of second(s)/minutes(s)/hours(s)/days(s) have elapsed.

Ring buffer with n files

Multiple files only: Form a ring buffer of the capture files with the given number of files.

Stop capture after n file(s)

Multiple files only: Stop capturing after switching to the next file the given number of times.

Stop Capture... frame

... after n packet(s)

Stop capturing after the given number of packets have been captured.

... after n megabytes(s)

Stop capturing after the given number of byte(s)/kilobyte(s)/megabyte(s)/gigabyte(s) have been captured. This option is greyed out if “Use multiple files” is selected.

... after n minute(s)

Stop capturing after the given number of second(s)/minutes(s)/hours(s)/days(s) have elapsed.

Display Options frame

Update list of packets in real time

This option allows you to specify that Wireshark should update the packet list pane in real time. If you do not specify this, Wireshark does not display any packets until you stop the capture. When you check this, Wireshark captures in a separate process and feeds the captures to the display process.

Automatic scrolling in live capture

This option allows you to specify that Wireshark should scroll the packet list pane as new packets come in, so you are always looking at the last packet. If you do not specify this Wireshark simply adds new packets onto the end of the list but does not scroll the packet list pane. This option is greyed out if “Update list of packets in real time” is disabled.

Name Resolution frame

Enable MAC name resolution

This option allows you to control whether or not Wireshark translates MAC addresses into names. See [Section 7.8](#), “Name Resolution”.

Enable network name resolution

This option allows you to control whether or not Wireshark translates network addresses into names. See [Section 7.8](#), “Name Resolution”.

Enable transport name resolution

This option allows you to control whether or not Wireshark translates transport addresses into protocols. See [Section 7.8](#), “Name Resolution”.

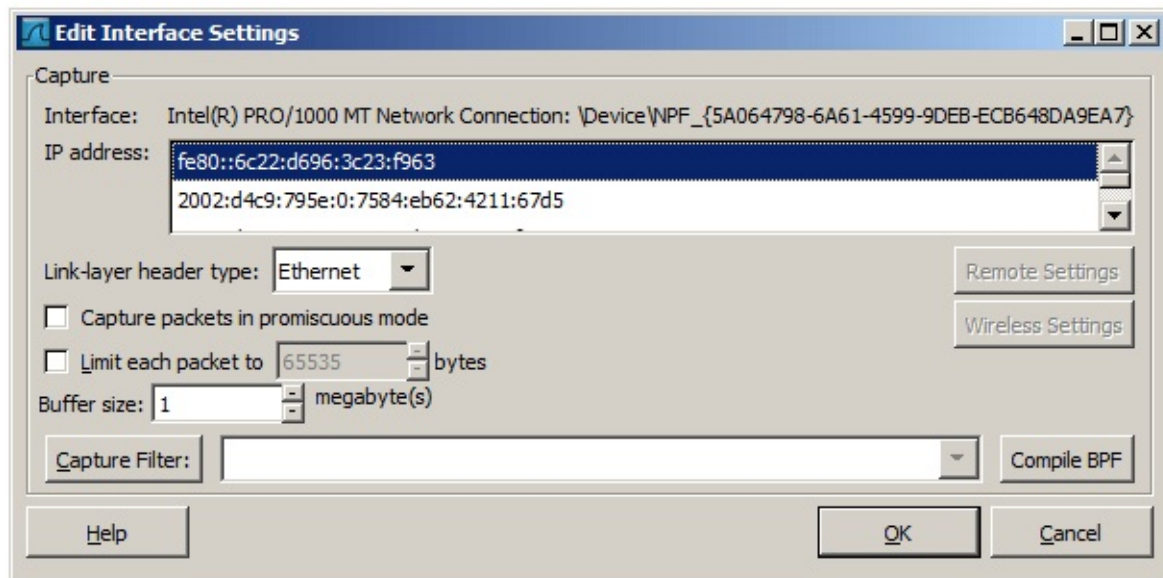
Buttons

Once you have set the values you desire and have selected the options you need, simply click on Start to commence the capture or Cancel to cancel the capture.

6. The “Edit Interface Settings” dialog box

If you double-click on an interface in [Figure 4.3](#), “The “Capture Options” dialog box” the following dialog box pops up.

Figure 4.4. The “Edit Interface Settings” dialog box



You can set the following fields in this dialog box:

IP address

The IP address(es) of the selected interface. If no address could be resolved from the system “none” will be shown.

Link-layer header type

Unless you are in the rare situation that requires this keep the default setting. For a detailed description. See [Section 4.12, “Link-layer header type”](#)

Wireless settings (Windows only)

Here you can set the settings for wireless capture using the AirPCap adapter. For a detailed description see the AirPCap Users Guide.

Remote settings (Windows only)

Here you can set the settings for remote capture. For a detailed description see [Section 4.9, “The “Remote Capture Interfaces” dialog box”](#)

Capture packets in promiscuous mode

This checkbox allows you to specify that Wireshark should put the interface in promiscuous mode when capturing. If you do not specify this Wireshark will only capture the packets going to or from your computer (not all packets on your LAN segment).

If some other process has put the interface in promiscuous mode you may be capturing in promiscuous mode even if you turn off this option.

Even in promiscuous mode you still won't necessarily see all packets on your LAN segment. See the [Wireshark FAQ](#) for more information.

Limit each packet to n bytes

This field allows you to specify the maximum amount of data that will be captured for each packet, and is sometimes referred to as the *snapshot*. If disabled the value is set to the maximum 65535 which will be sufficient for most protocols. Some rules of thumb:

- If you are unsure just keep the default value.
- If you don't need or don't want all of the data in a packet - for example, if you only need the link-layer, IP, and TCP headers - you might want to choose a small snapshot length, as less CPU time is required for copying

packets, less buffer space is required for packets, and thus perhaps fewer packets will be dropped if traffic is very heavy.

- If you don't capture all of the data in a packet you might find that the packet data you want is in the part that's dropped or that reassembly isn't possible as the data required for reassembly is missing.

Buffer size: n megabyte(s)

Enter the buffer size to be used while capturing. This is the size of the kernel buffer which will keep the captured packets, until they are written to disk. If you encounter packet drops, try increasing this value.

Capture packets in monitor mode (Unix/Linux only)

This checkbox allows you to setup the Wireless interface to capture all traffic it can receive, not just the traffic on the BSS to which it is associated, which can happen even when you set promiscuous mode. Also it might be necessary to turn this option on in order to see IEEE 802.11 headers and/or radio information from the captured frames.

In monitor mode the adapter might disassociate itself from the network it was associated to.

Capture Filter

This field allows you to specify a capture filter. Capture filters are discussed in more details in [Section 4.13, "Filtering while capturing"](#). It defaults to empty, or no filter.

You can also click on the Capture Filter button and Wireshark will bring up the "Capture Filters" dialog box and allow you to create and/or select a filter. Please see [Section 6.6, "Defining and saving filters"](#)

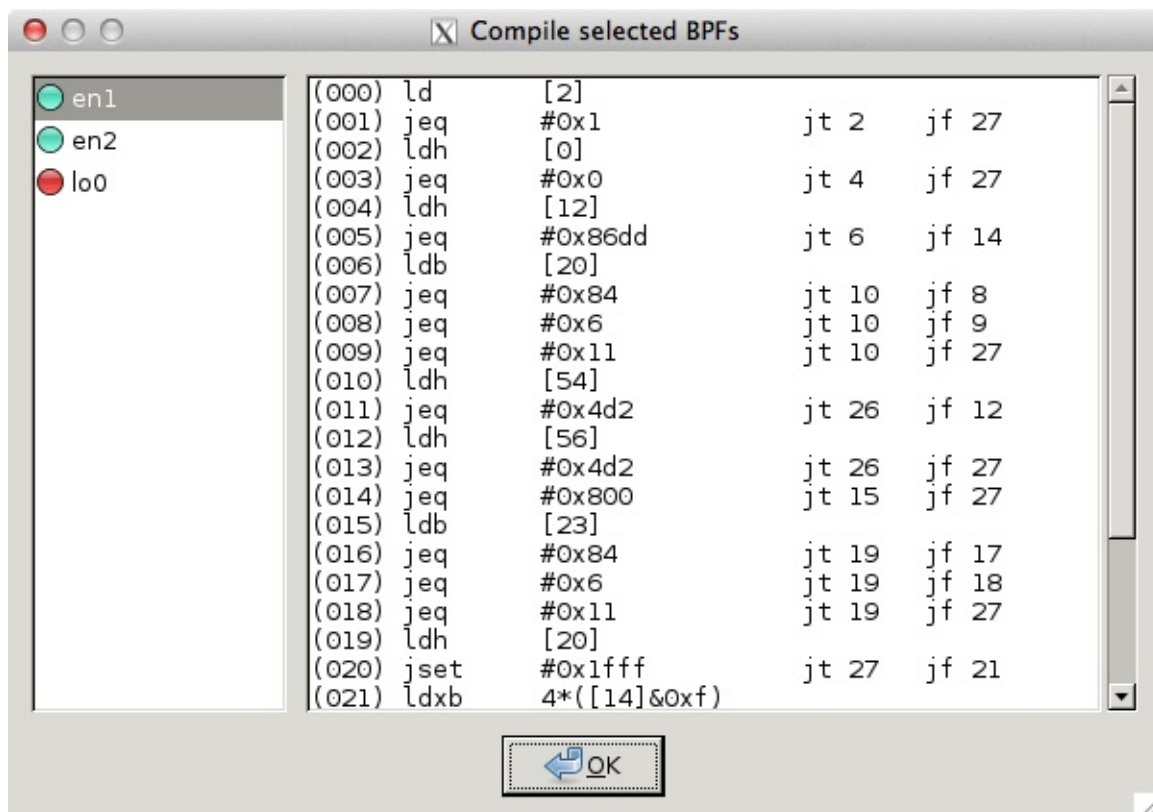
Compile BPF

This button allows you to compile the capture filter into BPF code and pop up a window showing you the resulting pseudo code. This can help in understanding the working of the capture filter you created.

7. The "Compile Results" dialog box

This figure shows the compile results of the selected interfaces.

Figure 4.5. The "Compile Results" dialog box

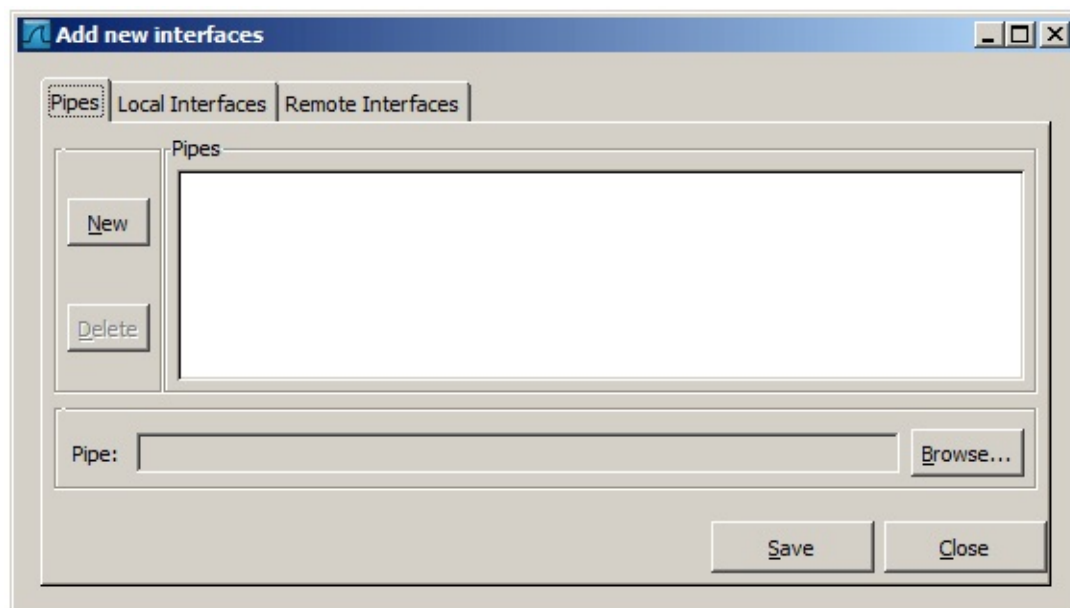


In the left window the interface names are listed. The results of an individual interface are shown in the right window when it is selected.

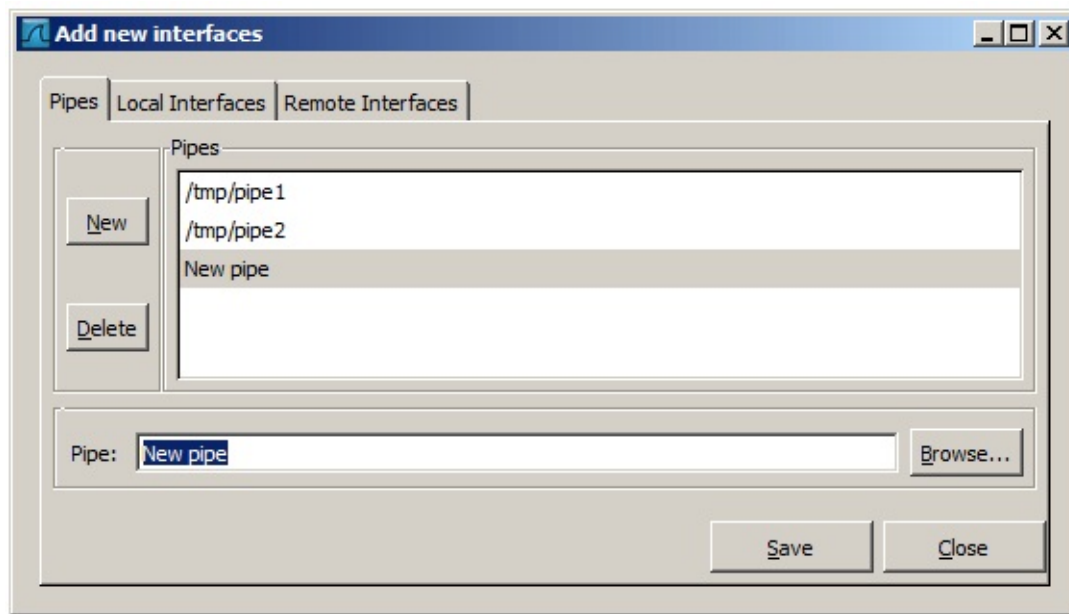
8. The “Add New Interfaces” dialog box

As a central point to manage interfaces this dialog box consists of three tabs to add or remove interfaces.

Figure 4.6. The “Add New Interfaces” dialog box



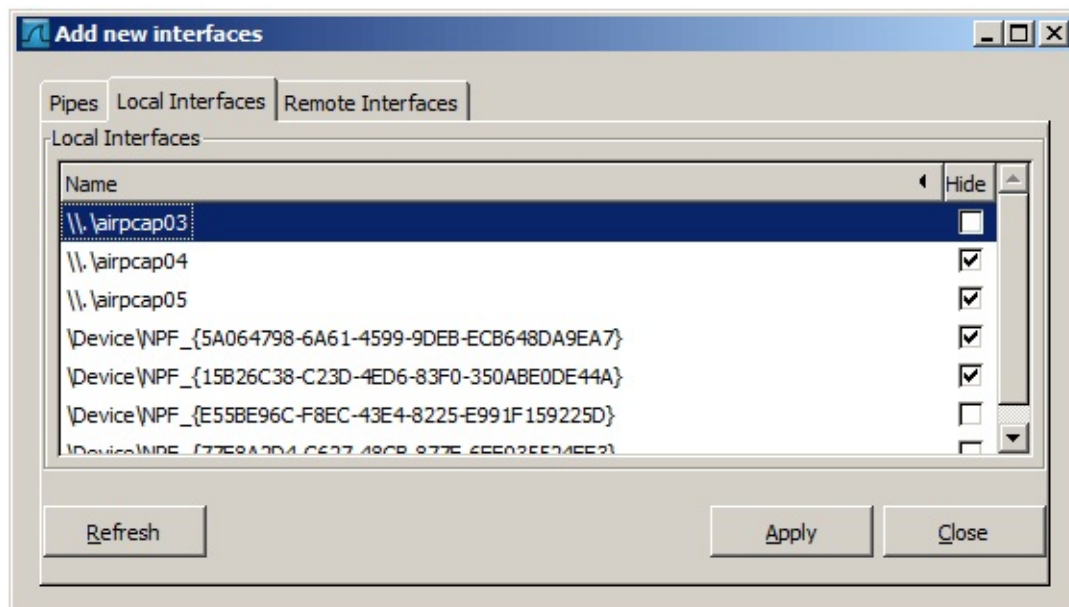
Add or remove pipes

Figure 4.7. The “Add New Interfaces - Pipes” dialog box

To successfully add a pipe, this pipe must have already been created. Click the New button and type the name of the pipe including its path. Alternatively, the Browse button can be used to locate the pipe. With the Save button the pipe is added to the list of available interfaces. Afterwards, other pipes can be added.

To remove a pipe from the list of interfaces it first has to be selected. Then click the Delete button.

Add or hide local interfaces

Figure 4.8. The “Add New Interfaces - Local Interfaces” dialog box

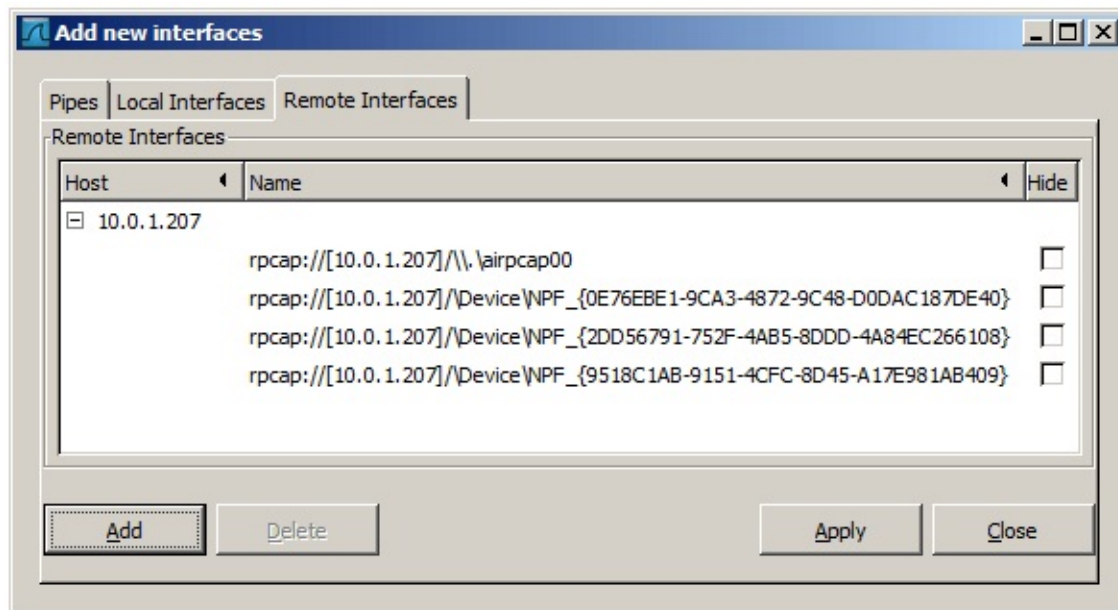
The tab “Local Interfaces” contains a list of available local interfaces, including the hidden ones, which are not shown in the other lists.

If a new local interface is added, for example, a wireless interface has been activated, it is not automatically added to the list to prevent the constant scanning for a change in the list of available interfaces. To renew the list a rescan can be done.

One way to hide an interface is to change the preferences. If the “Hide” checkbox is activated and the Apply button clicked, the interface will not be seen in the lists of the “Capture Interfaces” dialog box any more. The changes are also saved in the `preferences` file.

Add or hide remote interfaces

Figure 4.9. The “Add New Interfaces - Remote Interfaces” dialog box



In this tab interfaces on remote hosts can be added. One or more of these interfaces can be hidden. In contrast to the local interfaces they are not saved in the `preferences` file.

To remove a host including all its interfaces from the list, it has to be selected. Then click the Delete button.

For a detailed description see [Section 4.9, “The “Remote Capture Interfaces” dialog box”](#)

9. The “Remote Capture Interfaces” dialog box

Besides doing capture on local interfaces Wireshark is capable of reaching out across the network to a so called capture daemon or service processes to receive captured data from.

Microsoft Windows only

This dialog and capability is only available on Microsoft Windows. On Linux/Unix you can achieve the same effect (securely) through an SSH tunnel.

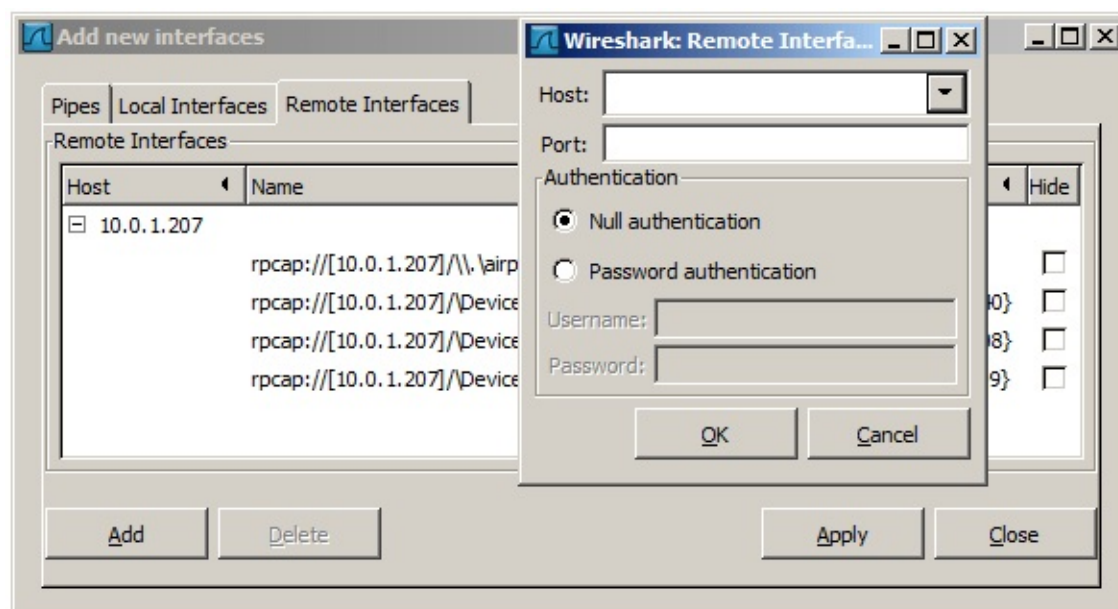
The Remote Packet Capture Protocol service must first be running on the target platform before Wireshark can connect to it. The easiest way is to install WinPcap from <https://www.winpcap.org/install/> on the target. Once installation is completed go to the Services control panel, find the Remote Packet Capture Protocol service and start it.

Make sure you have outside access to port 2002 on the target platform. This is the port where the Remote Packet Capture Protocol service can be reached by default.

To access the Remote Capture Interfaces dialog use the “Add New Interfaces - Remote” dialog. See [Figure 4.9](#), “The “Add New Interfaces - Remote Interfaces” dialog box” and select Add.

Remote Capture Interfaces

Figure 4.10. The “Remote Capture Interfaces” dialog box



You have to set the following parameters in this dialog:

Host

Enter the IP address or host name of the target platform where the Remote Packet Capture Protocol service is listening. The drop down list contains the hosts that have previously been successfully contacted. The list can be emptied by choosing “Clear list” from the drop down list.

Port

Set the port number where the Remote Packet Capture Protocol service is listening on. Leave open to use the default port (2002).

Null authentication

Select this if you don’t need authentication to take place for a remote capture to be started. This depends on the target platform. Configuring the target platform like this makes it insecure.

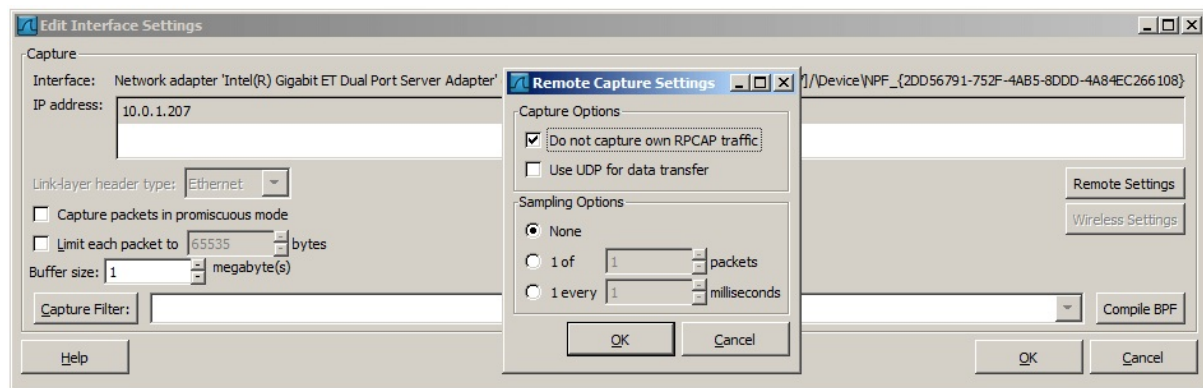
Password authentication

This is the normal way of connecting to a target platform. Set the credentials needed to connect to the Remote Packet Capture Protocol service.

Remote Capture Settings

The remote capture can be further fine tuned to match your situation. The Remote Settings button in [Figure 4.4](#), “The “Edit Interface Settings” dialog box” gives you this option. It pops up the dialog shown in [Figure 4.11](#), “The “Remote Capture Settings” dialog box”.

Figure 4.11. The “Remote Capture Settings” dialog box



You can set the following parameters in this dialog:

Do not capture own RPCAP traffic

This option sets a capture filter so that the traffic flowing back from the Remote Packet Capture Protocol service to Wireshark isn't captured as well and also send back. The recursion in this saturates the link with duplicate traffic.

You only should switch this off when capturing on an interface other than the interface connecting back to Wireshark.

Use UDP for data transfer

Remote capture control and data flows over a TCP connection. This option allows you to choose an UDP stream for data transfer.

Sampling option None

This option instructs the Remote Packet Capture Protocol service to send back all captured packets which have passed the capture filter. This is usually not a problem on a remote capture session with sufficient bandwidth.

Sampling option 1 of x packets

This option limits the Remote Packet Capture Protocol service to send only a sub sampling of the captured data, in terms of number of packets. This allows capture over a narrow band remote capture session of a higher bandwidth interface.

Sampling option 1 every x milliseconds

This option limits the Remote Packet Capture Protocol service to send only a sub sampling of the captured data in terms of time. This allows capture over a narrow band capture session of a higher bandwidth interface.

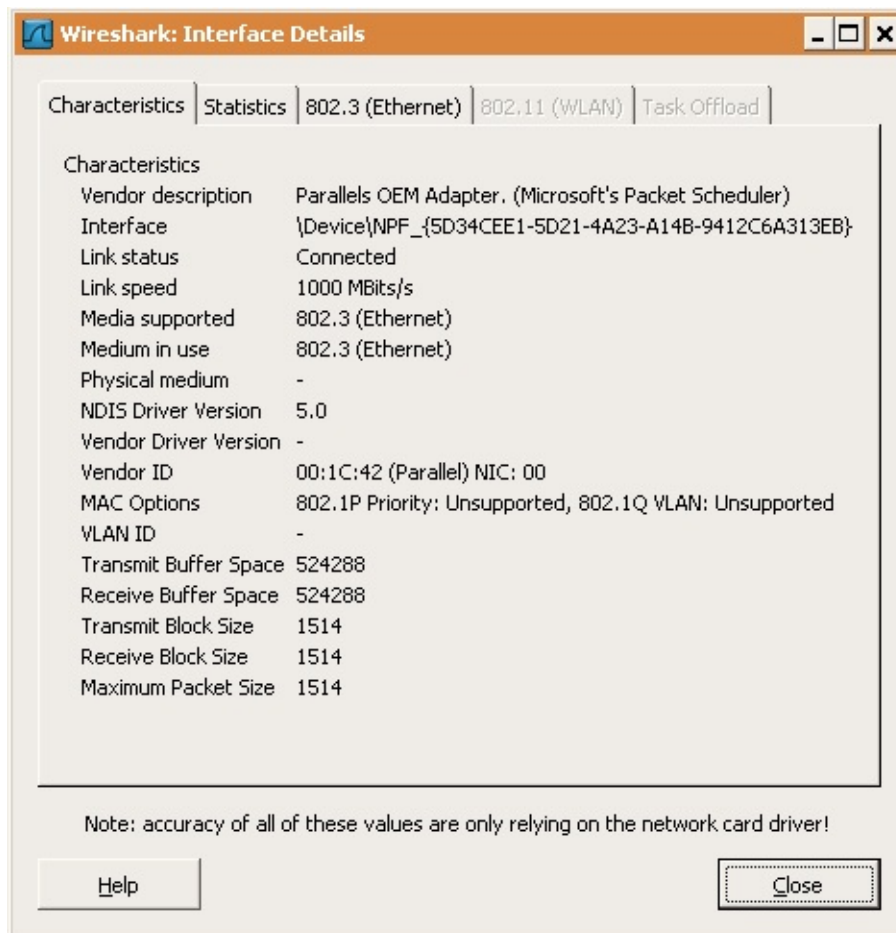
10. The “Interface Details” dialog box

When you select Details from the Capture Interface menu, Wireshark pops up the “Interface Details” dialog box as shown in [Figure 4.12](#), “The “Interface Details” dialog box”. This dialog shows various characteristics and statistics for the selected interface.

Microsoft Windows only

This dialog is only available on Microsoft Windows

Figure 4.12. The “Interface Details” dialog box

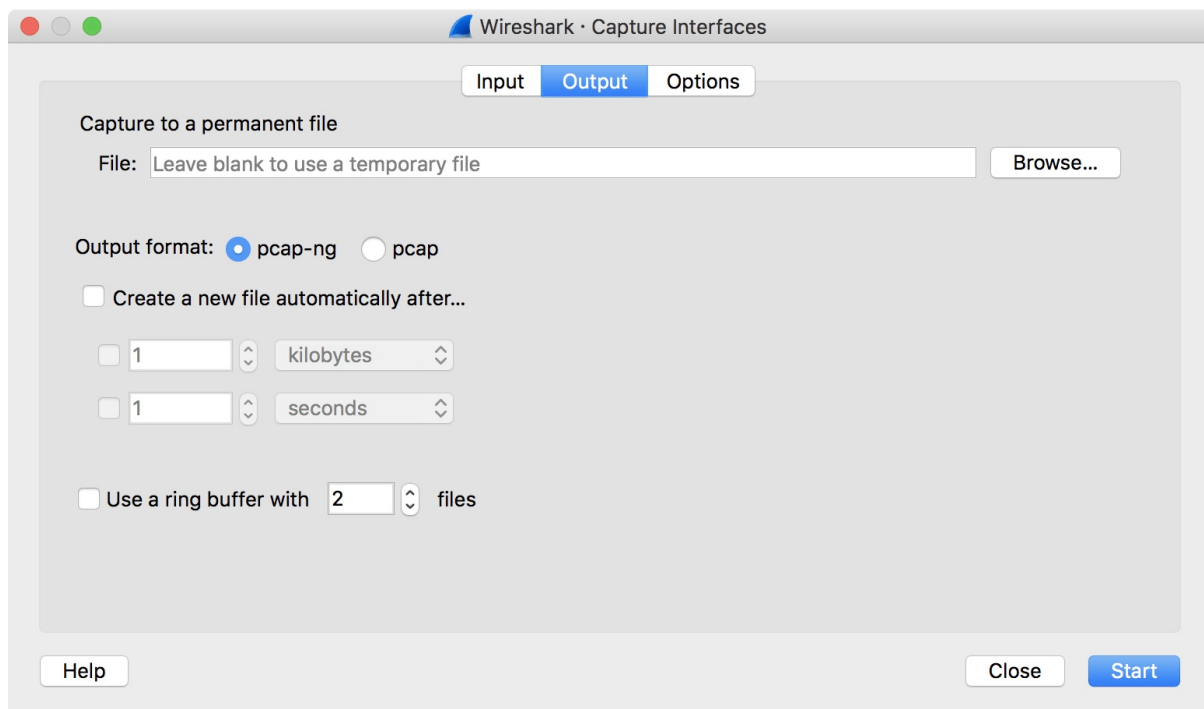


11. Capture files and file modes

While capturing the underlying libpcap capturing engine will grab the packets from the network card and keep the packet data in a (relatively) small kernel buffer. This data is read by Wireshark and saved into a capture file.

By default Wireshark saves packets to a temporary file. You can also tell Wireshark to save to a specific ("permanent") file and switch to a different file after a given time has elapsed or a given number of packets have been captured. These options are controlled in the "Output" tab in the "Capture Options" dialog.

Figure 4.13. Capture output options



Working with large files (several hundred MB) can be quite slow. If you plan to do a long term capture or capturing from a high traffic network, think about using one of the “Multiple files” options. This will spread the captured packets over several smaller files which can be much more pleasant to work with.

Using Multiple files may cut context related information. Wireshark keeps context information of the loaded packet data, so it can report context related problems (like a stream error) and keeps information about context related protocols (e.g. where data is exchanged at the establishing phase and only referred to in later packets). As it keeps this information only for the loaded file, using one of the multiple file modes may cut these contexts. If the establishing phase is saved in one file and the things you would like to see is in another, you might not see some of the valuable context related information.

Information about the folders used for capture files can be found in [Appendix B, Files and Folders](#).

Table 4.1. Capture file mode selected by capture options

File Name	“Create a new file...”	“Use a ring buffer...”	Mode	Resulting filename(s) used
-	-	-	<i>Single temporary file</i>	wiresharkXXXXXX (where XXXXXX is a unique number)
foo.cap	-	-	<i>Single named file</i>	foo.cap
foo.cap	x	-	<i>Multiple files, continuous</i>	foo_00001_20100205110102.cap, foo_00002_20100205110318.cap, ...
foo.cap	x	x	<i>Multiple files, ring buffer</i>	foo_00001_20100205110102.cap, foo_00002_20100205110318.cap, ...

Single temporary file

A temporary file will be created and used (this is the default). After capturing is stopped this file can be saved later under a user specified name.

Single named file

A single capture file will be used. If you want to place the new capture file in a specific folder choose this mode.

Multiple files, continuous

Like the “Single named file” mode, but a new file is created and used after reaching one of the multiple file switch conditions (one of the “Next file every ...” values).

Multiple files, ring buffer

Much like “Multiple files continuous”, reaching one of the multiple files switch conditions (one of the “Next file every ...” values) will switch to the next file. This will be a newly created file if value of “Ring buffer with n files” is not reached, otherwise it will replace the oldest of the formerly used files (thus forming a “ring”). This mode will limit the maximum disk usage, even for an unlimited amount of capture input data, only keeping the latest captured data.

12. Link-layer header type

In most cases you won’t have to modify link-layer header type. Some exceptions are as follows:

If you are capturing on an Ethernet device you might be offered a choice of “Ethernet” or “DOCSIS”. If you are capturing traffic from a Cisco Cable Modem Termination System that is putting DOCSIS traffic onto the Ethernet to be captured, select “DOCSIS”, otherwise select “Ethernet”.

If you are capturing on an 802.11 device on some versions of BSD you might be offered a choice of “Ethernet” or “802.11”. “Ethernet” will cause the captured packets to have fake (“cooked”) Ethernet headers. “802.11” will cause them to have full IEEE 802.11 headers. Unless the capture needs to be read by an application that doesn’t support 802.11 headers you should select “802.11”.

If you are capturing on an Endace DAG card connected to a synchronous serial line you might be offered a choice of “PPP over serial” or “Cisco HDLC”. If the protocol on the serial line is PPP, select “PPP over serial” and if the protocol on the serial line is Cisco HDLC, select “Cisco HDLC”.

If you are capturing on an Endace DAG card connected to an ATM network you might be offered a choice of “RFC 1483 IP-over-ATM” or “Sun raw ATM”. If the only traffic being captured is RFC 1483 LLC-encapsulated IP, or if the capture needs to be read by an application that doesn’t support SunATM headers, select “RFC 1483 IP-over-ATM”, otherwise select “Sun raw ATM”.

13. Filtering while capturing

Wireshark uses the libpcap filter language for capture filters. A brief overview of the syntax follows. Complete documentation can be found in the [pcap-filter man page](#). You can find a lot of Capture Filter examples at <https://wiki.wireshark.org/CaptureFilters>.

You enter the capture filter into the “Filter” field of the Wireshark “Capture Options” dialog box, as shown in [Figure 4.3](#), “The “Capture Options” dialog box”.

A capture filter takes the form of a series of primitive expressions connected by conjunctions (*and/or*) and optionally preceded by *not*:

```
[not] primitive [and|or [not] primitive ...]
```

An example is shown in [Example 4.1](#), “A capture filter for telnet that captures traffic to and from a particular host”.

Example 4.1. A capture filter for telnet that captures traffic to and from a particular host

A capture filter for telnet that captures traffic to and from a particular host

```
tcp port 23 and host 10.0.0.5
```

This example captures telnet traffic to and from the host 10.0.0.5, and shows how to use two primitives and the *and* conjunction. Another example is shown in [Example 4.2, “Capturing all telnet traffic not from 10.0.0.5”](#), and shows how to capture all telnet traffic except that from 10.0.0.5.

Example 4.2. Capturing all telnet traffic not from 10.0.0.5

Capturing all telnet traffic not from 10.0.0.5

```
tcp port 23 and not src host 10.0.0.5
```

A primitive is simply one of the following: *[src|dst] host *

This primitive allows you to filter on a host IP address or name. You can optionally precede the primitive with the keyword *src|dst* to specify that you are only interested in source or destination addresses. If these are not present, packets where the specified address appears as either the source or the destination address will be selected.

*ether [src|dst] host *

This primitive allows you to filter on Ethernet host addresses. You can optionally include the keyword *src|dst* between the keywords *ether* and *host* to specify that you are only interested in source or destination addresses. If these are not present, packets where the specified address appears in either the source or destination address will be selected.

*gateway host *

This primitive allows you to filter on packets that used *host* as a gateway. That is, where the Ethernet source or destination was *host* but neither the source nor destination IP address was *host*.

[src|dst] net \ [{mask }|{len }]

This primitive allows you to filter on network numbers. You can optionally precede this primitive with the keyword *src|dst* to specify that you are only interested in a source or destination network. If neither of these are present, packets will be selected that have the specified network in either the source or destination address. In addition, you can specify either the netmask or the CIDR prefix for the network if they are different from your own.

*[tcp|udp] [src|dst] port *

This primitive allows you to filter on TCP and UDP port numbers. You can optionally precede this primitive with the keywords *src|dst* and *tcp|udp* which allow you to specify that you are only interested in source or destination ports and TCP or UDP packets respectively. The keywords *tcp|udp* must appear before *src|dst*.

If these are not specified, packets will be selected for both the TCP and UDP protocols and when the specified address appears in either the source or destination port field.

*less|greater *

This primitive allows you to filter on packets whose length was less than or equal to the specified length, or greater than or equal to the specified length, respectively.

*ip|ether proto *

This primitive allows you to filter on the specified protocol at either the Ethernet layer or the IP layer.

ether|ip broadcast|multicast

This primitive allows you to filter on either Ethernet or IP broadcasts or multicasts.

*\ relop *

This primitive allows you to create complex filter expressions that select bytes or ranges of bytes in packets. Please see the pcap-filter man page at <http://www.tcpdump.org/manpages/pcap-filter.7.html> for more details.

Automatic Remote Traffic Filtering

If Wireshark is running remotely (using e.g. SSH, an exported X11 window, a terminal server, ...), the remote content has to be transported over the network, adding a lot of (usually unimportant) packets to the actually interesting traffic.

To avoid this, Wireshark tries to figure out if it's remotely connected (by looking at some specific environment variables) and automatically creates a capture filter that matches aspects of the connection.

The following environment variables are analyzed:

SSH_CONNECTION (ssh)

\\ \\

SSH_CLIENT (ssh)

\\

REMOTEHOST (tcsh, others?)

\

DISPLAY (x11)

[remote name]:\

SESSIONNAME (terminal server)

\

On Windows it asks the operating system if it's running in a Remote Desktop Services environment.

Stop the running capture

A running capture session will be stopped in one of the following ways:

1. Using the Stop button from the "Capture Info" dialog box.

The "Capture Info" dialog box might be hidden if the "Hide capture info dialog" option is used.

1. Using the Capture → Stop menu item.
2. Using the Stop toolbar button.
3. Pressing **Ctrl+E**.
4. The capture will be automatically stopped if one of the *Stop Conditions* is met, e.g. the maximum amount of data was captured.

Restart a running capture

A running capture session can be restarted with the same capture options as the last time, this will remove all packets previously captured. This can be useful, if some uninteresting packets are captured and there's no need to keep them.

Restart is a convenience function and equivalent to a capture stop following by an immediate capture start. A restart can be triggered in one of the following ways:

1. Using the Capture → Restart menu item.
2. Using the Restart toolbar button.

Sources

- https://www.wireshark.org/docs/wsug_html/#ChapterCapture

Travailler avec des captures Wireshark

- [1. Viewing packets you have captured](#)
- [2. Pop-up menus](#)
 - [2.1. Pop-up menu of the “Packet List” column header](#)
 - [2.2. Pop-up menu of the “Packet List” pane](#)
 - [2.3. Pop-up menu of the “Packet Details” pane](#)
- [3. Filtering packets while viewing](#)
- [4. Building display filter expressions](#)
 - [4.1. Display filter fields](#)
 - [4.2. Comparing values](#)
 - [4.3. Combining expressions](#)
 - [4.4. Substring Operator](#)
 - [4.5. Membership Operator](#)
 - [4.6. A Common Mistake](#)
- [5. The “Filter Expression” dialog box](#)
- [6. Defining and saving filters](#)
- [7. Defining and saving filter macros](#)
- [8. Finding packets](#)
 - [8.1. The “Find Packet” dialog box](#)
 - [8.2. The “Find Next” command](#)
 - [8.3. The “Find Previous” command](#)
- [9. Go to a specific packet](#)
 - [9.1. The “Go Back” command](#)
 - [9.2. The “Go Forward” command](#)
 - [9.3. The “Go to Packet” dialog box](#)
 - [9.4. The “Go to Corresponding Packet” command](#)
 - [9.5. The “Go to First Packet” command](#)
 - [9.6. The “Go to Last Packet” command](#)
- [10. Marking packets](#)
- [11. Ignoring packets](#)
- [12. Time display formats and time references](#)
 - [12.1. Packet time referencing](#)
- [Sources](#)

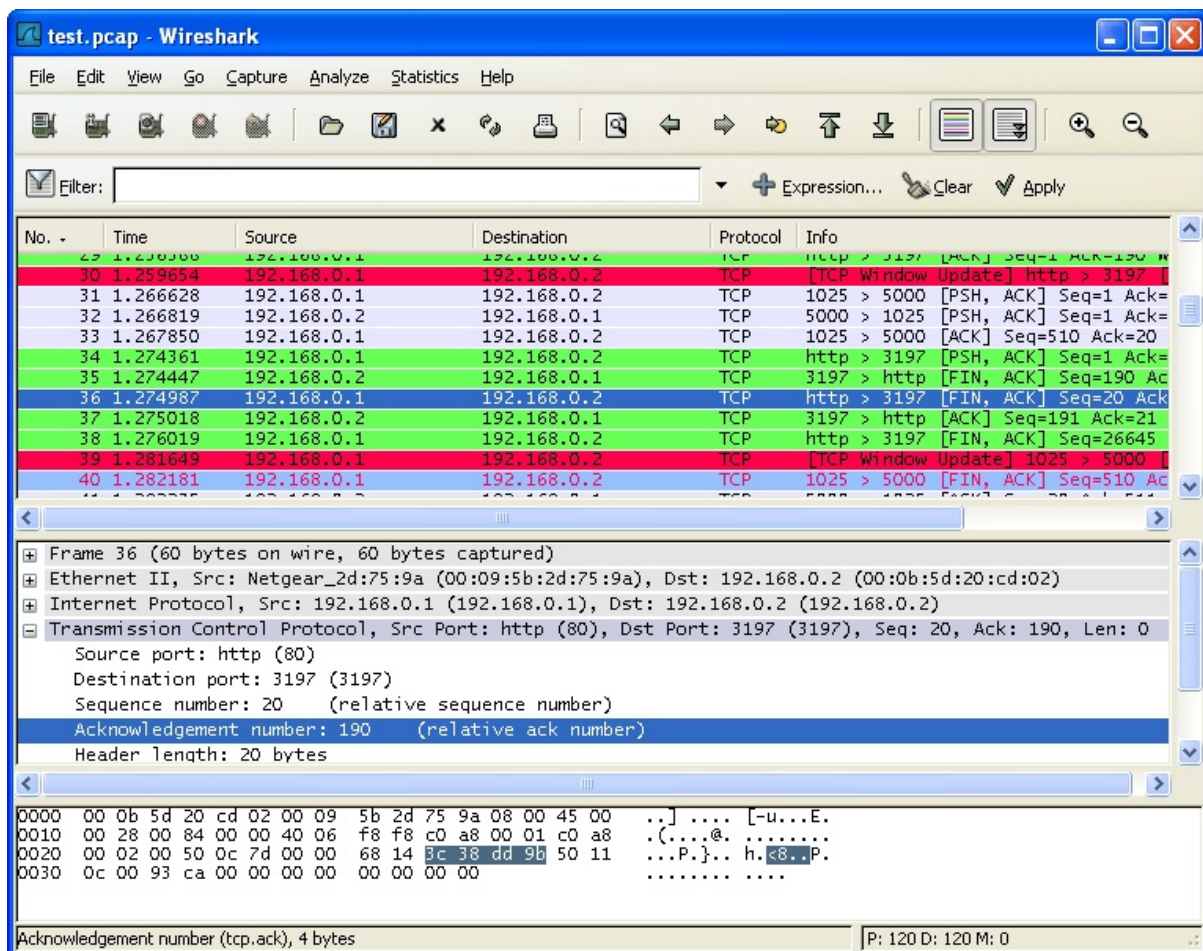
Texte en anglais

1. Viewing packets you have captured

Once you have captured some packets or you have opened a previously saved capture file, you can view the packets that are displayed in the packet list pane by simply clicking on a packet in the packet list pane, which will bring up the selected packet in the tree view and byte view panes.

You can then expand any part of the tree to view detailed information about each protocol in each packet. Clicking on an item in the tree will highlight the corresponding bytes in the byte view. An example with a TCP packet selected is shown in [Figure 6.1, “Wireshark with a TCP packet selected for viewing”](#). It also has the Acknowledgment number in the TCP header selected, which shows up in the byte view as the selected bytes.

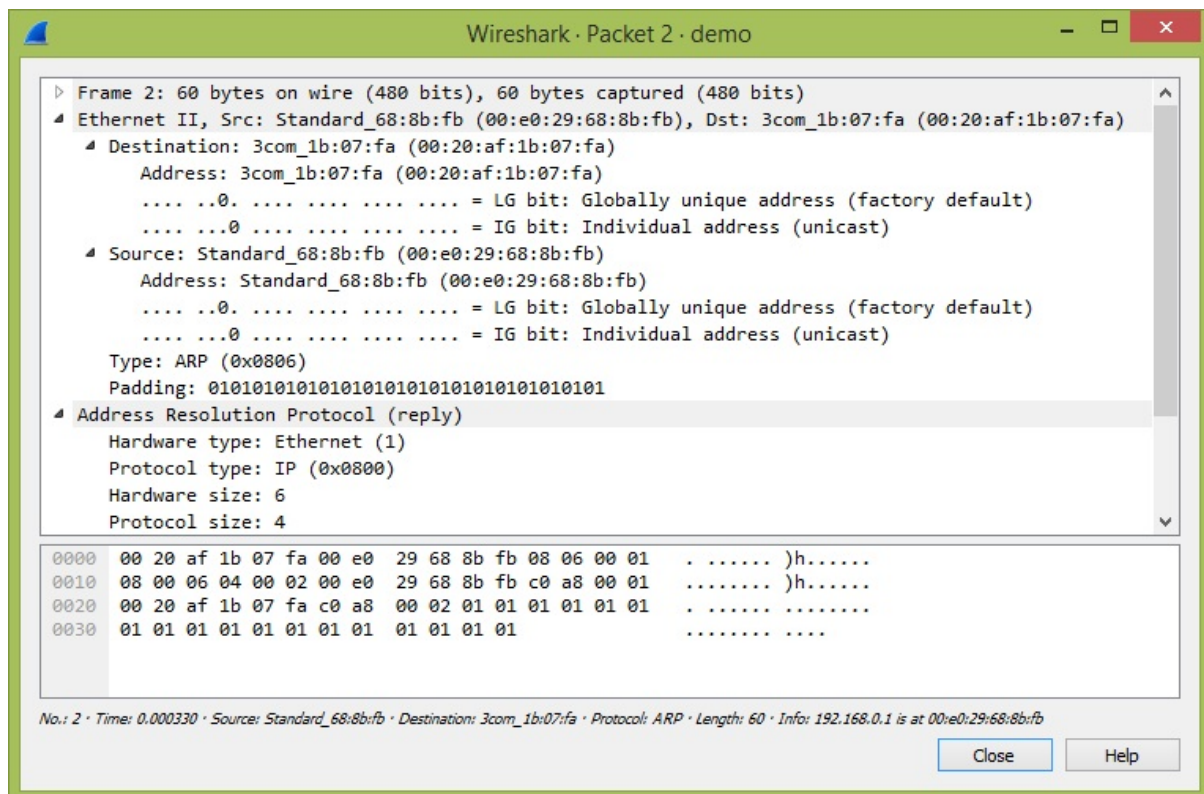
Figure 6.1. Wireshark with a TCP packet selected for viewing



You can also select and view packets the same way while Wireshark is capturing if you selected “Update list of packets in real time” in the “Capture Preferences” dialog box.

In addition you can view individual packets in a separate window as shown in [Figure 6.2, “Viewing a packet in a separate window”](#). You can do this by double-clicking on an item in the packet list or by selecting the packet in which you are interested in the packet list pane and selecting View → Show Packet in New Window. This allows you to easily compare two or more packets, even across multiple files.

Figure 6.2. Viewing a packet in a separate window



Along with double-clicking the packet list and using the main menu there are a number of other ways to open a new packet window:

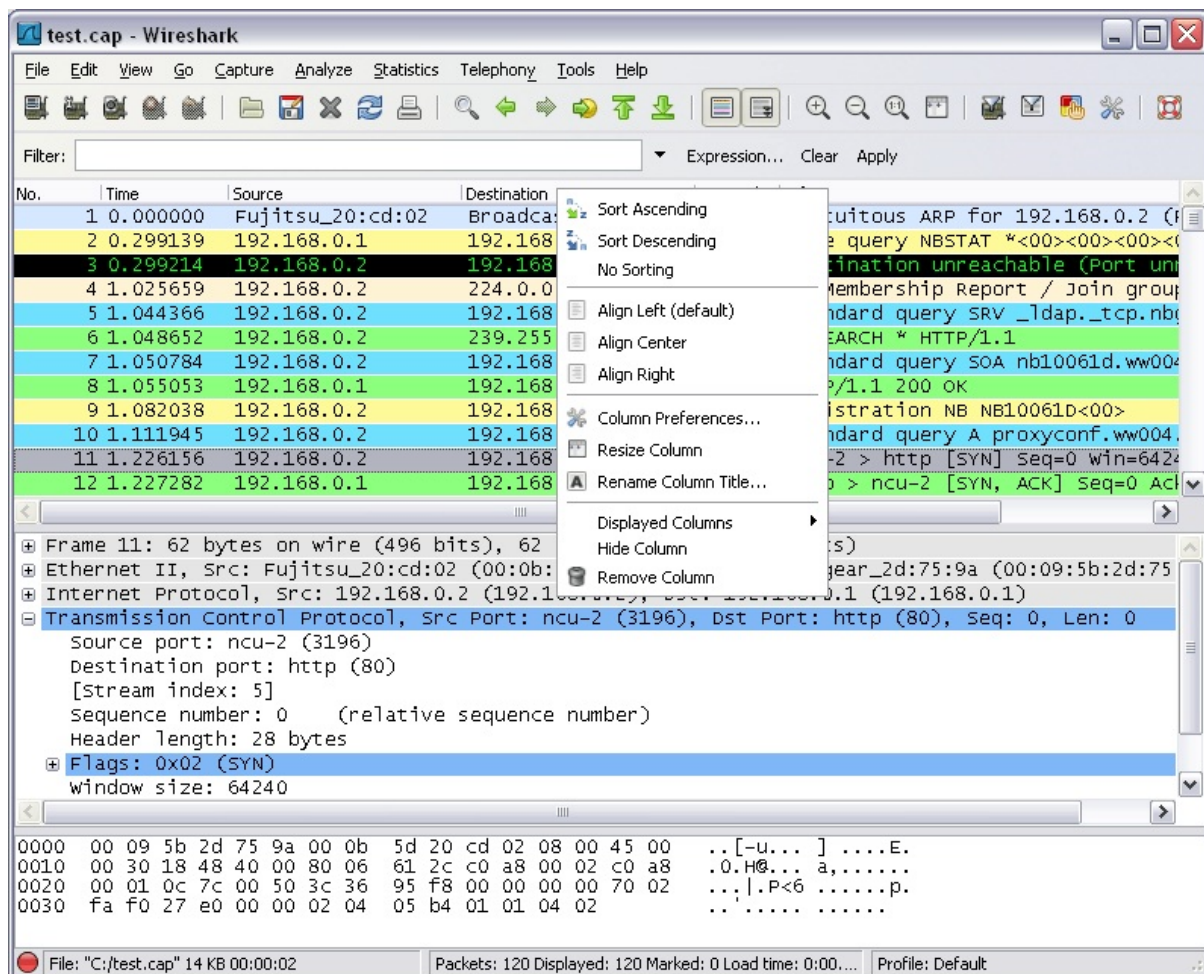
- Hold down the shift key and double-click on a frame link in the packet details.
- From [Table 6.2](#), “The menu items of the “Packet List” pop-up menu”.
- From [Table 6.3](#), “The menu items of the “Packet Details” pop-up menu”.

2. Pop-up menus

You can bring up a pop-up menu over either the “Packet List”, its column header, or “Packet Details” pane by clicking your right mouse button at the corresponding pane.

2.1. Pop-up menu of the “Packet List” column header

Figure 6.3. Pop-up menu of the “Packet List” column header



The following table gives an overview of which functions are available in this header, where to find the corresponding function in the main menu, and a short description of each item.

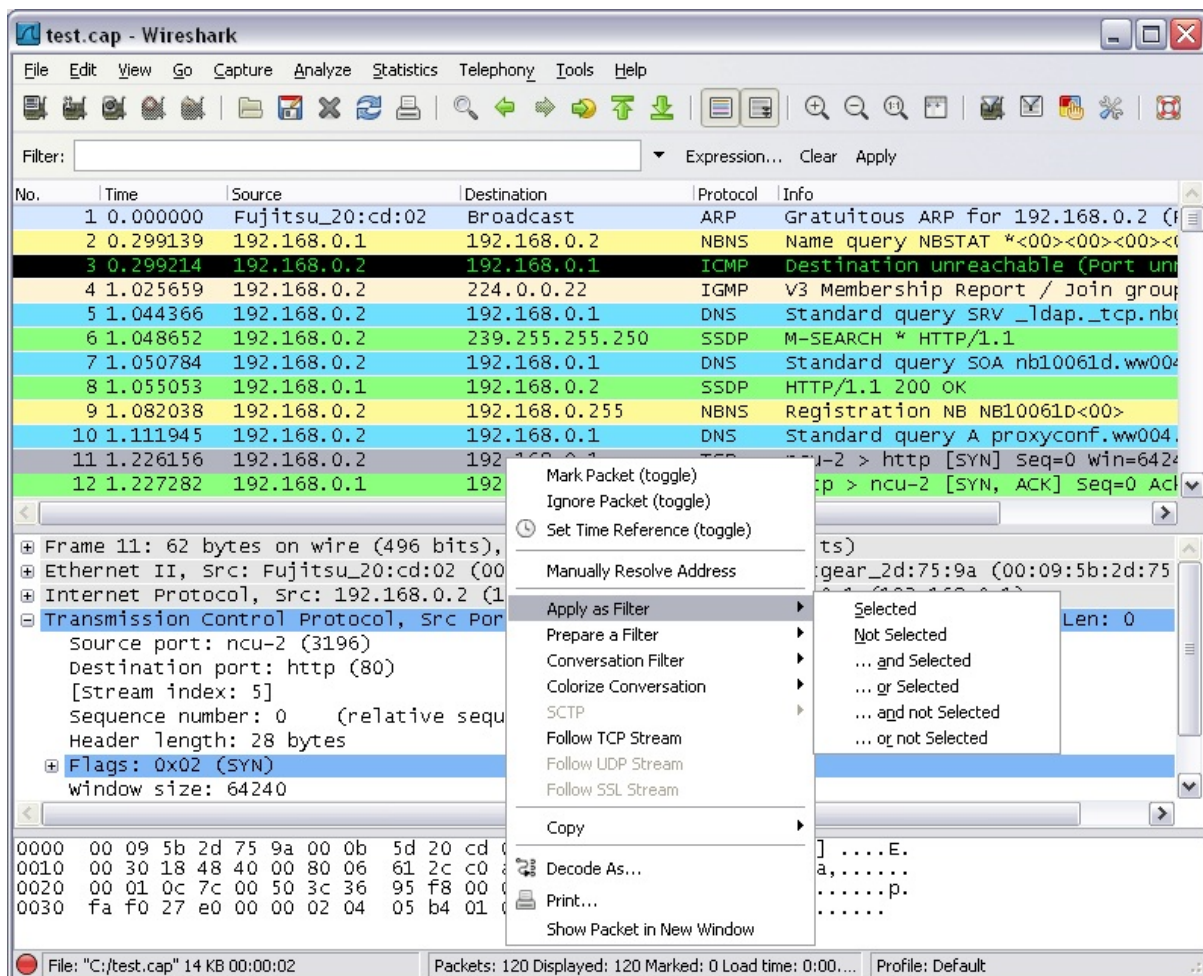
Table 6.1. The menu items of the “Packet List” column header pop-up menu

Item	Identical to main menu's item:	Description
Sort Ascending		Sort the packet list in ascending order based on this column.
Sort Descending		Sort the packet list in descending order based on this column.
No Sort		Remove sorting order based on this column.
Align Left		Set left alignment of the values in this column.
Align Center		Set center alignment of the values in this column.
Align Right		Set right alignment of the values in this column.
Column Preferences...		Open the Preferences dialog box on the column tab.
Resize Column		Resize the column to fit the values.
Rename Column Title		Allows you to change the title of the column header.

Displayed Column	View	This menu items folds out with a list of all configured columns. These columns can now be shown or hidden in the packet list.
Hide Column		Allows you to hide the column from the packet list.
Remove Column		Allows you to remove the column from the packet list.

2.2. Pop-up menu of the “Packet List” pane

Figure 6.4. Pop-up menu of the “Packet List” pane



The following table gives an overview of which functions are available in this pane, where to find the corresponding function in the main menu, and a short description of each item.

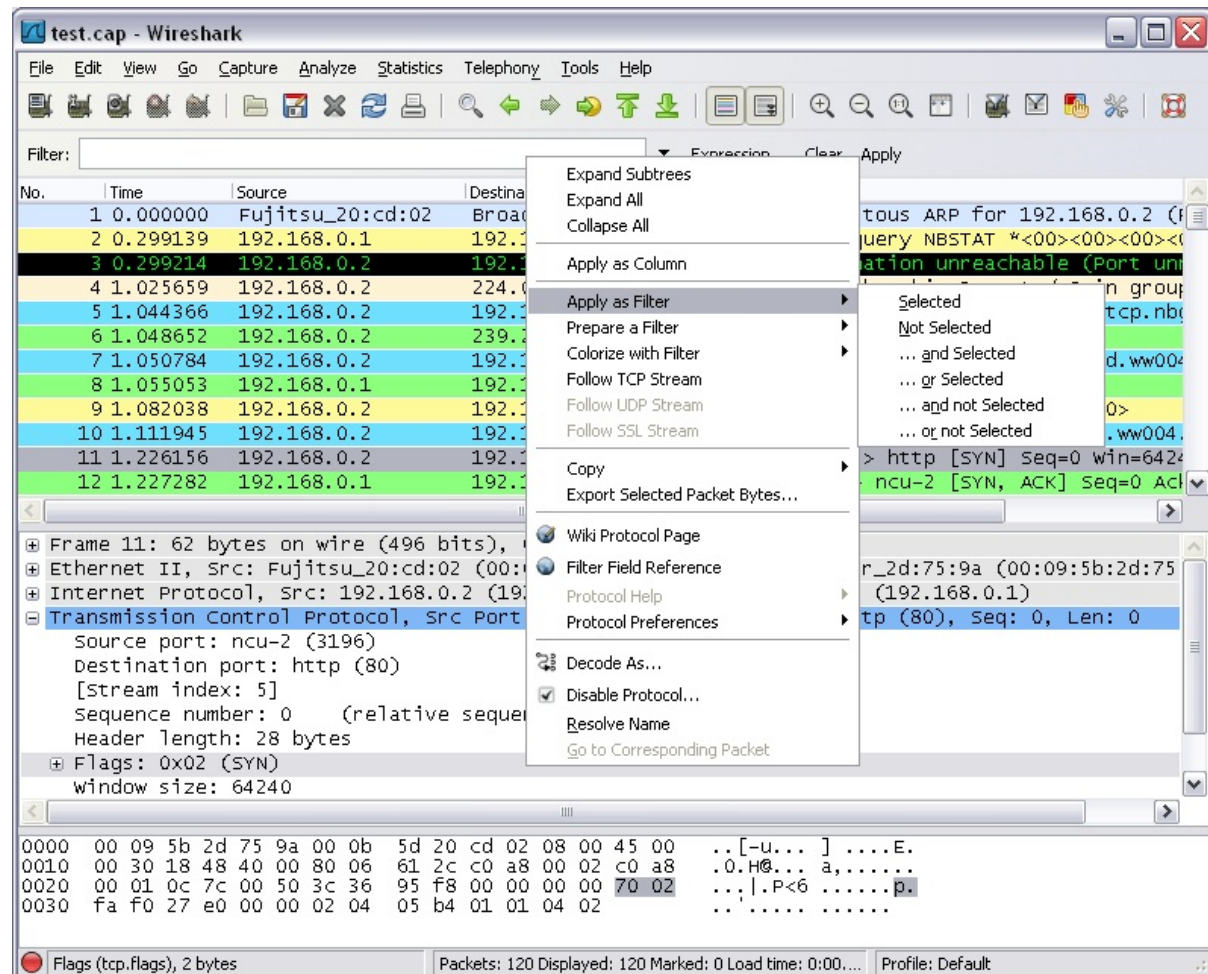
Table 6.2. The menu items of the “Packet List” pop-up menu

Item	Identical to main menu's item:	Description
Mark Packet (toggle)	Edit	Mark/unmark a packet.
Ignore Packet (toggle)	Edit	Ignore or inspect this packet while dissecting the capture file.
Set Time		

Reference (toggle)	Edit	Set/reset a time reference.
Manually Resolve Address		Allows you to enter a name to resolve for the selected address.
Apply as Filter	Analyze	Prepare and apply a display filter based on the currently selected item.
Prepare a Filter	Analyze	Prepare a display filter based on the currently selected item.
Conversation Filter		This menu item applies a display filter with the address information from the selected packet. E.g. the IP menu entry will set a filter to show the traffic between the two IP addresses of the current packet. XXX - add a new section describing this better.
Colorize Conversation		This menu item uses a display filter with the address information from the selected packet to build a new colorizing rule.
SCTP		Allows you to analyze and prepare a filter for this SCTP association.
Follow TCP Stream	Analyze	Allows you to view all the data on a TCP stream between a pair of nodes.
Follow UDP Stream	Analyze	Allows you to view all the data on a UDP datagram stream between a pair of nodes.
Follow SSL Stream	Analyze	Same as "Follow TCP Stream" but for SSL. XXX - add a new section describing this better.
Copy/ Summary (Text)		Copy the summary fields as displayed to the clipboard, as tab-separated text.
Copy/ Summary (CSV)		Copy the summary fields as displayed to the clipboard, as comma-separated text.
Copy/ As Filter		Prepare a display filter based on the currently selected item and copy that filter to the clipboard.
Copy/ Bytes (Offset Hex Text)		Copy the packet bytes to the clipboard in hexdump-like format.
Copy/ Bytes (Offset Hex)		Copy the packet bytes to the clipboard in hexdump-like format, but without the text portion.
Copy/ Bytes (Printable Text Only)		Copy the packet bytes to the clipboard as ASCII text, excluding non-printable characters.
Copy/ Bytes (Hex Stream)		Copy the packet bytes to the clipboard as an unpunctuated list of hex digits.
Copy/ Bytes (Binary Stream)		Copy the packet bytes to the clipboard as raw binary. The data is stored in the clipboard as MIME-type "application/octet-stream".
Decode As...	Analyze	Change or apply a new relation between two dissectors.
Print...	File	Print packets.
Show Packet in New Window	View	Display the selected packet in a new window.

2.3. Pop-up menu of the “Packet Details” pane

Figure 6.5. Pop-up menu of the “Packet Details” pane



The following table gives an overview of which functions are available in this pane, where to find the corresponding function in the main menu, and a short description of each item.

Table 6.3. The menu items of the “Packet Details” pop-up menu

Item	Identical to main menu's item:	Description
Expand Subtrees	View	Expand the currently selected subtree.
Collapse Subtrees	View	Collapse the currently selected subtree.
Expand All	View	Expand all subtrees in all packets in the capture.
Collapse All	View	Wireshark keeps a list of all the protocol subtrees that are expanded, and uses it to ensure that the correct subtrees are expanded when you display a packet. This menu item collapses the tree view of all packets in the capture list.
Apply as Column		Use the selected protocol item to create a new column in the packet list.
Apply as Filter	Analyze	Prepare and apply a display filter based on the currently selected item.

Prepare a Filter	Analyze	Prepare a display filter based on the currently selected item.
Colorize with Filter		This menu item uses a display filter with the information from the selected protocol item to build a new colorizing rule.
Follow TCP Stream	Analyze	Allows you to view all the data on a TCP stream between a pair of nodes.
Follow UDP Stream	Analyze	Allows you to view all the data on a UDP datagram stream between a pair of nodes.
Follow SSL Stream	Analyze	Same as “Follow TCP Stream” but for SSL. XXX - add a new section describing this better.
Copy/ Description	Edit	Copy the displayed text of the selected field to the system clipboard.
Copy/ Fieldname	Edit	Copy the name of the selected field to the system clipboard.
Copy/ Value	Edit	Copy the value of the selected field to the system clipboard.
Copy/ As Filter	Edit	Prepare a display filter based on the currently selected item and copy it to the clipboard.
Copy/ Bytes (Offset Hex Text)		Copy the packet bytes to the clipboard in hexdump-like format; similar to the Packet List Pane command, but copies only the bytes relevant to the selected part of the tree (the bytes selected in the Packet Bytes Pane).
Copy/ Bytes (Offset Hex)		Copy the packet bytes to the clipboard in hexdump-like format, but without the text portion; similar to the Packet List Pane command, but copies only the bytes relevant to the selected part of the tree (the bytes selected in the Packet Bytes Pane).
Copy/ Bytes (Printable Text Only)		Copy the packet bytes to the clipboard as ASCII text, excluding non-printable characters; similar to the Packet List Pane command, but copies only the bytes relevant to the selected part of the tree (the bytes selected in the Packet Bytes Pane).
Copy/ Bytes (Hex Stream)		Copy the packet bytes to the clipboard as an unpunctuated list of hex digits; similar to the Packet List Pane command, but copies only the bytes relevant to the selected part of the tree (the bytes selected in the Packet Bytes Pane).
Copy/ Bytes (Binary Stream)		Copy the packet bytes to the clipboard as raw binary; similar to the Packet List Pane command, but copies only the bytes relevant to the selected part of the tree (the bytes selected in the Packet Bytes Pane). The data is stored in the clipboard as MIME-type “application/octet-stream”.
Export Selected Packet Bytes...	File	This menu item is the same as the File menu item of the same name. It allows you to export raw packet bytes to a binary file.
Wiki Protocol Page		Show the wiki page corresponding to the currently selected protocol in your web browser.
Filter Field Reference		Show the filter field reference web page corresponding to the currently selected protocol in your web browser.
Protocol Preferences...		The menu item takes you to the properties dialog and selects the page corresponding to the protocol if there are properties associated with the highlighted field. More information on preferences can be found in Figure 10.7, “The preferences dialog box” .
Decode As...	Analyze	Change or apply a new relation between two dissectors.
Disable Protocol		Allows you to temporarily disable a protocol dissector, which may be blocking the legitimate dissector.

Resolve Name	View	Causes a name resolution to be performed for the selected packet, but NOT every packet in the capture.
Go to Corresponding Packet	Go	If the selected field has a corresponding packet, go to it. Corresponding packets will usually be a request/response packet pair or such.

3. Filtering packets while viewing

Wireshark has two filtering languages: One used when capturing packets, and one used when displaying packets. In this section we explore that second type of filter: Display filters. The first one has already been dealt with in [Section 4.13, “Filtering while capturing”](#).

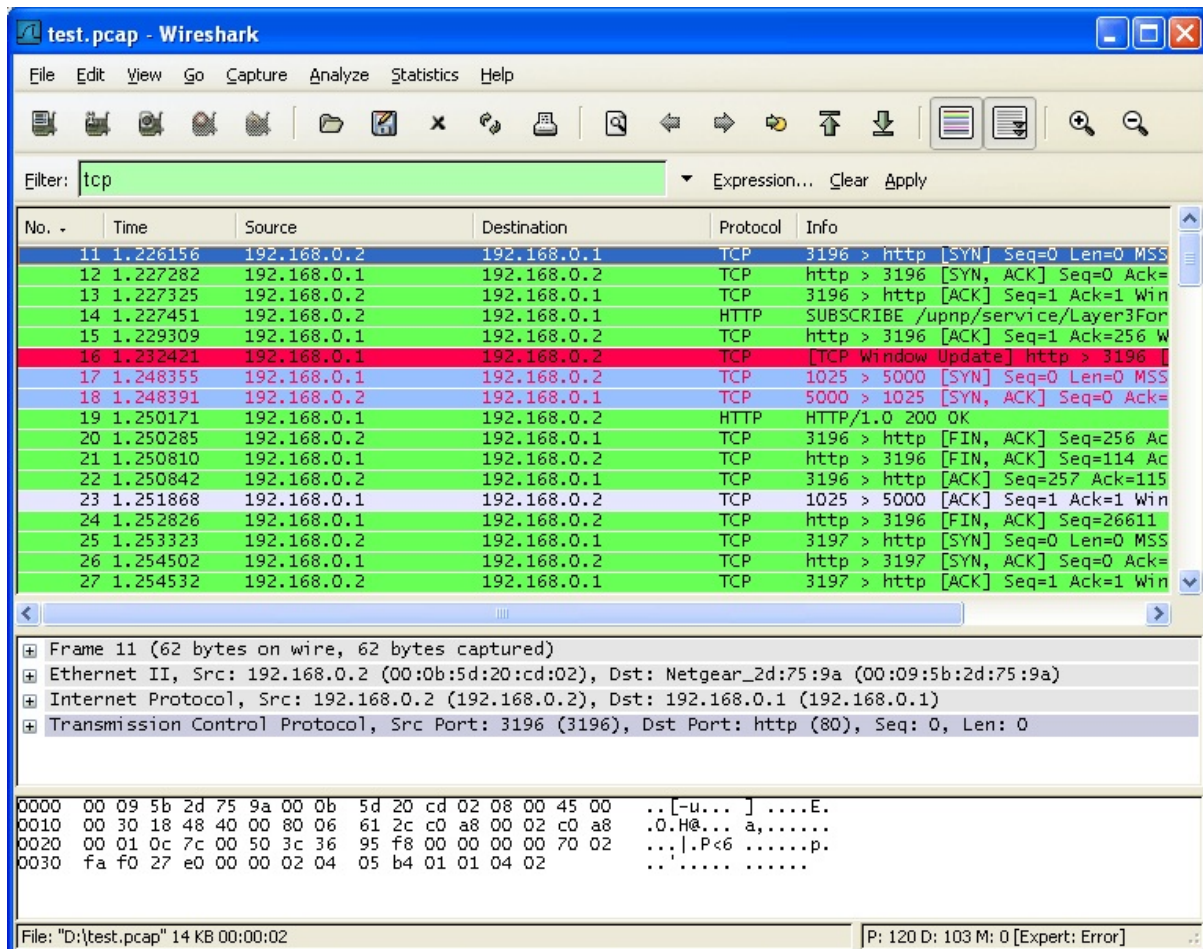
Display filters allow you to concentrate on the packets you are interested in while hiding the currently uninteresting ones. They allow you to select packets by:

- Protocol
- The presence of a field
- The values of fields
- A comparison between fields
- ... and a lot more!

To select packets based on protocol type, simply type the protocol in which you are interested in the *Filter:* field in the filter toolbar of the Wireshark window and press enter to initiate the filter. [Figure 6.6, “Filtering on the TCP protocol”](#) shows an example of what happens when you type *tcp* in the filter field.

All protocol and field names are entered in lowercase. Also, don't forget to press enter after entering the filter expression.

Figure 6.6. Filtering on the TCP protocol



As you might have noticed, only packets of the TCP protocol are displayed now (e.g. packets 1-10 are hidden). The packet numbering will remain as before, so the first packet shown is now packet number 11.

When using a display filter, all packets remain in the capture file. The display filter only changes the display of the capture file but not its content!

You can filter on any protocol that Wireshark understands. You can also filter on any field that a dissector adds to the tree view, but only if the dissector has added an abbreviation for the field. A list of such fields is available in Wireshark in the *Add Expression...* dialog box. You can find more information on the *Add Expression...* dialog box in [Section 6.5](#), “The “Filter Expression” dialog box”.

For example, to narrow the packet list pane down to only those packets to or from the IP address 192.168.0.1, use `ip.addr==192.168.0.1`.

To remove the filter, click on the Clear button to the right of the filter field.

4. Building display filter expressions

Wireshark provides a simple but powerful display filter language that allows you to build quite complex filter expressions. You can compare values in packets as well as combine expressions into more specific expressions. The following sections provide more information on doing this.

You will find a lot of Display Filter examples at the *Wireshark Wiki Display Filter page* at: <https://wiki.wireshark.org/DisplayFilters>.

4.1. Display filter fields

Every field in the packet details pane can be used as a filter string, this will result in showing only the packets where this field exists. For example: the filter string: `tcp` will show all packets containing the tcp protocol.

There is a complete list of all filter fields available through the menu item Help → Supported Protocols in the page “Display Filter Fields” of the “Supported Protocols” dialog.

4.2. Comparing values

You can build display filters that compare values using a number of different comparison operators. They are shown in Table 6.4, “Display Filter comparison operators”.

You can use English and C-like terms in the same way, they can even be mixed in a filter string.

Table 6.4. Display Filter comparison operators

English	C-like	Description and example		
eq	==	Equal. <code>ip.src==10.0.0.5</code>		
ne	!=	Not equal. <code>ip.src!=10.0.0.5</code>		
gt	>	Greater than. <code>frame.len > 10</code>		
lt	<	Less than. <code>frame.len < 128</code>		
ge	>=	Greater than or equal to. <code>frame.len ge 0x100</code>		
le	<=	Less than or equal to. <code>frame.len <= 0x20</code>		
contains		Protocol, field or slice contains a value. <code>sip.To contains "a1762"</code>		
matches	~	Protocol or text field match Perl regular expression. <code>`http.host matches "acme.(org</code>	com	net)""
bitwise_and	&	Compare bit field value. <code>tcp.flags & 0x02</code>		

In addition, all protocol fields have a type. [Display Filter Field Types](#) provides a list of the types and example of how to express them.

Display Filter Field Types

Unsigned integer

Can be 8, 16, 24, 32, or 64 bits. You can express integers in decimal, octal, or hexadecimal. The following display filters are equivalent:

```
ip.len le 1500
```

```
ip.len le 02734
```

```
ip.len le 0x436
```

Signed integer

Can be 8, 16, 24, 32, or 64 bits. As with unsigned integers you can use decimal, octal, or hexadecimal.

Boolean

A boolean field is present in the protocol decode only if its value is true. For example, `tcp.flags.syn` is present, and thus true, only if the SYN flag is present in a TCP segment header.

```
The filter expression +tcp.flags.syn+ will select only those packets for which
this flag exists, that is, TCP segments where the segment header contains the
SYN flag. Similarly, to find source-routed token ring packets, use a filter
expression of +tr.sr+.
```

Ethernet address

6 bytes separated by a colon (:), dot (.) or dash (-) with one or two bytes between separators:

```
eth.dst == ff:ff:ff:ff:ff:ff
```

```
eth.dst == ff-ff-ff-ff-ff-ff
```

```
eth.dst == ffff.ffff.ffff
```

IPv4 address

```
ip.addr == 192.168.0.1
```

```
Classless InterDomain Routing (CIDR) notation can be used to test if
an IPv4 address is in a certain subnet. For example, this display
filter will find all packets in the 129.111 Class-B network:
```

```
ip.addr == 129.111.0.0/16
```

IPv6 address

```
ipv6.addr == ::1
```

```
As with IPv4 addresses, IPv6 addresses can match a subnet.
```

Text string

```
http.request.uri == "https://www.wireshark.org/"
```

```
udp contains 81:60:03
```

The example above match packets that contains the 3-byte sequence 0x81, 0x60, 0x03 anywhere in the UDP header or payload.

```
sip.To contains "a1762"
```

Above example match packets where SIP To-header contains the string "a1762" anywhere in the header.

```
http.host matches "acme\.(org|com|net)"
```

The example above match HTTP packets where the HOST header contains acme.org or acme.com or acme.net.
Note: Wireshark needs to be built with libpcr in order to be able to use the `matches` resp. `~` operator.

```
tcp.flags & 0x02
```

That expression will match all packets that contain a "tcp.flags" field with the 0x02 bit, i.e. the SYN bit, set.

4.3. Combining expressions

You can combine filter expressions in Wireshark using the logical operators shown in [Table 6.5, "Display Filter Logical Operations"](#)

Table 6.5. Display Filter Logical Operations

English	C-like	Description and example
and	&&	Logical AND. <code>ip.src==10.0.0.5 and tcp.flags.fin</code>
or		Logical OR. <code>ip.scr==10.0.0.5 or ip.src==192.1.1.1</code>
xor	<code>^</code>	Logical XOR. <code>tr.dst[0:3] == 0.6.29 xor tr.src[0:3] == 0.6.29</code>
not	!	Logical NOT. <code>not llc</code>
[...]		See "Substring Operator" below.
in		See "Membership Operator" below.

6.4.4. Substring Operator

Wireshark allows you to select subsequences of a sequence in rather elaborate ways. After a label you can place a pair of brackets `[]` containing a comma separated list of range specifiers.

```
eth.src[0:3] == 00:00:83
```

The example above uses the n:m format to specify a single range. In this case n is the beginning offset and m is the length of the range being specified.

```
eth.src[1-2] == 00:83
```

The example above uses the n-m format to specify a single range. In this case n is the beginning offset and m is the ending offset.

```
eth.src[:4] == 00:00:83:00
```


The example above uses the `:m` format, which takes everything from the beginning of a sequence to offset `m`. It is equivalent to `0:m`

```
eth.src[4:] == 20:20
```

The example above uses the `n:` format, which takes everything from offset `n` to the end of the sequence.

```
eth.src[2] == 83
```

The example above uses the `n` format to specify a single range. In this case the element in the sequence at offset `n` is selected. This is equivalent to `n:1`.

```
eth.src[0:3,1-2,:4,4:,2] ==  
00:00:83:00:83:00:00:83:00:20:20:83
```

Wireshark allows you to string together single ranges in a comma separated list to form compound ranges as shown above.

4.5. Membership Operator.

Wireshark allows you to test a field for membership in a set of values or fields. After the field name, use the `in` operator followed by the set items surrounded by braces `{}`.

```
tcp.port in {80 443 8080}
```

This can be considered a shortcut operator, as the previous expression could have been expressed as:

```
tcp.port == 80 || tcp.port == 443 || tcp.port == 8080
```

4.6. A Common Mistake

Using the `!=` operator on combined expressions like `eth.addr`, `ip.addr`, `tcp.port`, and `udp.port` will probably not work as expected.

Often people use a filter string to display something like `ip.addr == 1.2.3.4` which will display all packets containing the IP address 1.2.3.4.

Then they use `ip.addr != 1.2.3.4` to see all packets not containing the IP address 1.2.3.4 in it. Unfortunately, this does *not* do the expected.

Instead, that expression will even be true for packets where either source or destination IP address equals 1.2.3.4. The reason for this, is that the expression `ip.addr != 1.2.3.4` must be read as “the packet contains a field named `ip.addr` with a value different from 1.2.3.4”. As an IP datagram contains both a source and a destination address, the expression will evaluate to true whenever at least one of the two addresses differs from 1.2.3.4.

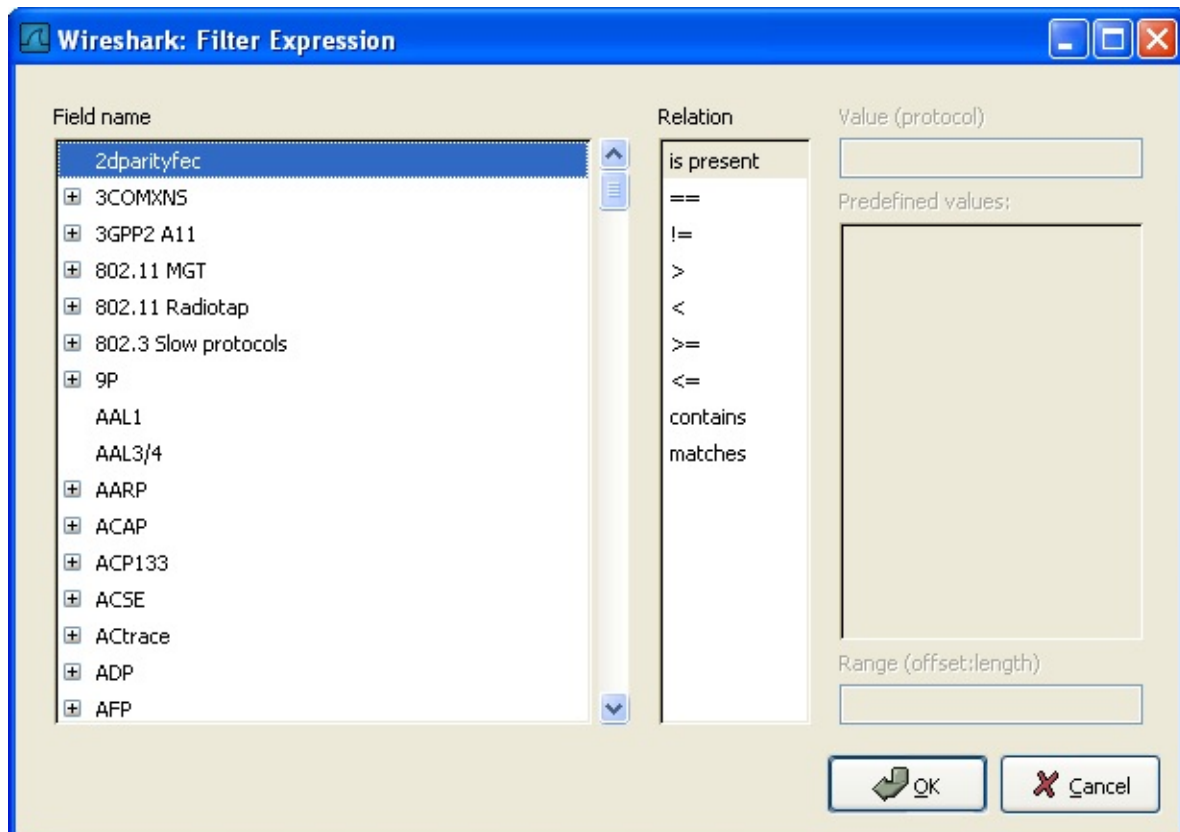
If you want to filter out all packets containing IP datagrams to or from IP address 1.2.3.4, then the correct filter is `!(ip.addr == 1.2.3.4)` as it reads “show me all the packets for which it is not true that a field named `ip.addr` exists with a value of 1.2.3.4”, or in other words, “filter out all packets for which there are no occurrences of a field named `ip.addr` with the value 1.2.3.4”.

5. The “Filter Expression” dialog box

When you are accustomed to Wireshark's filtering system and know what labels you wish to use in your filters it can be very quick to simply type a filter string. However if you are new to Wireshark or are working with a slightly unfamiliar protocol it can be very confusing to try to figure out what to type. The "Filter Expression" dialog box helps with this.

The "Filter Expression" dialog box is an excellent way to learn how to write Wireshark display filter strings.

Figure 6.7. The "Filter Expression" dialog box



When you first bring up the Filter Expression dialog box you are shown a tree of field names, organized by protocol, and a box for selecting a relation.

Field Name

Select a protocol field from the protocol field tree. Every protocol with filterable fields is listed at the top level. (You can search for a particular protocol entry by entering the first few letters of the protocol name). By expanding a protocol name you can get a list of the field names available for filtering for that protocol.

Relation

Select a relation from the list of available relation. The *is present* is a unary relation which is true if the selected field is present in a packet. All other listed relations are binary relations which require additional data (e.g. a *Value* to match) to complete.

When you select a field from the field name list and select a binary relation (such as the equality relation `==`) you will be given the opportunity to enter a value, and possibly some range information.

Value

You may enter an appropriate value in the *Value* text box. The *Value* will also indicate the type of value for the *field name* you have selected (like character string).

Predefined values

Some of the protocol fields have predefined values available, much like enum's in C. If the selected protocol field has such values defined, you can choose one of them here.

Range

A range of integers or a group of ranges, such as `1-12` or `39-42, 98-2000`.

OK

When you have built a satisfactory expression click OK and a filter string will be built for you.

Cancel

You can leave the “Add Expression...” dialog box without any effect by clicking the Cancel button.

6. Defining and saving filters

You can define filters with Wireshark and give them labels for later use. This can save time in remembering and retyping some of the more complex filters you use.

To define a new filter or edit an existing one, select Capture → Capture Filters... or Analyze → Display Filters....

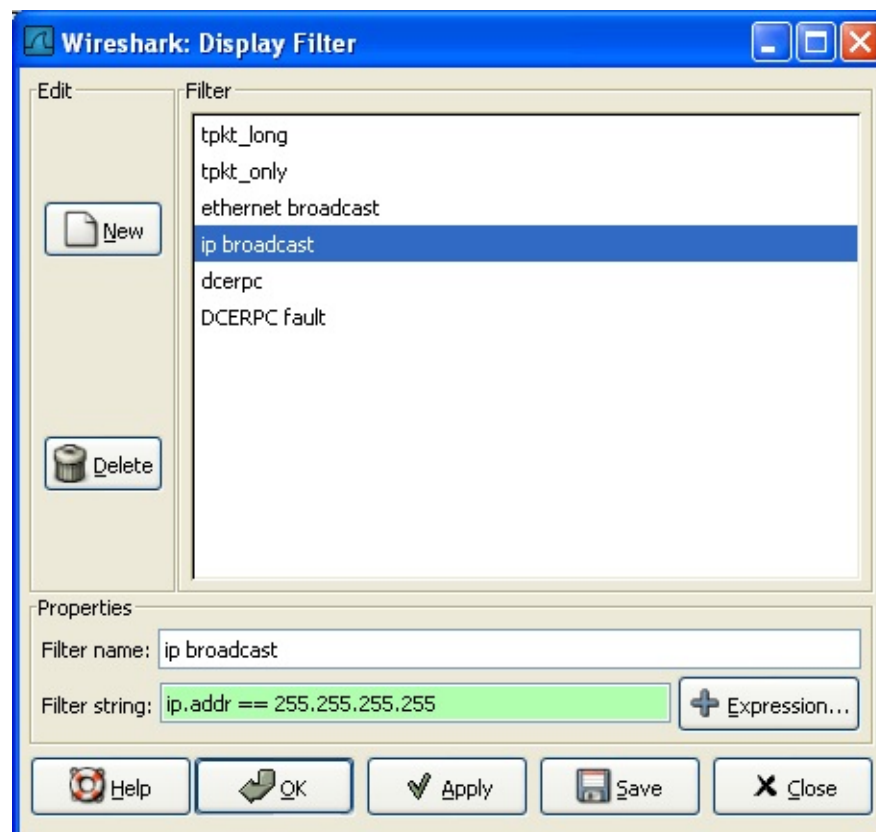
Wireshark will then pop up the Filters dialog as shown in [Figure 6.8](#), “The “Capture Filters” and “Display Filters” dialog boxes”.

The mechanisms for defining and saving capture filters and display filters are almost identical. Both will be described here but the differences between these two will be marked as such.

Warning

You must use Save to save your filters permanently. OK or Apply will not save the filters and they will be lost when you close Wireshark.

Figure 6.8. The “Capture Filters” and “Display Filters” dialog boxes



New

This button adds a new filter to the list of filters. The currently entered values from Filter name and Filter string will be used. If any of these fields are empty, it will be set to “new”.

Delete

This button deletes the selected filter. It will be greyed out, if no filter is selected.

Filter

You can select a filter from this list (which will fill in the filter name and filter string in the fields down at the bottom of the dialog box).

Filter name:

You can change the name of the currently selected filter here.

The filter name will only be used in this dialog to identify the filter for your convenience, it will not be used elsewhere. You can add multiple filters with the same name, but this is not very useful.

Filter string:

You can change the filter string of the currently selected filter here. Display Filter only: the string will be syntax checked while you are typing.

Add Expression...

Display Filter only: This button brings up the Add Expression dialog box which assists in building filter strings. You can find more information about the Add Expression dialog in [Section 6.5, “The “Filter Expression” dialog box”](#)

OK

Display Filter only: This button applies the selected filter to the current display and closes the dialog.

Apply

Display Filter only: This button applies the selected filter to the current display, and keeps the dialog open.

Save

Save the current settings in this dialog. The file location and format is explained in [Appendix B, Files and Folders](#).

Close

Close this dialog. This will discard unsaved settings.

7. Defining and saving filter macros

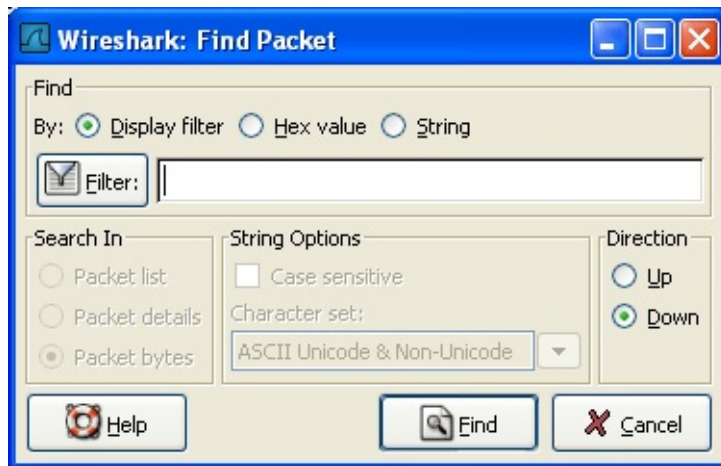
You can define filter macros with Wireshark and give them labels for later use. This can save time in remembering and retying some of the more complex filters you use.

8. Finding packets

You can easily find packets once you have captured some packets or have read in a previously saved capture file. Simply select the *Find Packet...* menu item from the *Edit* menu. Wireshark will pop up the dialog box shown in [Figure 6.9, “The “Find Packet” dialog box”](#).

8.1. The “Find Packet” dialog box

Figure 6.9. The “Find Packet” dialog box



You might first select the kind of thing to search for:

- *Display filter*

Simply enter a display filter string into the *Filter:* field, select a direction, and click on OK.

For example, to find the three way handshake for a connection from host 192.168.0.1, use the following filter string:

```
ip.src==192.168.0.1 and tcp.flags.syn==1
```

For more details on display filters, see [Section 6.3, “Filtering packets while viewing”](#)

- *Hex Value*

Search for a specific byte sequence in the packet data.

For example, use “00:00” to find the next packet including two null bytes in the packet data.

- *String*

Find a string in the packet data, with various options.

The value to be found will be syntax checked while you type it in. If the syntax check of your value succeeds, the background of the entry field will turn green, if it fails, it will turn red.

You can choose the search direction:

- *Up*

Search upwards in the packet list (decreasing packet numbers).

- *Down*

Search downwards in the packet list (increasing packet numbers).

8.2. The “Find Next” command

“Find Next” will continue searching with the same options used in the last “Find Packet”.

8.3. The “Find Previous” command

“Find Previous” will do the same thing as “Find Next”, but in the reverse direction.

9. Go to a specific packet

You can easily jump to specific packets with one of the menu items in the Go menu.

6.9.1. The “Go Back” command

Go back in the packet history, works much like the page history in current web browsers.

6.9.2. The “Go Forward” command

Go forward in the packet history, works much like the page history in current web browsers.

9.3. The “Go to Packet” dialog box

Figure 6.10. The “Go To Packet” dialog box



This dialog box will let you enter a packet number. When you press OK, Wireshark will jump to that packet.

9.4. The “Go to Corresponding Packet” command

If a protocol field is selected which points to another packet in the capture file, this command will jump to that packet.

As these protocol fields now work like links (just as in your Web browser), it's easier to simply double-click on the field to jump to the corresponding field.

9.5. The “Go to First Packet” command

This command will simply jump to the first packet displayed.

9.6. The “Go to Last Packet” command

This command will simply jump to the last packet displayed.

10. Marking packets

You can mark packets in the “Packet List” pane. A marked packet will be shown with black background, regardless of the coloring rules set. Marking a packet can be useful to find it later while analyzing in a large capture file.

The packet marks are not stored in the capture file or anywhere else. All packet marks will be lost when you close the capture file.

You can use packet marking to control the output of packets when saving, exporting, or printing. To do so, an option in the packet range is available, see [Section 5.9, “The “Packet Range” frame”](#).

There are three functions to manipulate the marked state of a packet:

- *Mark packet (toggle)* toggles the marked state of a single packet.
- *Mark all displayed packets* set the mark state of all displayed packets.

- *Unmark all packets* reset the mark state of all packets.

These mark functions are available from the “Edit” menu, and the “Mark packet (toggle)” function is also available from the pop-up menu of the “Packet List” pane.

11. Ignoring packets

You can ignore packets in the “Packet List” pane. Wireshark will then pretend that this packets does not exist in the capture file. An ignored packet will be shown with white background and gray foreground, regardless of the coloring rules set.

The packet ignored marks are not stored in the capture file or anywhere else. All “packet ignored” marks will be lost when you close the capture file.

There are three functions to manipulate the ignored state of a packet:

- *Ignore packet (toggle)* toggles the ignored state of a single packet.
- *Ignore all displayed packets* set the ignored state of all displayed packets.
- *Un-Ignore all packets* reset the ignored state of all packets.

These ignore functions are available from the “Edit” menu, and the “Ignore packet (toggle)” function is also available from the pop-up menu of the “Packet List” pane.

12. Time display formats and time references

While packets are captured, each packet is timestamped. These timestamps will be saved to the capture file, so they will be available for later analysis.

A detailed description of timestamps, timezones and alike can be found at: [Section 7.5, “Time Stamps”](#).

The timestamp presentation format and the precision in the packet list can be chosen using the View menu, see [Figure 3.5, “The “View” Menu”](#).

The available presentation formats are:

- *Date and Time of Day: 1970-01-01 01:02:03.123456* The absolute date and time of the day when the packet was captured.
- *Time of Day: 01:02:03.123456* The absolute time of the day when the packet was captured.
- *Seconds Since Beginning of Capture: 123.123456* The time relative to the start of the capture file or the first “Time Reference” before this packet (see [Section 6.12.1, “Packet time referencing”](#)).
- *Seconds Since Previous Captured Packet: 1.123456* The time relative to the previous captured packet.
- *Seconds Since Previous Displayed Packet: 1.123456* The time relative to the previous displayed packet.
- *Seconds Since Epoch (1970-01-01): 1234567890.123456* The time relative to epoch (midnight UTC of January 1, 1970).

The available precisions (aka. the number of displayed decimal places) are:

- *Automatic* The timestamp precision of the loaded capture file format will be used (the default).
- *Seconds, Deciseconds, Centiseconds, Milliseconds, Microseconds or Nanoseconds* The timestamp precision will be forced to the given setting. If the actually available precision is smaller, zeros will be appended. If the precision is larger, the remaining decimal places will be cut off.

Precision example: If you have a timestamp and it's displayed using, “Seconds Since Previous Packet”, : the value might be 1.123456. This will be displayed using the “Automatic” setting for libpcap files (which is microseconds). If you use Seconds it would show simply 1 and if you use Nanoseconds it shows 1.123456000.

12.1. Packet time referencing

The user can set time references to packets. A time reference is the starting point for all subsequent packet time calculations. It will be useful, if you want to see the time values relative to a special packet, e.g. the start of a new request. It's possible to set multiple time references in the capture file.

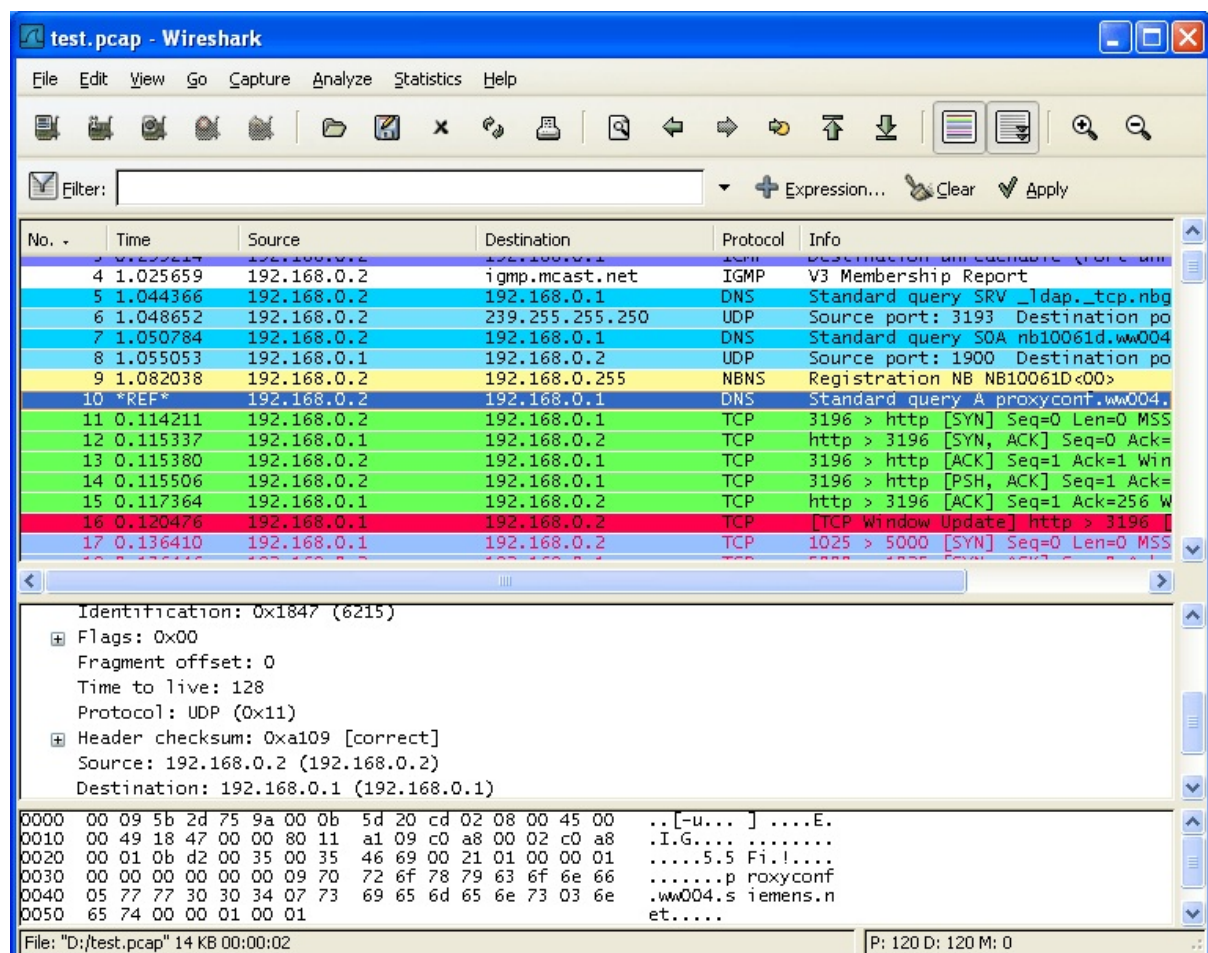
The time references will not be saved permanently and will be lost when you close the capture file.

Time referencing will only be useful if the time display format is set to "Seconds Since Beginning of Capture". If one of the other time display formats are used, time referencing will have no effect (and will make no sense either).

To work with time references, choose one of the Time Reference items in the Edit menu or from the pop-up menu of the "Packet List" pane. See [Section 3.6, "The "Edit" menu"](#).

- *Set Time Reference (toggle)* Toggles the time reference state of the currently selected packet to on or off.
- *Find Next* Find the next time referenced packet in the "Packet List" pane.
- *Find Previous* Find the previous time referenced packet in the "Packet List" pane.

Figure 6.11. Wireshark showing a time referenced packet



A time referenced packet will be marked with the string *REF* in the Time column (see packet number 10). All subsequent packets will show the time since the last time reference.

Sources

- https://www.wireshark.org/docs/wsug_html/#ChapterWork

Statistiques Wireshark

- [1. Introduction](#)
- [2. The “Summary” window](#)
- [3. The “Protocol Hierarchy” window](#)
- [4. Conversations](#)
 - [4.1. The “Conversations” window](#)
- [5. Endpoints](#)
 - [5.1. The “Endpoints” window](#)
- [6. The “IO Graphs” window](#)
- [7. Service Response Time](#)
 - [7.1. The “Service Response Time DCE-RPC” window](#)
- [8. Compare two capture files](#)
- [9. WLAN Traffic Statistics](#)
- [10. The protocol specific statistics windows](#)
- [Sources](#)

Texte en anglais

1. Introduction

Wireshark provides a wide range of network statistics which can be accessed via the Statistics menu.

These statistics range from general information about the loaded capture file (like the number of captured packets), to statistics about specific protocols (e.g. statistics about the number of HTTP requests and responses captured).

- General statistics:
 - **Summary** about the capture file.
 - **Protocol Hierarchy** of the captured packets.
 - **Conversations** e.g. traffic between specific IP addresses.
 - **Endpoints** e.g. traffic to and from an IP addresses.
 - **IO Graphs** visualizing the number of packets (or similar) in time.
- Protocol specific statistics:
 - **Service Response Time** between request and response of some protocols.
 - Various other protocol specific statistics.

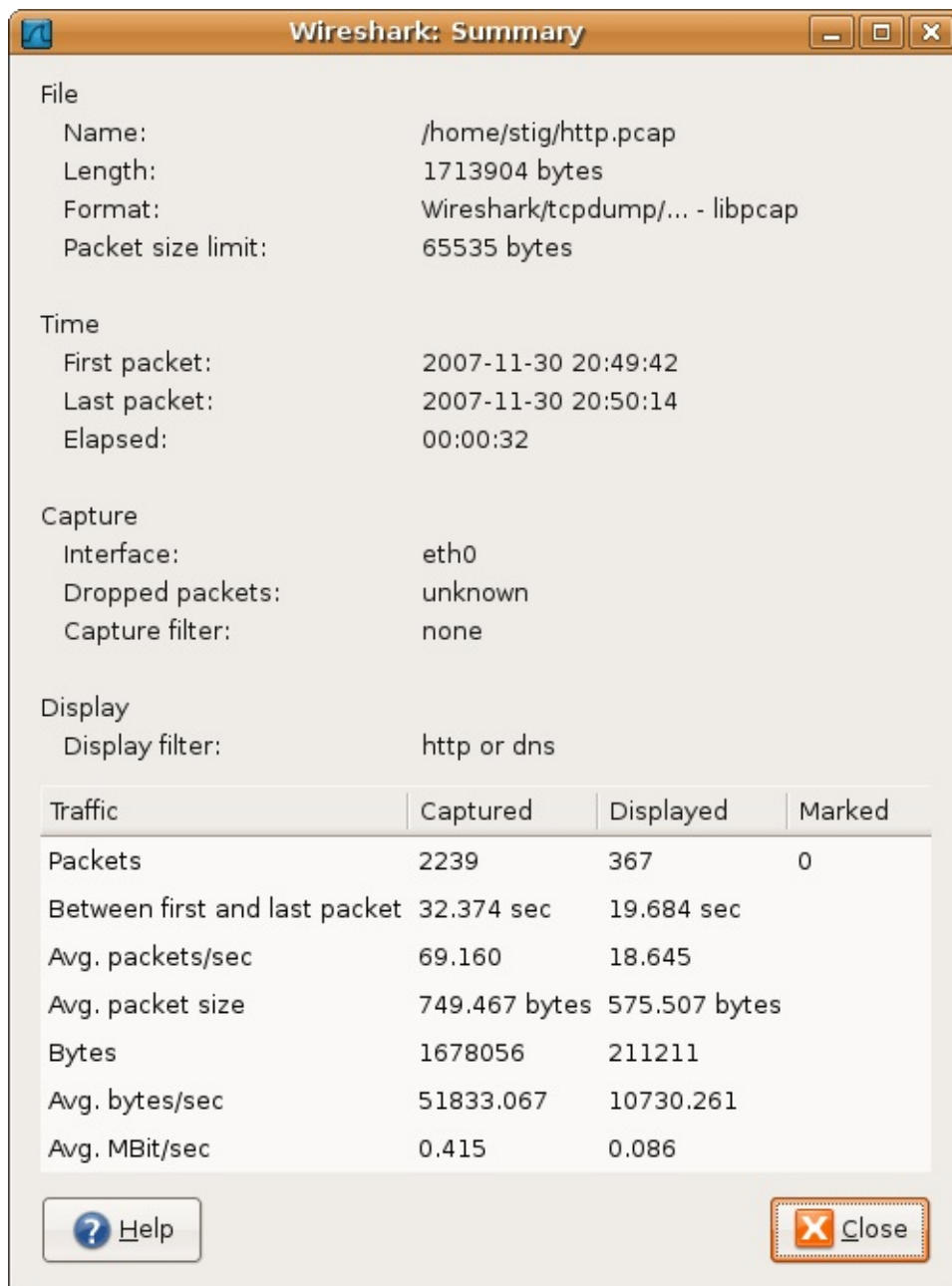
Note

The protocol specific statistics require detailed knowledge about the specific protocol. Unless you are familiar with that protocol, statistics about it will be pretty hard to understand.

2. The “Summary” window

General statistics about the current capture file.

Figure 8.1. The “Summary” window



- *File*: general information about the capture file.
- *Time*: the timestamps when the first and the last packet were captured (and the time between them).
- *Capture*: information from the time when the capture was done (only available if the packet data was captured from the network and not loaded from a file).
- *Display*: some display related information.
- *Traffic*: some statistics of the network traffic seen. If a display filter is set, you will see values in the *Captured* column, and if any packages are marked, you will see values in the *Marked* column. The values in the *Captured* column will remain the same as before, while the values in the *Displayed* column will reflect the values corresponding to the packets shown in the display. The values in the *Marked* column will reflect the values corresponding to the marked packages.

3. The “Protocol Hierarchy” window

The protocol hierarchy of the captured packets.

Figure 8.2. The “Protocol Hierarchy” window

This is a tree of all the protocols in the capture. Each row contains the statistical values of one protocol. Two of the columns (*Percent Packets* and *Percent Bytes*) serve double duty as bar graphs. If a display filter is set it will be shown at the bottom.

The Copy button will let you copy the window contents as CSV or YAML.

Protocol hierarchy columns

Protocol

This protocol's name

Percent Packets

The percentage of protocol packets relative to all packets in the capture

Packets

The total number of packets of this protocol

Percent Bytes

The percentage of protocol bytes relative to the total bytes in the capture

Bytes

The total number of bytes of this protocol

Bits/s

The bandwidth of this protocol relative to the capture time

End Packets

The absolute number of packets of this protocol where it was the highest protocol in the stack (last dissected)

End Bytes

The absolute number of bytes of this protocol where it was the highest protocol in the stack (last dissected)

End Bits/s

The bandwidth of this protocol relative to the capture time where was the highest protocol in the stack (last dissected)

Packets usually contain multiple protocols. As a result more than one protocol will be counted for each packet.

Example: In the screenshot IP has 99.9% and TCP 98.5% (which is together much more than 100%).

Protocol layers can consist of packets that won't contain any higher layer protocol, so the sum of all higher layer packets may not sum up to the protocols packet count. Example: In the screenshot TCP has 98.5% but the sum of the subprotocols (SSL, HTTP, etc) is much less. This can be caused by continuation frames, TCP protocol overhead, and other undissected data.

A single packet can contain the same protocol more than once. In this case, the protocol is counted more than once. For example ICMP replies and many tunneling protocols will carry more than one IP header.

4. Conversations

A network conversation is the traffic between two specific endpoints. For example, an IP conversation is all the traffic between two IP addresses. The description of the known endpoint types can be found in [Section 8.5, “Endpoints”](#).

4.1. The “Conversations” window

The conversations window is similar to the endpoint Window. See [Section 8.5.1, “The “Endpoints” window”](#) for a description of their common features. Along with addresses, packet counters, and byte counters the conversation window adds four columns: the start time of the conversation (“Rel Start”) or (“Abs Start”), the duration of the conversation in seconds, and the average bits (not bytes) per second in each direction. A timeline graph is also drawn across the “Rel Start” / “Abs Start” and “Duration” columns.

Figure 8.3. The “Conversations” window



Each row in the list shows the statistical values for exactly one conversation.

Name resolution will be done if selected in the window and if it is active for the specific protocol layer (MAC layer for the selected Ethernet endpoints page). *Limit to display filter* will only show conversations matching the current display filter. *Absolute start time* switches the start time column between relative (“Rel Start”) and absolute (“Abs Start”) times. Relative start times match the “Seconds Since Beginning of Capture” time display format in the packet list and absolute start times match the “Time of Day” display format.

The Copy button will copy the list values to the clipboard in CSV (Comma Separated Values) or YAML format. The Follow Stream... button will show the stream contents as described in [Figure 7.1, “The “Follow TCP Stream” dialog box”](#) dialog. The Graph... button will show a graph as described in [Section 8.6, “The “IO Graphs” window”](#).

Conversation Types lets you choose which traffic type tabs are shown. See [Section 8.5, “Endpoints”](#) for a list of endpoint types. The enabled types are saved in your profile settings.

Tip

This window will be updated frequently so it will be useful even if you open it before (or while) you are doing a live capture.

5. Endpoints

A network endpoint is the logical endpoint of separate protocol traffic of a specific protocol layer. The endpoint statistics of Wireshark will take the following endpoints into account:

Tip

If you are looking for a feature other network tools call a *hostlist*, here is the right place to look. The list of Ethernet or IP endpoints is usually what you’re looking for.

Endpoint and Conversation types

Bluetooth

A MAC-48 address similar to Ethernet.

Ethernet

Identical to the Ethernet device’s MAC-48 identifier.

Fibre Channel

A MAC-48 address similar to Ethernet.

IEEE 802.11

A MAC-48 address similar to Ethernet.

FDDI

Identical to the FDDI MAC-48 address.

IPv4

Identical to the 32-bit IPv4 address.

IPv6

Identical to the 128-bit IPv6 address.

IPX

A concatenation of a 32 bit network number and 48 bit node address, by default the Ethernet interface's MAC-48 address.

JXTA

A 160 bit SHA-1 URN.

NCP

Similar to IPX.

RSVP

A combination of various RSVP session attributes and IPv4 addresses.

SCTP

A combination of the host IP addresses (plural) and the SCTP port used. So different SCTP ports on the same IP address are different SCTP endpoints, but the same SCTP port on different IP addresses of the same host are still the same endpoint.

TCP

A combination of the IP address and the TCP port used. Different TCP ports on the same IP address are different TCP endpoints.

Token Ring

Identical to the Token Ring MAC-48 address.

UDP

A combination of the IP address and the UDP port used, so different UDP ports on the same IP address are different UDP endpoints.

USB

Identical to the 7-bit USB address.

Broadcast and multicast endpoints

Broadcast and multicast traffic will be shown separately as additional endpoints. Of course, as these aren't physical endpoints the real traffic will be received by some or all of the listed unicast endpoints.

5.1. The “Endpoints” window

This window shows statistics about the endpoints captured.

Figure 8.4. The “Endpoints” window



For each supported protocol, a tab is shown in this window. Each tab label shows the number of endpoints captured (e.g. the tab label “Ethernet · 4” tells you that four ethernet endpoints have been captured). If no endpoints of a specific protocol were captured, the tab label will be greyed out (although the related page can still be selected).

Each row in the list shows the statistical values for exactly one endpoint.

Name resolution will be done if selected in the window and if it is active for the specific protocol layer (MAC layer for the selected Ethernet endpoints page). *Limit to display filter* will only show conversations matching the current display filter. Note that in this example we have GeoIP configured which gives us extra geographic columns. See [Section 10.10, “GeoIP Database Paths”](#) for more information.

The Copy button will copy the list values to the clipboard in CSV (Comma Separated Values) or YAML format. The Map button will show the endpoints mapped in your web browser.

Endpoint Types lets you choose which traffic type tabs are shown. See [Section 8.5, “Endpoints”](#) above for a list of endpoint types. The enabled types are saved in your profile settings.

Tip

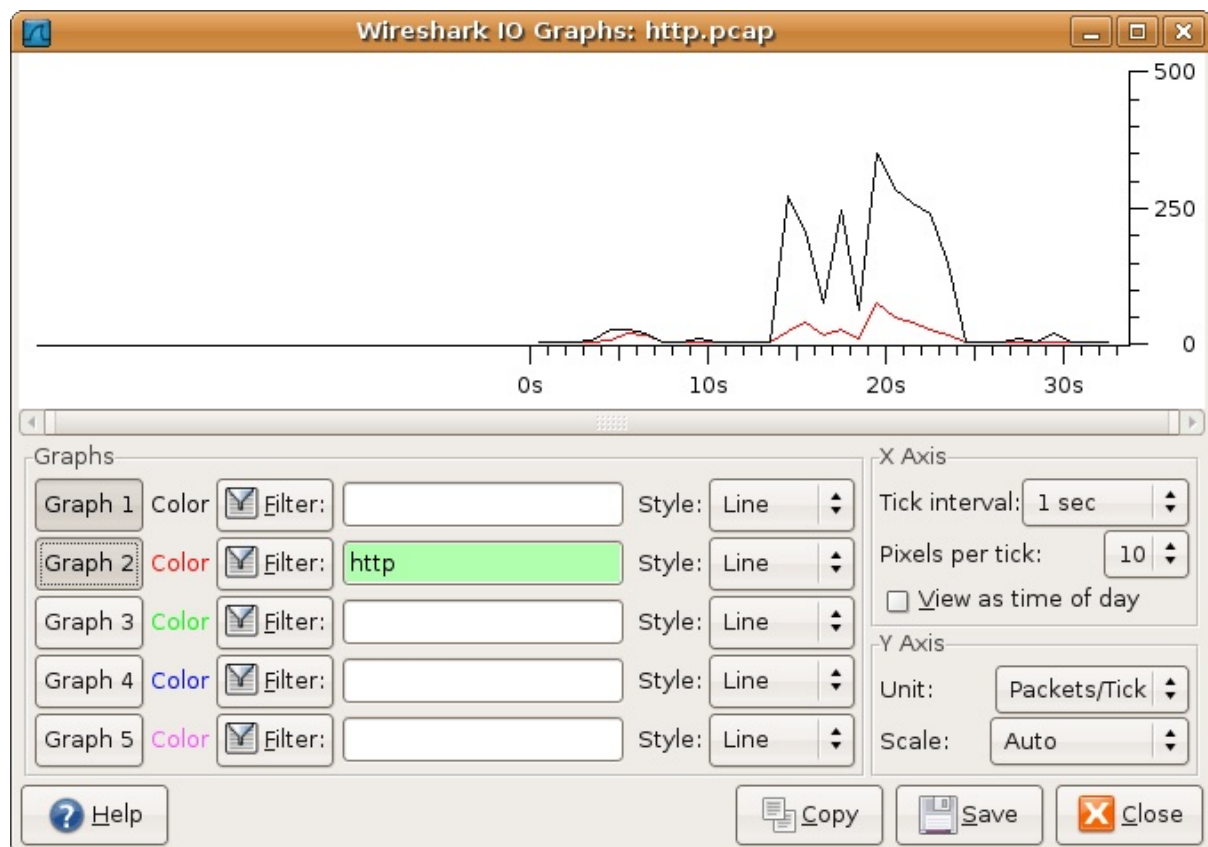
This window will be updated frequently, so it will be useful even if you open it before (or while) you are doing a live capture.

6. The “IO Graphs” window

User configurable graph of the captured network packets.

You can define up to five differently colored graphs.

Figure 8.5. The “IO Graphs” window



The user can configure the following things:

- *Graphs*
 - *Graph 1-5*: enable the specific graph 1-5 (only graph 1 is enabled by default)
 - *Color*: the color of the graph (cannot be changed)
 - *Filter*: a display filter for this graph (only the packets that pass this filter will be taken into account for this)

- graph)
 - *Style*: the style of the graph (Line/Impulse/FBar/Dot)
- *X Axis*
 - *Tick interval*: an interval in x direction lasts (10/1 minutes or 10/1/0.1/0.01/0.001 seconds)
 - *Pixels per tick*: use 10/5/2/1 pixels per tick interval
 - *View as time of day*: option to view x direction labels as time of day instead of seconds or minutes since beginning of capture
- *Y Axis*
 - *Unit*: the unit for the y direction (Packets/Tick, Bytes/Tick, Bits/Tick, Advanced...) [XXX - describe the Advanced feature.]
 - *Scale*: the scale for the y unit (Logarithmic,Auto,10,20,50,100,200,500,...)

The Save button will save the currently displayed portion of the graph as one of various file formats.

The Copy button will copy values from selected graphs to the clipboard in CSV (Comma Separated Values) format.

Tip

Click in the graph to select the first package in the selected interval.

7. Service Response Time

The service response time is the time between a request and the corresponding response. This information is available for many protocols.

Service response time statistics are currently available for the following protocols:

- *DCE-RPC*
- *Fibre Channel*
- *H.225 RAS*
- *LDAP*
- *LTE MAC*
- *MGCP*
- *ONC-RPC*
- *SMB*

As an example, the DCE-RPC service response time is described in more detail.

Note

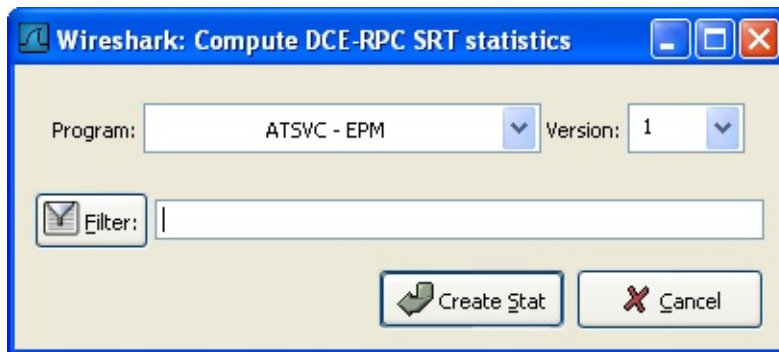
The other Service Response Time windows will work the same way (or only slightly different) compared to the following description.

7.1. The "Service Response Time DCE-RPC" window

The service response time of DCE-RPC is the time between the request and the corresponding response.

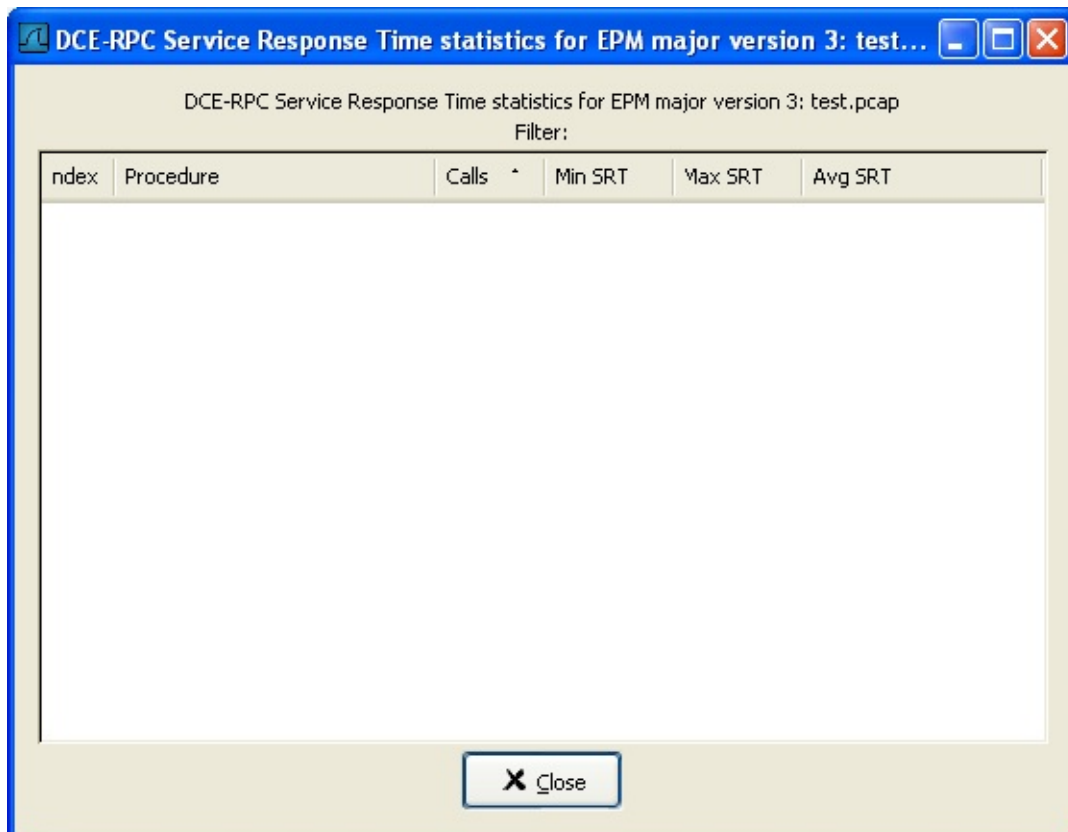
First of all, you have to select the DCE-RPC interface:

Figure 8.6. The "Compute DCE-RPC statistics" window



You can optionally set a display filter, to reduce the amount of packets.

Figure 8.7. The "DCE-RPC Statistic for ..." window



Each row corresponds to a method of the interface selected (so the EPM interface in version 3 has 7 methods). For each method the number of calls, and the statistics of the SRT time is calculated.

8. Compare two capture files

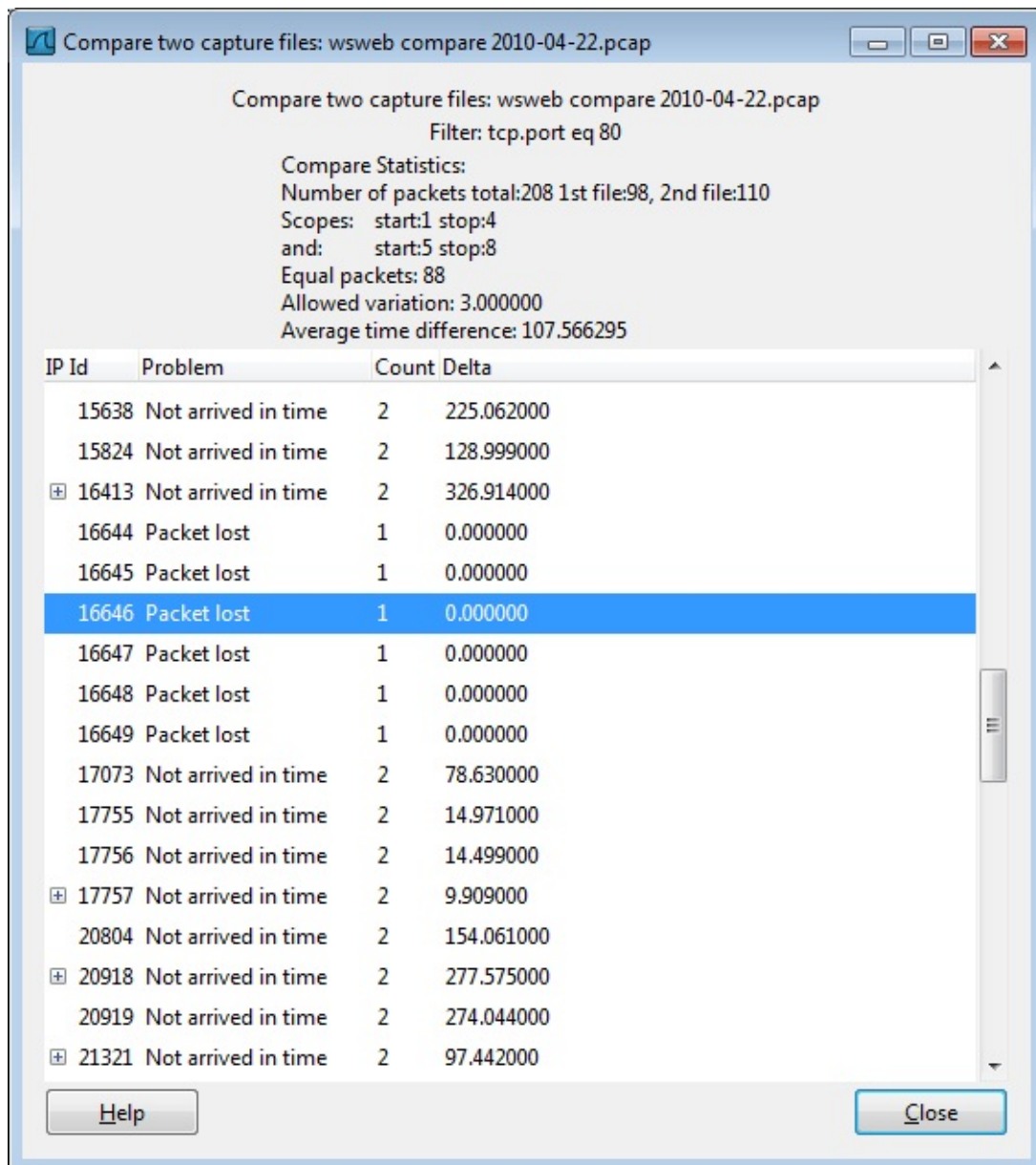
Compare two capture files.

This feature works best when you have merged two capture files chronologically, one from each side of a client/server connection.

The merged capture data is checked for missing packets. If a matching connection is found it is checked for:

- IP header checksums
- Excessive delay (defined by the "Time variance" setting)
- Packet order

Figure 8.8. The "Compare" window



You can configure the following:

- *Start compare*: Start comparing when this many IP IDs are matched. A zero value starts comparing immediately.
- *Stop compare*: Stop comparing when we can no longer match this many IP IDs. Zero always compares.
- *Endpoint distinction*: Use MAC addresses or IP time-to-live values to determine connection endpoints.
- *Check order*: Check for the same IP ID in the previous packet at each end.
- *Time variance*: Trigger an error if the packet arrives this many milliseconds after the average delay.
- *Filter*: Limit comparison to packets that match this display filter.

The info column contains new numbering so the same packets are parallel.

The color filtering differentiates the two files from each other. A "zebra" effect is created if the Info column is sorted.

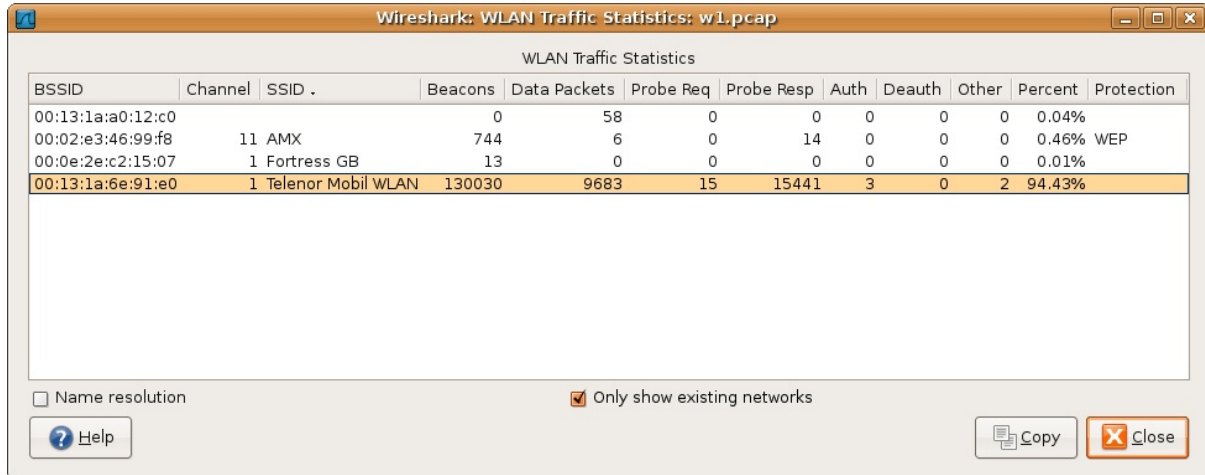
Tip

If you click on an item in the error list its corresponding packet will be selected in the main window.

9. WLAN Traffic Statistics

Statistics of the captured WLAN traffic. This window will summarize the wireless network traffic found in the capture. Probe requests will be merged into an existing network if the SSID matches.

Figure 8.9. The "WLAN Traffic Statistics" window



The screenshot shows the 'Wireshark: WLAN Traffic Statistics: w1.pcap' window. It contains a table titled 'WLAN Traffic Statistics' with the following data:

BSSID	Channel	SSID	Beacons	Data Packets	Probe Req	Probe Resp	Auth	Deauth	Other	Percent	Protection
00:13:1a:a0:12:c0			0	58	0	0	0	0	0	0.04%	
00:02:e3:46:99:f8	11	AMX	744	6	0	14	0	0	0	0.46%	WEP
00:0e:2e:c2:15:07	1	Fortress GB	13	0	0	0	0	0	0	0.01%	
00:13:1a:6e:91:e0	1	Telenor Mobil WLAN	130030	9683	15	15441	3	0	2	94.43%	

Below the table, there are two checkboxes: 'Name resolution' (unchecked) and 'Only show existing networks' (checked). At the bottom right, there are 'Copy' and 'Close' buttons. A 'Help' button is located at the bottom left.

Each row in the list shows the statistical values for exactly one wireless network.

Name resolution will be done if selected in the window and if it is active for the MAC layer.

Only show existing networks will exclude probe requests with a SSID not matching any network from the list.

The Copy button will copy the list values to the clipboard in CSV (Comma Separated Values) format.

Tip

This window will be updated frequently, so it will be useful, even if you open it before (or while) you are doing a live capture.

10. The protocol specific statistics windows

The protocol specific statistics windows display detailed information of specific protocols and might be described in a later version of this document.

Some of these statistics are described at <https://wiki.wireshark.org/Statistics>.

Sources

- https://www.wireshark.org/docs/wsug_html/#ChStatistics

Analyse VoIP Wireshark

- 1. Fichiers d'exemple
- 2. Configuration générale
 - Création d'un profil de Configuration
 - Affichage des colonnes
 - Horodatage
 - Alignement des colonnes à gauche
 - Champs comme colonne
 - Règles de colorisation
- 3. Première analyse
 - Propriétés du fichier
 - Hiérarchie de protocoles
 - Points d'extrémités et conversations
- 4. Analyse VoIP
 - Protocoles VoIP
 - Champs de références Wireshark
 - SIP Statistics
 - Menu VoIP Calls
 - Champs SIP
 - Jouer une conversation téléphonique
 - Analyse RTP
 - SSRC
 - Sample Rate
 - Paramètres RTP
 - Analyse de flux RTP
- 5. Captures SIP/RTP/VoIP

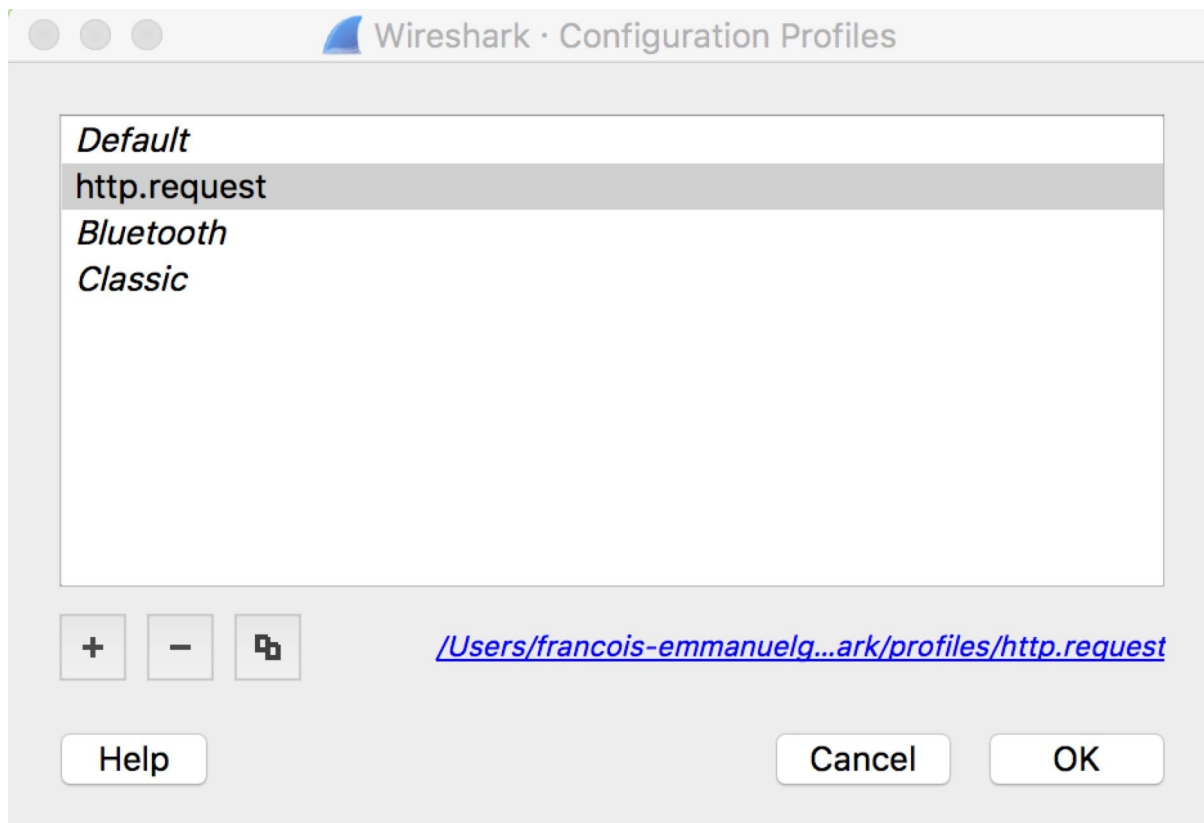
1. Fichiers d'exemple

- get http + dns : <https://www.cloudshark.org/captures/26c43039ccd6>
- sip-2phones-B2BUA-Asterisk.pcapng : <https://www.cloudshark.org/captures/b89478f0b2d8/>
- sip.pcapng : <https://www.cloudshark.org/captures/3380b7960f4d/>
- Forensic_challenge_4.pcap : <https://www.cloudshark.org/captures/908f122e3300/>

2. Configuration générale

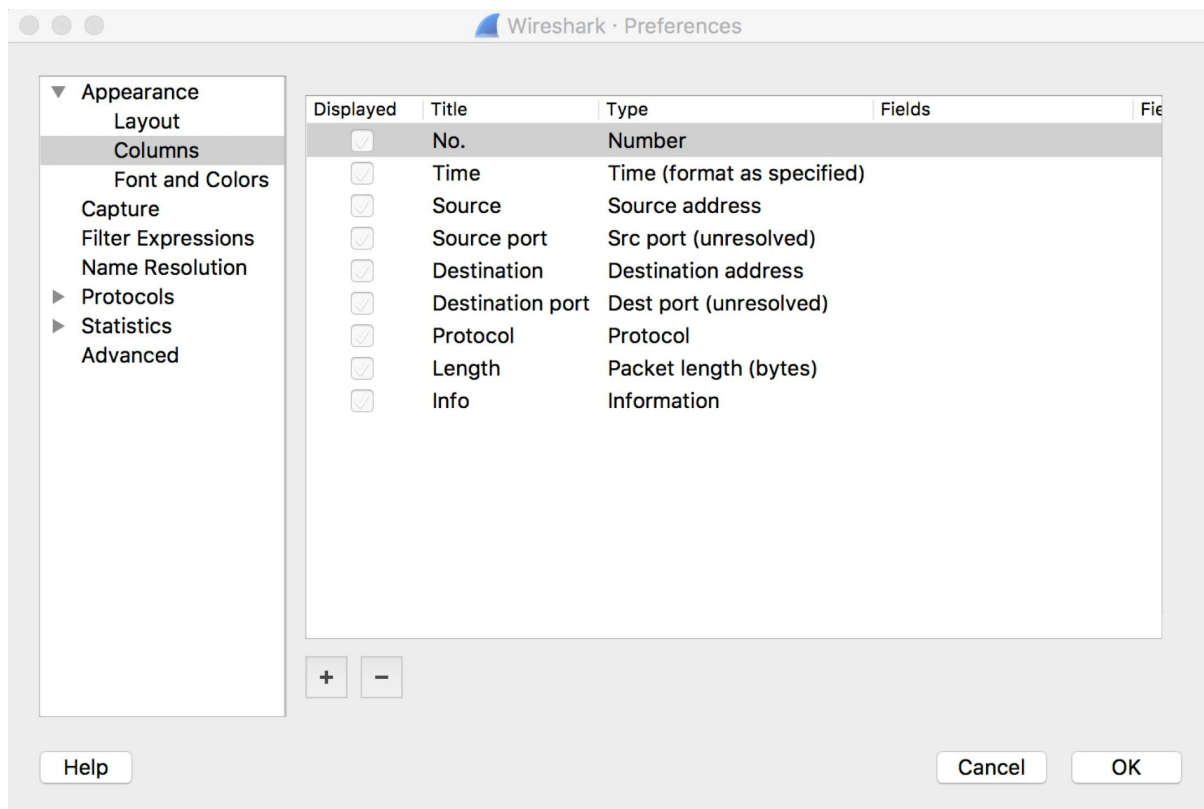
Création d'un profil de Configuration

Menu : Edit → Configuration Profiles ...



Affichage des colonnes

Menu : Edit → Preferences

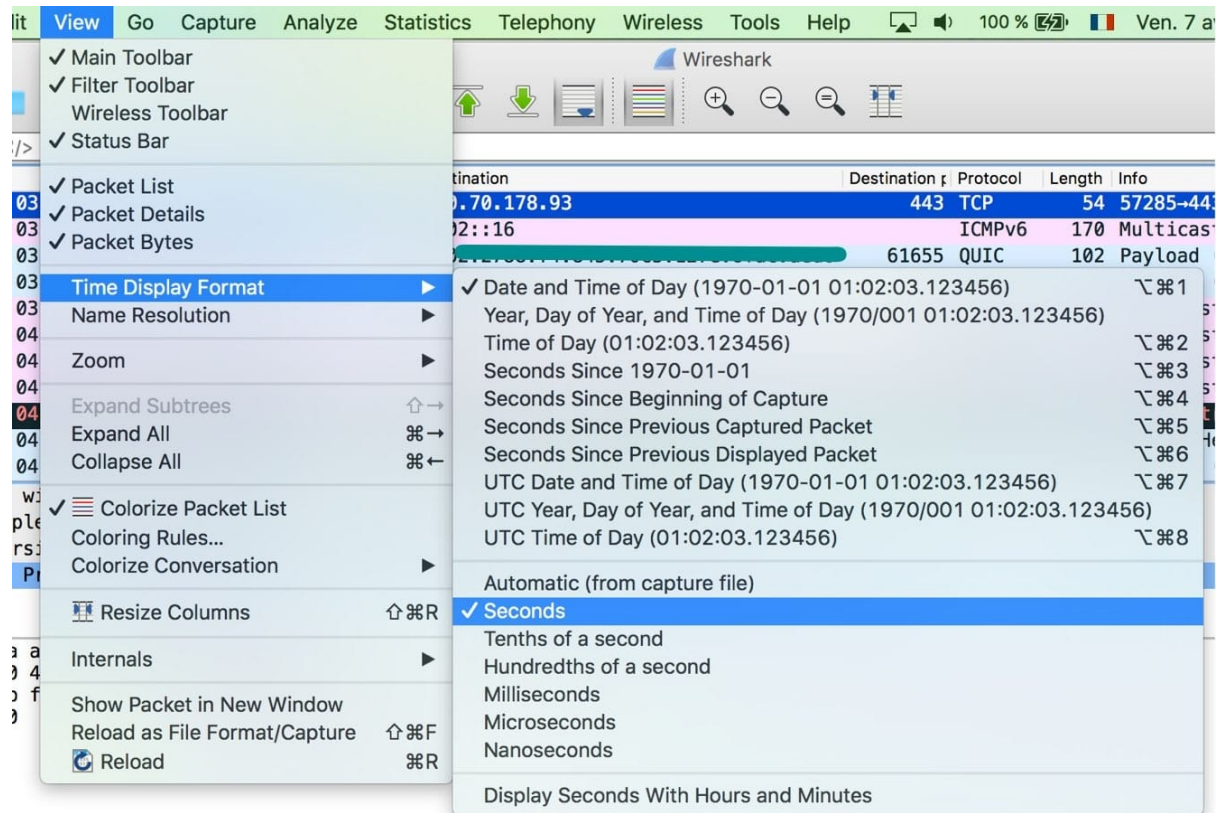


Horodatage

Le format par défaut de l'horodatage est "Seconds Since Beginning Capture".

Pour changer en "Moment de la journée" : Aller dans le menu View → Time Display Format → Date and Time of Day.

Avec un affichage à la "Seconds" View → Time Display Format → "Seconds: 0"



Alignement des colonnes à gauche

Clic droit et alignement à gauche

Champs comme colonne

Par exemple ici avec le filtre `http.request`, utiliser le champ HTTP `Host:` en colonne :

Propriétés du fichier

Menu : Statistics → Capture File Properties

Wireshark · Capture File Properties · sip

Details

Length: 1000 kB
 Format: Wireshark/... - pcapng
 Encapsulation: Ethernet

Time

First packet: 2005-01-14 18:58:02
 Last packet: 2005-01-14 18:58:37
 Elapsed: 00:00:34

Capture

Hardware: Unknown
 OS: Unknown
 Application: Editcap 2.2.5

Interfaces

<u>Interface</u>	<u>Dropped packets</u>	<u>Capture filter</u>	<u>Link type</u>	<u>Packet size limit</u>
Unknown	Unknown	Unknown	Ethernet	65535 bytes

Statistics

<u>Measurement</u>	<u>Captured</u>	<u>Displayed</u>	<u>Marked</u>
Packets	4269	4269 (100.0%)	N/A
Time span, s	34.909	34.909	N/A
Average pps	122.3	122.3	N/A
Average packet size, B	200.5	200.5	N/A
Bytes	855508	855508 (100.0%)	0
Average bytes/s	24 k	24 k	N/A
Average bits/s	196 k	196 k	N/A

Capture file comments

Help Refresh Copy To Clipboard Close Save Comments

Hiérarchie de protocoles

Pour avoir une vue d'ensemble des protocoles de la capture.

Menu : Statistics → Protocol Hierarchy

Couche 3

Wireshark · Protocol Hierarchy Statistics · sip

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
▼ Frame	100.0	4269	100.0	855508	196 k	0	0	0
▼ Ethernet	100.0	4269	7.0	59766	13 k	0	0	0
▶ Logical-Link Control	3.8	162	0.8	6561	1503	0	0	0
▶ Internet Protocol Version 4	96.2	4105	9.6	82100	18 k	0	0	0
Address Resolution Protocol	0.0	2	0.0	56	12	2	56	12

No display filter.

Help Copy Close

- LLC
 - CDP
 - STP
- IPv4
- ARP

Couche 4 IPv4

- UDP (84%)
- TCP (12%)

Couche 7 IPv4/UDP

Wireshark · Protocol Hierarchy Statistics · sip

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
▼ Frame	100.0	4269	100.0	855508	196 k	0	0	0
▼ Ethernet	100.0	4269	7.0	59766	13 k	0	0	0
▶ Logical-Link Control	3.8	162	0.8	6561	1503	0	0	0
▼ Internet Protocol Version 4	96.2	4105	9.6	82100	18 k	0	0	0
▼ User Datagram Protocol	83.9	3583	3.4	28664	6568	0	0	0
▶ Simple Network Management Protocol	11.8	502	3.7	31465	7210	278	20482	4693
Session Initiation Protocol	0.3	14	0.8	7236	1658	14	7236	1658
Real-Time Transport Protocol	33.8	1445	28.8	245996	56 k	1445	245996	56 k
Real-time Transport Control Protocol	1.3	54	0.3	2296	526	26	0	0
▶ Media Gateway Control Protocol	0.7	30	0.0	0	0	28	0	0
H323-MESSAGES	0.6	24	0.2	1664	381	24	1664	381
Data	36.1	1542	31.8	272156	62 k	1542	272156	62 k
▶ Transmission Control Protocol	12.2	522	12.9	110206	25 k	364	48182	11 k
Address Resolution Protocol	0.0	2	0.0	56	12	2	56	12

No display filter.

Help Copy Close

- SNMP
- SIP
- RTP
- RTCP
- MGCP
- H323

Couche 7 IPv4/TCP

Wireshark · Protocol Hierarchy Statistics · sip

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End E
▼ Frame	100.0	4269	100.0	855508	196 k	0	0	0
▼ Ethernet	100.0	4269	7.0	59766	13 k	0	0	0
▶ Logical-Link Control	3.8	162	0.8	6561	1503	0	0	0
▼ Internet Protocol Version 4	96.2	4105	9.6	82100	18 k	0	0	0
▶ User Datagram Protocol	83.9	3583	3.4	28664	6568	0	0	0
▼ Transmission Control Protocol	12.2	522	12.9	110206	25 k	364	48182	11 k
▶ Yahoo YMSG Messenger Protocol	0.1	3	0.0	107	24	3	107	24
▶ Telnet	0.4	19	0.8	6605	1513	19	6605	1513
▶ Short Message Peer to Peer	0.0	2	0.0	66	15	0	0	0
▶ Post Office Protocol	0.6	27	0.1	937	214	27	937	214
▶ MSN Messenger Service	0.1	6	0.1	628	143	6	628	143
▶ Hypertext Transfer Protocol	1.9	83	10.6	90812	20 k	47	50609	11 k
▶ Data	0.4	18	0.0	127	29	18	127	29
▶ Address Resolution Protocol	0.0	2	0.0	56	12	2	56	12

No display filter.

Help Copy Close

- Yahoo Messenger
- POP
- MSN Messenger
- HTTP

Points d'extrémités et conversations

Pour avoir une idée des Hôtes physiques et logiques de la capture.

Menu : Statistics → Endpoints

Menu : Statistics → Conversations

Couche 2

Wireshark · Endpoints · sip

Ethernet · 23 IPv4 · 21 IPv6 TCP · 74 UDP · 31

Address	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A
00:00:00_60:dd:19	1,861	428 k	782	148 k	1,079	279 k
Dfi_0c:0f:8a	28	2087	17	1020	11	1067
OracleCo_94:63:3e	145	43 k	71	22 k	74	20 k
OracleCo_94:64:c6	603	71 k	285	35 k	318	35 k
CiscoInc_95:b8:a8	80	6565	39	3667	41	2898
HewlettP_de:3c:51	98	13 k	52	4606	46	8541
Xircom_b4:cb:61	18	4134	9	527	9	3607
Dell_37:75:9b	2,922	616 k	1,457	307 k	1,465	309 k
Dell_37:f1:5c	916	188 k	438	125 k	478	62 k
AudioCod_04:5f:13	1,505	312 k	936	192 k	569	120 k
AskeyCom_ba:03:d6	36	4147	21	2425	15	1722
CiscoInc_10:84:4f	18	1117	18	1117	0	0
CiscoInc_10:84:50	18	1117	18	1117	0	0
CiscoInc_10:84:51	18	1117	18	1117	0	0
CiscoInc_10:84:52	18	1117	18	1117	0	0
CiscoInc_10:84:54	18	1117	18	1117	0	0
CiscoInc_10:84:55	18	1117	18	1117	0	0
CiscoInc_10:84:56	18	1117	18	1117	0	0
CiscoInc_10:84:57	18	1117	18	1117	0	0
CiscoInc_10:84:58	18	1117	18	1117	0	0
CDP/VTP/DTP/PagP/UDLD	9	873	0	0	9	873
Spanning-tree-(for-bridges)_00	153	9180	0	0	153	9180
Broadcast	2	120	0	0	2	120

☒ Name resolution ☐ Limit to display filter Endpoint Types ▼

Help Copy Map Close

On reconnaît du Cisco, du Dell, du Xircom, ...

UDP

Wireshark · Endpoints · sip

Ethernet · 23 IPv4 · 21 IPv6 TCP · 74 UDP · 31

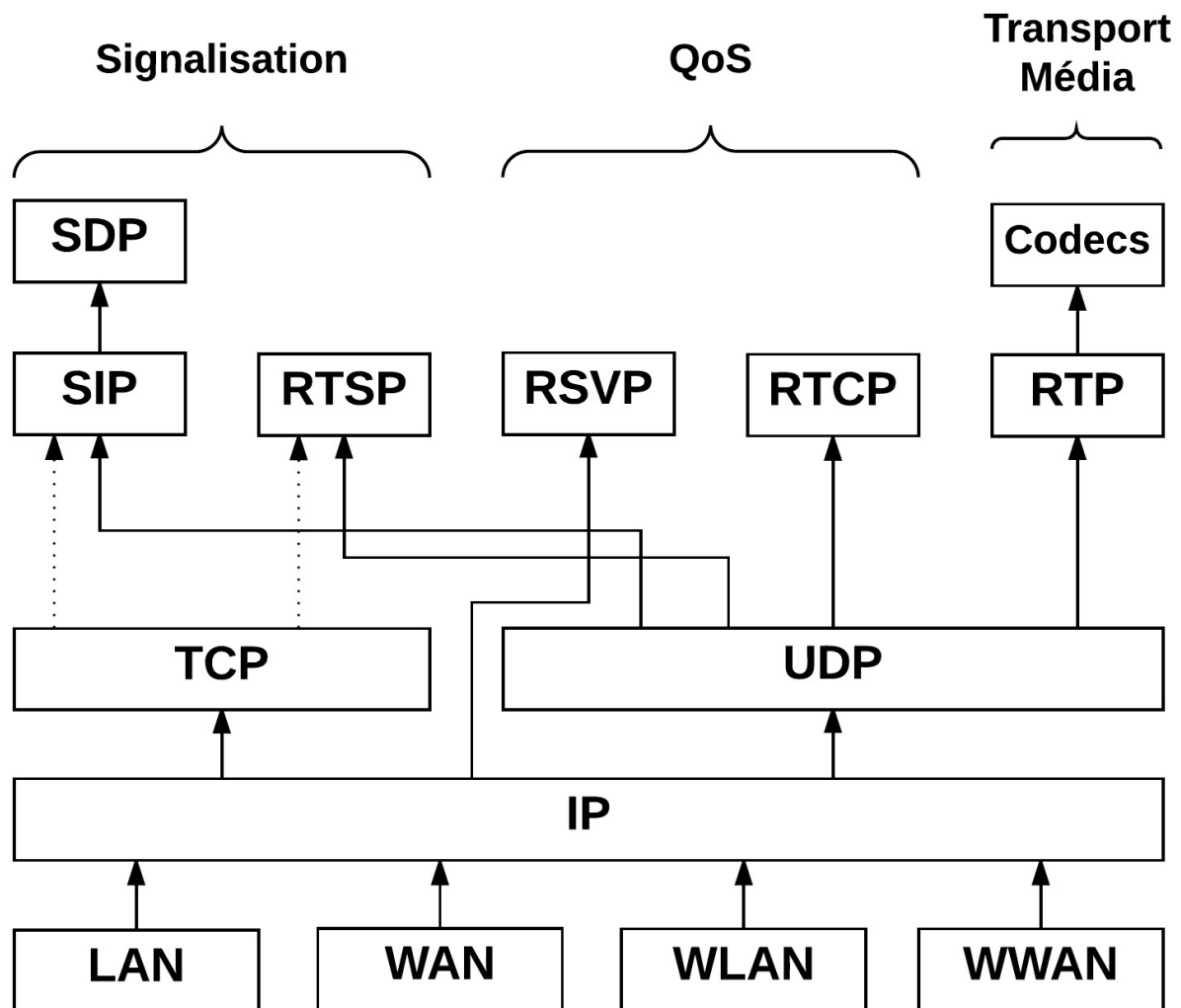
Address	Port	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Latitude
200.57.7.196	40379	14	1856	8	1076	6	780	—
200.57.7.196	40378	1,439	305 k	548	117 k	891	188 k	—
200.57.7.196	40377	14	1856	6	780	8	1076	—
200.57.7.196	40376	1,439	305 k	891	188 k	548	117 k	—
200.57.7.196	40362	6	1284	0	0	6	1284	—
200.57.7.202	30000	6	1284	6	1284	0	0	—
200.57.7.202	9693	4	314	2	120	2	194	—
200.57.7.188	8033	4	357	2	163	2	194	—
200.57.7.204	8001	14	1856	8	1076	6	780	—
200.57.7.204	8000	1,439	305 k	548	117 k	891	188 k	—
200.57.7.204	5061	10	5876	6	3377	4	2499	—
200.57.7.205	5061	7	2128	5	1100	2	1028	—
200.57.7.195	5060	17	8004	6	3527	11	4477	—
200.57.7.195	5010	4	4292	2	2061	2	2231	—
200.57.7.197	5010	4	4292	2	2231	2	2061	—
200.57.7.195	5002	36	19 k	18	9316	18	10 k	—
200.57.7.197	5002	6	2253	3	1098	3	1155	—
200.57.7.194	5001	42	21 k	21	11 k	21	10 k	—
200.57.7.199	4801	14	1856	6	780	8	1076	—
200.57.7.199	4800	1,439	305 k	891	188 k	548	117 k	—
200.57.7.205	3529	8	1018	4	624	4	394	—
200.57.7.197	2428	52	5352	13	1747	39	3605	—
200.57.7.195	2427	30	2863	15	1722	15	1141	—
200.57.7.202	2427	26	2549	13	1021	13	1528	—

☐ Name resolution ☐ Limit to display filter Endpoint Types ▾

Help Copy ▾ Map Close

4. Analyse VoIP

Protocoles VoIP



Champs de références Wireshark

- SIP : <https://www.wireshark.org/docs/dfref/s/sip.html>
- SDP : <https://www.wireshark.org/docs/dfref/s/sdp.html>
- RTP : <https://www.wireshark.org/docs/dfref/r/rtp.html>
- RTCP : <https://www.wireshark.org/docs/dfref/r/rtcp.html>

SIP Statistics

Menu : Telephony → SIP Statistics

```
tshark -r sip-2phones-B2BUA-Asterisk.pcapng -q -z sip,stat
```

```
=====
SIP Statistics

Number of SIP messages: 60
Number of resent SIP messages: 1

* SIP Status Codes in reply packets
SIP 401 Unauthorized      :    7 Packets
SIP 404 Not Found        :    1 Packets
```

```

SIP 180 Ringing      :    2 Packets
SIP 200 OK           :    9 Packets
SIP 480 Temporarily Unavailable :    1 Packets
SIP 489 Bad Event    :    6 Packets
SIP 100 Trying       :    3 Packets

```

* List of SIP Request methods

```

INVITE      :    7 Packets
ACK         :    6 Packets
REGISTER    :    4 Packets
PUBLISH     :    3 Packets
SUBSCRIBE   :    9 Packets
BYE         :    2 Packets

```

* Average setup time 2430 ms

Min 1 ms

Max 7071 ms

=====

→ SIP Flows

Wireshark · SIP Flows · sip2

Start Time	Stop Time	Initial Speaker	From	To	Protocol	Packets	State	Comments
0.000000	0.001208	172.16.98.1	<sip:telephone1@172.16.98.101;transport=UDP	<sip:telephone1@172.16.98.101;transport=UDP	SIP	4	REJECTED	REGISTER 401
7.346350	7.350481	172.16.98.145	<sip:telephone2@172.16.98.101;transport=UDP	<sip:telephone2@172.16.98.101;transport=UDP	SIP	4	REJECTED	REGISTER 401
7.456642	7.457011	172.16.98.145	<sip:telephone2@172.16.98.101;transport=UDP	<sip:telephone2@172.16.98.101;transport=UDP	SIP	2	REJECTED	PUBLISH 489
7.456788	7.459570	172.16.98.145	<sip:telephone2@172.16.98.101;transport=UDP	<sip:telephone2@172.16.98.101;transport=UDP	SIP	4	REJECTED	SUBSCRIBE 401
7.456880	7.461719	172.16.98.145	<sip:telephone2@172.16.98.101;transport=UDP	<sip:telephone2@172.16.98.101;transport=UDP	SIP	4	REJECTED	SUBSCRIBE 401
12.038851	26.728901	172.16.98.1	<sip:telephone1@172.16.98.101;transport=UDP	<sip:2302@172.16.98.101;transport=UDP	SIP	13	COMPLETED	INVITE 401 200
12.042406	26.974147	172.16.98.101	<sip:telephone1@172.16.98.101	<sip:telephone2@172.16.98.145:4...=98920ece8998d44b;transport=UDP	SIP	15	COMPLETED	INVITE 200 200 200
13.197562	13.223478	172.16.98.1	<sip:telephone2@domain.xyz;transport=UDP	<sip:telephone2@domain.xyz;transport=UDP	SIP	2	REJECTED	SUBSCRIBE 480
19.473847	19.475791	172.16.98.145	<sip:telephone2@172.16.98.101;transport=UDP	<sip:telephone2@172.16.98.101;transport=UDP	SIP	2	REJECTED	PUBLISH 489
19.474392	19.505281	172.16.98.145	<sip:telephone2@172.16.98.101;transport=UDP	<sip:telephone2@172.16.98.101;transport=UDP	SIP	4	REJECTED	SUBSCRIBE 401
27.181702	27.182098	172.16.98.145	<sip:telephone2@172.16.98.101;transport=UDP	<sip:telephone2@172.16.98.101;transport=UDP	SIP	2	REJECTED	PUBLISH 489
27.181844	27.184256	172.16.98.145	<sip:telephone2@172.16.98.101;transport=UDP	<sip:telephone2@172.16.98.101;transport=UDP	SIP	4	REJECTED	SUBSCRIBE 401

Help Prepare Filter Flow Sequence Play Streams Copy Cancel OK

Menu VoIP Calls

```
tshark -t ud -r sip.pcapng -Y "sip.CSeq.method eq INVITE"
```

```

1 2005-01-14 17:58:02.965944 200.57.7.195 → 200.57.7.204 SIP/SDP Request: INVITE sip:francisco@bestel.com:550
60 |
2 2005-01-14 17:58:02.973833 200.57.7.204 → 200.57.7.195 SIP Status: 100 Trying |
3 2005-01-14 17:58:03.013468 200.57.7.204 → 200.57.7.195 SIP Status: 180 Ringing |
498 2005-01-14 17:58:11.443869 200.57.7.204 → 200.57.7.195 SIP/SDP Status: 200 Ok |
2964 2005-01-14 17:58:27.631897 200.57.7.195 → 200.57.7.204 SIP/SDP Request: INVITE sip:francisco@bestel.com:55
060 |
2965 2005-01-14 17:58:27.640624 200.57.7.204 → 200.57.7.195 SIP Status: 100 Trying |
2967 2005-01-14 17:58:27.658696 200.57.7.204 → 200.57.7.195 SIP Status: 180 Ringing |

```

```
tshark -t ud -r sip-2phones-B2BUA-Asterisk.pcapng -Y "sip.CSeq.method eq INVITE"
```

```

19 2016-05-18 18:23:15.920336 172.16.98.1 → 172.16.98.101 SIP/SDP Request: INVITE sip:2302@172.16.98.101;tran
sport=UDP |
20 2016-05-18 18:23:15.921110 172.16.98.101 → 172.16.98.1 SIP Status: 401 Unauthorized |
22 2016-05-18 18:23:15.921826 172.16.98.1 → 172.16.98.101 SIP/SDP Request: INVITE sip:2302@172.16.98.101;tran
sport=UDP |
23 2016-05-18 18:23:15.922973 172.16.98.101 → 172.16.98.1 SIP Status: 100 Trying |
24 2016-05-18 18:23:15.923891 172.16.98.101 → 172.16.98.145 SIP/SDP Request: INVITE sip:telephone2@172.16.98.1
45:40216;rinstance=98920ece8998d44b;transport=UDP |
25 2016-05-18 18:23:16.423460 172.16.98.101 → 172.16.98.145 SIP/SDP Request: INVITE sip:telephone2@172.16.98.1
45:40216;rinstance=98920ece8998d44b;transport=UDP |
26 2016-05-18 18:23:16.559683 172.16.98.145 → 172.16.98.101 SIP Status: 100 Trying |
27 2016-05-18 18:23:16.559967 172.16.98.145 → 172.16.98.101 SIP Status: 100 Trying |

```

```

28 2016-05-18 18:23:16.664606 172.16.98.145 → 172.16.98.101 SIP Status: 180 Ringing |
29 2016-05-18 18:23:16.665071 172.16.98.101 → 172.16.98.1 SIP Status: 180 Ringing |
32 2016-05-18 18:23:22.982256 172.16.98.145 → 172.16.98.101 SIP/SDP Status: 200 OK |
34 2016-05-18 18:23:22.986781 172.16.98.101 → 172.16.98.1 SIP/SDP Status: 200 OK |
35 2016-05-18 18:23:22.987124 172.16.98.101 → 172.16.98.145 SIP/SDP Request: INVITE sip:telephone2@172.16.98.1
45:40216, in-dialog |
37 2016-05-18 18:23:22.993949 172.16.98.101 → 172.16.98.1 SIP/SDP Request: INVITE sip:telephone1@172.16.98.1:
44285;transport=UDP, in-dialog |
38 2016-05-18 18:23:23.016554 172.16.98.1 → 172.16.98.101 SIP/SDP Status: 200 OK |
41 2016-05-18 18:23:23.173703 172.16.98.145 → 172.16.98.101 SIP/SDP Status: 200 OK |
406 2016-05-18 18:23:30.610529 172.16.98.101 → 172.16.98.145 SIP/SDP Request: INVITE sip:telephone2@172.16.98.1
45:40216, in-dialog |
407 2016-05-18 18:23:30.849885 172.16.98.145 → 172.16.98.101 SIP/SDP Status: 200 OK |

```

Champs SIP

Ajuster les colonnes sur base des filtres ou baser des recherches sur :

sip.User-Agent

sip.Status-Line

sip.Callid

sip.Status-Code > 399

Jouer une conversation téléphonique

Menu : Telephony → VoIP Calls → choisir un flux → Play Stream

The screenshot shows the Wireshark RTP Player interface. The top part displays a waveform of the audio stream. Below the waveform is a table with columns: Source Address, Source Port, Destination Address, Destination Port, SSRC, Setup Frame, Packets, Time Span (s), Sample Rate (Hz), and Payloads. The bottom part of the window shows a list of packets with columns: Start Time, Stop Time, and Comments. The 'Play Streams' button is circled in red, and the 'Play' button is also circled in red.

Jouer la session RTP Forensic_challenge_4.pcap (<https://www.cloudshark.org/captures/908f122e3300>)

Analyse RTP

- SSRC
- Setup frame
- Time Spans (s)
- Sample Rate (Hz)
- Payloads (Codecs)

SSRC

La valeur SSRC (Synchronization source) identifie la source de synchronisation, **identifie un flux RTP**. La valeur est choisie de manière aléatoire, avec l'intention qu'aucune source de synchronisation dans la même session RTP n'aura le même SSRC. Bien que la probabilité que les sources multiples choisissant le même identifiant soit faible, toutes les implémentations RTP doivent être préparées pour détecter et résoudre les collisions. Si une source change son adresse de transport source, elle doit également choisir un nouveau SSRC pour éviter d'être interprété comme une source en boucle.

Sample Rate

La famille des codecs G.711 utilisent une fréquence d'échantillonnage de 8kHz (8000 Hz).

Cela signifie que le flux audio est échantillonné 8000 fois par seconde.

La longueur d'un échantillon audio est alors de $1/8000 \text{ s} = 0.000125 \text{ s} = 0.125 \text{ ms}$

Paramètres RTP

- Paquet: ce paramètre indique le nombre de paquets dans le fichier capturé.
- Sequence: ce paramètre indique le numéro de séquence RTP.
- Delta (ms): c'est la différence de temps entre le paquet actuel et le paquet précédent dans le flux.
- Filtered Jitter (ms): ce paramètre fait référence à la différence entre l'heure d'arrivée réelle et le paramètre d'horodatage RTP. Il devrait être aussi bas que possible et de préférence zéro.
- Skew *Inclinaison* (ms): Ce paramètre indique le début (ou le retard) du paquet par rapport à l'endroit où il était supposé être. Par exemple, si nous disposons d'un taux de paquets de 20 paquets par seconde, nous devrions avoir 50 ms entre les paquets, et si un paquet arrive 49 ms après le précédent, ce sera une inclinaison de -1 ms.
- IP BW (kbps): ce paramètre fait référence à la consommation de bande passante au niveau IP qui est avec tous les en-têtes jusqu'à la couche 3.
- Marqueur: ce paramètre indique si le marqueur est SET (SET = 1, UNSET = 0). Un marqueur indique différents phénomènes tels que la fin de la période de silence et la fin de la trame vidéo, et est ajouté par l'application.
Statut: ce paramètre vous permet de vérifier si le statut est correct.

Analyse de flux RTP

Menu : Telephony → RTP → RTP Streams / Stream Analysis

172.25.105.3:63184 ↔
172.25.105.40:18150

Forward

SSRC 0xa254e017
Max Delta 1940.06 ms @ 3767
Max Jitter 122.24 ms
Mean Jitter 11.28 ms
Max Skew -4872.48 ms
RTP Packets 1811
Expected 1781
Lost -30 (-1.68 %)
Seq Errs 71
Duration 38.69 s
Clock Drift -34363 ms
Freq Drift 894 Hz (-88.82 %)

Reverse

SSRC 0xa254e017
Max Delta 0.00 ms @ 0
Max Jitter 0.00 ms
Mean Jitter 0.00 ms
Max Skew 0.00 ms
RTP Packets 0
Expected 1
Lost 1 (100.00 %)
Seq Errs 0
Duration 0.00 s
Clock Drift 0 ms
Freq Drift 1 Hz (0.00 %)

Forward Reverse Graph

Packet	Sequence	Delta (ms)	Jitter (ms)	Skew	Bandwidth	Marker	Status
1310	6402	0.00	0.00	0.00	1.60		✓
1311	6403	1.15	1.18	18.85	3.20		✓
1312	6404	18.79	1.18	20.06	4.80		✓
1313	6405	20.01	1.11	20.05	6.40		✓
1314	6406	20.02	1.04	20.03	8.00		✓
1315	6407	20.07	0.98	19.95	9.60		✓
1316	6408	22.09	1.05	17.86	11.20		✓
1317	6409	16.97	1.17	20.89	12.80		✓
1318	6410	20.05	1.10	20.84	14.40		✓
1319	6411	20.07	1.04	20.77	16.00		✓
1320	6412	20.19	0.98	20.57	17.60		✓
1321	6413	25.11	1.24	15.46	19.20		✓
1322	6414	14.96	1.48	20.50	20.80		✓
1323	6415	20.13	1.40	20.37	22.40		✓
1324	6416	19.91	1.31	20.45	24.00		✓
1325	6417	19.92	1.24	20.53	25.60		✓
1326	6419	71.92	3.15	-11.39	27.20		Wrong sequence number
1327	6420	0.23	4.19	8.38	28.80		✓
1328	6421	8.01	4.68	20.36	30.40		✓
1329	6422	20.02	4.39	20.34	32.00		✓
1330	6423	19.06	4.17	21.28	33.60		✓
1331	6424	21.91	4.03	19.37	35.20		✓
1332	6425	23.24	3.98	16.13	36.80		✓
1333	6426	15.01	4.04	21.12	38.40		✓
1334	6427	20.09	3.80	21.04	40.00		✓
1335	6428	20.14	3.57	20.90	41.60		✓
1336	6429	20.06	3.35	20.84	43.20		✓
1337	6430	20.06	3.14	20.78	44.80		✓
1338	6431	20.04	2.95	20.74	46.40		✓

1 streams found.

Help Play Streams Close Save

172.25.105.3:63184 ↔
172.25.105.40:18150

Forward

SSRC 0xa254e017
Max Delta 1940.06 ms @ 3767
Max Jitter 122.24 ms
Mean Jitter 11.28 ms
Max Skew -4872.48 ms
RTP Packets 1811
Expected 1781
Lost -30 (-1.68 %)
Seq Errs 71
Duration 38.69 s
Clock Drift -34363 ms
Freq Drift 894 Hz (-88.82 %)

Reverse

SSRC 0xa254e017
Max Delta 0.00 ms @ 0
Max Jitter 0.00 ms
Mean Jitter 0.00 ms
Max Skew 0.00 ms
RTP Packets 0
Expected 1
Lost 1 (100.00 %)
Seq Errs 0
Duration 0.00 s
Clock Drift 0 ms
Freq Drift 1 Hz (0.00 %)

Forward Reverse Graph

1 streams found.

Help Play Streams Close Save

```
tshark -r sip-2phones-B2BUA-Asterisk.pcapng -q -z rtp,streams
```

===== RTP Streams =====										
Src IP addr	Port	Dest IP addr	Port	SSRC	Payload	Pkts	Lost	Max Delta(ms)	Max Jitter(ms)	Mean Jitter(ms)
172.16.98.1	8000	172.16.98.145	8000	0x44830D58	ITU-T G.711 PCMU	1	0 (0.0%)	0.00	0.00	0.00
172.16.98.145	8000	172.16.98.1	8000	0x6AB21E82	ITU-T G.711 PCMU	355	0 (0.0%)	52.35	10.17	7.64

```
tshark -r sip.pcapng -q -z rtp,streams
```


===== RTP Streams =====											
Src IP addr	Port	Dest IP addr	Port	SSRC	Payload	Pkts	Lost	Max Delta(ms)	Max	Jitter(ms)	Mean Jitter(ms)
200.57.7.204	8000	200.57.7.196	40376	0xD2BD4E3E	ITU-T G.711 PCMA	548	0 (0.0%)	5843.74		7.41	2.60
200.57.7.196	40376	200.57.7.204	8000	0x58F33DEA	ITU-T G.711 PCMA	891	0 (0.0%)	379.90		0.25	0.10
200.57.7.202	30000	200.57.7.196	40362	0x00002E3D	ITU-T G.711 PCMA	6	0 (0.0%)	30.04		0.64	1.74
=====											

5. Captures SIP/RTP/VoIP

Captures SIP/RTP/VoIP disponibles sur : https://github.com/goffinet/sip_captures/archive/master.zip

- voip-skype-conn-disconn.pcap : <https://www.cloudshark.org/captures/0b42a83a7bdb/>
- voip-extension2downata.pcap : <https://www.cloudshark.org/captures/edabc354d883/>
- voip-extension.pcap : <https://www.cloudshark.org/captures/74259932d9a7/>
- Forensic_challenge_4.pcap : <https://www.cloudshark.org/captures/908f122e3300/>
- nitroba-1.pcap : <https://www.cloudshark.org/captures/cab3a2212fc6/>
- sip.pcapng : <https://www.cloudshark.org/captures/3380b7960f4d/>
- uac-uas-scenario-sipp.pcapng : <https://www.cloudshark.org/captures/3670f914be66/>
- branch-scenario-sipp.pcapng : <https://www.cloudshark.org/captures/4b851a42650e/>
- sip-direct.pcapng : <https://www.cloudshark.org/captures/272c8a794a46/>
- RTP-Protocol.pcap : <https://www.cloudshark.org/captures/71c1b394bbbf/>
- SIP-SDP-Example.pcap : <https://www.cloudshark.org/captures/7d2a7dab3e8b/>
- SIP-Call-with-Proxy-Server.pcap : <https://www.cloudshark.org/captures/92cd2d7493e4/>
- SIP-Call-Flow-Over-TCP.pcap : <https://www.cloudshark.org/captures/ef94ec30c072/>
- REGISTER-Request.pcap : <https://www.cloudshark.org/captures/f29704feaf5a/>
- sip-2phones-B2BUA-Asterisk.pcapng : <https://www.cloudshark.org/captures/b89478f0b2d8/>
- sip-routing-error-wireshark.pcapng : <https://www.cloudshark.org/captures/53201ba9cda7/>
- sip-register-ok-unregister-2.pcapng : <https://www.cloudshark.org/captures/423ab1d45e27/>
- sip-register-publish-subscribe-489-401.pcapng : <https://www.cloudshark.org/captures/bc2a28f52239/>
- sip-register-wrong_password-401-403-401-403.pcapng : <https://www.cloudshark.org/captures/784f68cbb09a/>
- sip-register-wrong_user-401-403-404.pcapng : <https://www.cloudshark.org/captures/10222a758905/>
- sip-register-ok-unregister.pcapng : <https://www.cloudshark.org/captures/b9341bae1ff6/>
- sip-register-to-repo-proxy.pcapng : <https://www.cloudshark.org/captures/dcb7a3a608e3/>
- SIP-Call-with-Proxy-Server.pcap : <https://www.cloudshark.org/captures/dd1c9b582495/>
- SIP-Call-Flow-Over-TCP.pcap : <https://www.cloudshark.org/captures/4329c91164c0/>
- sip-rem-register.pcapng : <https://www.cloudshark.org/captures/f9308bf8648f/>
- sip-register.pcapng : <https://www.cloudshark.org/captures/de9c2cf75368/>
- SIP-SDP-Example.pcap : <https://www.cloudshark.org/captures/2b0d15028b9e/>
- sip-flow-private-3.pcapng : <https://www.cloudshark.org/captures/5881e936996f/>
- sip-flow-private-2.pcapng : <https://www.cloudshark.org/captures/e6a110f970e1/>
- sip-flow-private.pcapng : <https://www.cloudshark.org/captures/74ccb6b3c4ee/>
- sip-register-dns-srv.pcapng : <https://www.cloudshark.org/captures/7c416a39f3ff/>
- private-test-sip-2.pcapng : <https://www.cloudshark.org/captures/ef900c8a5097/>
- private-test-sip.pcapng : <https://www.cloudshark.org/captures/1ad93ac96295/>
- voip1b.pcapng : <https://www.cloudshark.org/captures/8c0fd816a8c7/>
- sip-phone-to-phone.pcapng : <https://www.cloudshark.org/captures/f22c8443205b/>
- sip-cme-zoiper-register.pcapng : <https://www.cloudshark.org/captures/e64f64d6bb1d/>
- test_voip.pcap.pcapng : <https://www.cloudshark.org/captures/0b3de158f25c/>

Wireshark en ligne de commande

- 1. [Introduction](#)
- 2. [tshark](#): Terminal-based Wireshark
- 3. [tcpdump](#): Capturing with `tcpdump` for viewing with Wireshark
- 4. [dumpcap](#): Capturing with `dumpcap` for viewing with Wireshark
- 5. [capinfos](#): Print information about capture files
- 6. [rawshark](#): Dump and analyze network traffic.
- 7. [editcap](#): Edit capture files
- 8. [mergcap](#): Merging multiple capture files into one
- 9. [text2pcap](#): Converting ASCII hexdumps to network captures
- 10. [reordercap](#): Reorder a capture file
- [Sources](#)

Texte en anglais

1. Introduction

Along with the main application, Wireshark comes with an array of command line tools which can be helpful for specialized tasks. These tools will be described in this chapter. You can find more information about each command in the [Manual Pages](#).

2. *tshark*: Terminal-based Wireshark

TShark is a terminal oriented version of Wireshark designed for capturing and displaying packets when an interactive user interface isn't necessary or available. It supports the same options as `wireshark`. For more information on `tshark` see the manual pages (`man tshark`).

Help information available from `tshark` .

```
TShark (Wireshark) 2.1.0 (v2.1.0rc0-502-g328fbc0 from master)
Dump and analyze network traffic.
See https://www.wireshark.org for more information.

Usage: tshark [options] ...

Capture interface:
-i <interface>          name or idx of interface (def: first non-loopback)
-f <capture filter>     packet filter in libpcap filter syntax
-s <snaplen>            packet snapshot length (def: 65535)
-p                      don't capture in promiscuous mode
-I                      capture in monitor mode, if available
-B <buffer size>        size of kernel buffer (def: 2MB)
-y <link type>          link layer type (def: first appropriate)
-D                      print list of interfaces and exit
-L                      print list of link-layer types of iface and exit

Capture stop conditions:
-c <packet count>       stop after n packets (def: infinite)
-a <autostop cond.> ... duration:NUM - stop after NUM seconds
                      filesize:NUM - stop this file after NUM KB
                      files:NUM - stop after NUM files

Capture output:
-b <ringbuffer opt.> ... duration:NUM - switch to next file after NUM secs
                      filesize:NUM - switch to next file after NUM KB
```

```

files:NUM - ringbuffer: replace after NUM files

RPCAP options:
-A <user>:<password>    use RPCAP password authentication
Input file:
-r <infile>             set the filename to read from (- to read from stdin)

Processing:
-2                     perform a two-pass analysis
-R <read filter>        packet Read filter in Wireshark display filter syntax
-Y <display filter>     packet display filter in Wireshark display filter
                        syntax
-n                     disable all name resolutions (def: all enabled)
-N <name resolve flags> enable specific name resolution(s): "mnNtCd"
-d <layer_type>==<selector>,<decode_as_protocol> ...
                        "Decode As", see the man page for details
                        Example: tcp.port==8888,http
-H <hosts file>         read a list of entries from a hosts file, which will
                        then be written to a capture file. (Implies -W n)
--disable-protocol <proto_name>
                        disable dissection of proto_name
--enable-heuristic <short_name>
                        enable dissection of heuristic protocol
--disable-heuristic <short_name>
                        disable dissection of heuristic protocol

Output:
-w <outfile|->         write packets to a pcap-format file named "outfile"
                        (or to the standard output for "-")
-C <config profile>     start with specified configuration profile
-F <output file type>   set the output file type, default is pcapng
                        an empty "-F" option will list the file types
-v                     add output of packet tree (Packet Details)
-O <protocols>         Only show packet details of these protocols, comma
                        separated
-P                     print packet summary even when writing to a file
-S <separator>         the line separator to print between packets
-x                     add output of hex and ASCII dump (Packet Bytes)
-T phtml|ps|psml|text|fields
                        format of text output (def: text)
-e <field>             field to print if -Tfields selected (e.g. tcp.port,
                        _ws.col.Info)
                        this option can be repeated to print multiple fields
-E<fieldsoption>=<value> set options for output when -Tfields selected:
    header=y|n         switch headers on and off
    separator=/t|/s|<char> select tab, space, printable character as separator
    occurrence=f|l|a    print first, last or all occurrences of each field
    aggregator=,<char> select comma, space, printable character as
                        aggregator
    quote=d|s|n        select double, single, no quotes for values
-t a|ad|d|dd|e|r|u|ud output format of time stamps (def: r: rel. to first)
-u s|hms              output format of seconds (def: s: seconds)
-l                   flush standard output after each packet
-q                   be more quiet on stdout (e.g. when using statistics)
-Q                   only log true errors to stderr (quieter than -q)
-g                   enable group read access on the output file(s)
-W n                 Save extra information in the file, if supported.
                        n = write network address resolution information
-X <key>:<value>       extension options, see the man page for details
-z <statistics>       various statistics, see the man page for details
--capture-comment <comment>
                        add a capture comment to the newly created
                        output file (only for pcapng)

Miscellaneous:
-h                   display this help and exit
-v                   display version info and exit
-o <name>:<value> ...  override preference setting
-K <keytab>          keytab file to use for kerberos decryption
-G [report]         dump one of several available reports and exit
                        default report="fields"

```

```
use "-G ?" for more help
```

```
WARNING: dumpcap will enable kernel BPF JIT compiler if available.
You might want to reset it
By doing "echo 0 > /proc/sys/net/core/bpf_jit_enable"
```

3. *tcpdump*: Capturing with `tcpdump` for viewing with Wireshark

It's often more useful to capture packets using `tcpdump` rather than `wireshark`. For example, you might want to do a remote capture and either don't have GUI access or don't have Wireshark installed on the remote machine.

Older versions of `tcpdump` truncate packets to 68 or 96 bytes. If this is the case, use `-s` to capture full-sized packets:

```
$ tcpdump -i <interface> -s 65535 -w <some-file>
```

You will have to specify the correct *interface* and the name of a *file* to save into. In addition, you will have to terminate the capture with `^C` when you believe you have captured enough packets.

`tcpdump` is not part of the Wireshark distribution. You can get it from <http://www.tcpdump.org/> or as a standard package in most Linux distributions.

4. *dumpcap*: Capturing with `dumpcap` for viewing with Wireshark

Dumpcap is a network traffic dump tool. It captures packet data from a live network and writes the packets to a file. Dumpcap's native capture file format is pcapng, which is also the format used by Wireshark.

Without any options set it will use the pcap library to capture traffic from the first available network interface and write the received raw packet data, along with the packets' time stamps into a pcapng file. The capture filter syntax follows the rules of the pcap library.

Help information available from dumpcap.

```
Dumpcap (Wireshark) 2.1.0 (v2.1.0rc0-502-g328fbc0 from master)
Capture network packets and dump them into a pcapng or pcap file.
See https://www.wireshark.org for more information.

Usage: dumpcap [options] ...

Capture interface:
  -i <interface>          name or idx of interface (def: first non-loopback),
                           or for remote capturing, use one of these formats:
                           rpcap://<host>/<interface>
                           TCP@<host>:<port>
  -f <capture filter>     packet filter in libpcap filter syntax
  -s <snaplen>            packet snapshot length (def: 65535)
  -p                      don't capture in promiscuous mode
  -I                      capture in monitor mode, if available
  -B <buffer size>        size of kernel buffer in MiB (def: 2MiB)
  -y <link type>          link layer type (def: first appropriate)
  -D                      print list of interfaces and exit
  -L                      print list of link-layer types of iface and exit
  -d                      print generated BPF code for capture filter
```

```

-k          set channel on wifi interface <freq>,[<type>]
-S          print statistics for each interface once per second
-M          for -D, -L, and -S, produce machine-readable output

RPCAP options:
-r          don't ignore own RPCAP traffic in capture
-u          use UDP for RPCAP data transfer
-A <user>:<password> use RPCAP password authentication
-m <sampling type> use packet sampling
            count:NUM - capture one packet of every NUM
            timer:NUM - capture no more than 1 packet in NUM ms

Stop conditions:
-c <packet count> stop after n packets (def: infinite)
-a <autostop cond.> ... duration:NUM - stop after NUM seconds
                        filesize:NUM - stop this file after NUM KB
                        files:NUM - stop after NUM files

Output (files):
-w <filename> name of file to save (def: tempfile)
-g          enable group read access on the output file(s)
-b <ringbuffer opt.> ... duration:NUM - switch to next file after NUM secs
                        filesize:NUM - switch to next file after NUM KB
                        files:NUM - ringbuffer: replace after NUM files
-n          use pcapng format instead of pcap (default)
-P          use libpcap format instead of pcapng
--capture-comment <comment>
            add a capture comment to the output file
            (only for pcapng)

Miscellaneous:
-N <packet_limit> maximum number of packets buffered within dumpcap
-C <byte_limit> maximum number of bytes used for buffering packets
                within dumpcap
-t          use a separate thread per interface
-q          don't report packet capture counts
-v          print version information and exit
-h          display this help and exit

```

WARNING: dumpcap will enable kernel BPF JIT compiler if available.

You might want to reset it

By doing "echo 0 > /proc/sys/net/core/bpf_jit_enable"

Example: dumpcap -i eth0 -a duration:60 -w output.pcapng

"Capture packets from interface eth0 until 60s passed into output.pcapng"

Use Ctrl-C to stop capturing at any time.

5. *capinfos*: Print information about capture files

`capinfos` can print information about binary capture files.

Help information available from `capinfos`.

```

Capinfos (Wireshark) 2.1.0 (v2.1.0rc0-502-g328fbc0 from master)
Print various information (infos) about capture files.
See https://www.wireshark.org for more information.

```

```
Usage: capinfos [options] <infile> ...
```

General infos:

```

-t display the capture file type
-E display the capture file encapsulation
-I display the capture file interface information
-F display additional capture file information
-H display the SHA1, RMD160, and MD5 hashes of the file
-k display the capture comment

```

```

Size infos:
-c display the number of packets
-s display the size of the file (in bytes)
-d display the total length of all packets (in bytes)
-l display the packet size limit (snapshot length)

Time infos:
-u display the capture duration (in seconds)
-a display the capture start time
-e display the capture end time
-o display the capture file chronological status (True/False)
-S display start and end times as seconds

Statistic infos:
-y display average data rate (in bytes/sec)
-i display average data rate (in bits/sec)
-z display average packet size (in bytes)
-x display average packet rate (in packets/sec)

Output format:
-L generate long report (default)
-T generate table report
-M display machine-readable values in long reports

Table report options:
-R generate header record (default)
-r do not generate header record

-B separate infos with TAB character (default)
-m separate infos with comma (,) character
-b separate infos with SPACE character

-N do not quote infos (default)
-q quote infos with single quotes (')
-Q quote infos with double quotes (")

Miscellaneous:
-h display this help and exit
-C cancel processing if file open fails (default is to continue)
-A generate all infos (default)

Options are processed from left to right order with later options superceding
or adding to earlier options.

If no options are given the default is to display all infos in long report
output format.

```

6. rawshark: Dump and analyze network traffic.

Rawshark reads a stream of packets from a file or pipe, and prints a line describing its output, followed by a set of matching fields for each packet on stdout.

Help information available from rawshark.

```

Rawshark (Wireshark) 2.1.0 (v2.1.0rc0-502-g328fbc0 from master)
Dump and analyze network traffic.
See https://www.wireshark.org for more information.

Usage: rawshark [options] ...

Input file:
  -r <infile>          set the pipe or file name to read from

Processing:

```

```

-d <encap:linktype>|<proto:protoname>
                                packet encapsulation or protocol
-F <field>                       field to display
-n                               disable all name resolution (def: all enabled)
-N <name resolve flags>         enable specific name resolution(s): "mnNtCd"
-p                               use the system's packet header format
                                (which may have 64-bit timestamps)
-R <read filter>                packet filter in Wireshark display filter syntax
-s                               skip PCAP header on input

Output:
-l                               flush output after each packet
-S                               format string for fields
                                (%D - name, %S - stringval, %N numval)
-t ad|a|r|d|dd|e               output format of time stamps (def: r: rel. to first)

Miscellaneous:
-h                               display this help and exit
-o <name>:<value> ...           override preference setting
-v                               display version info and exit

```

7. *editcap*: Edit capture files

`editcap` is a general-purpose utility for modifying capture files. Its main function is to remove packets from capture files, but it can also be used to convert capture files from one format to another, as well as to print information about capture files.

Help information available from `editcap`.

```

Editcap (Wireshark) 2.1.0 (v2.1.0rc0-502-g328fbc0 from master)
Edit and/or translate the format of capture files.
See https://www.wireshark.org for more information.

Usage: editcap [options] ... <infile> <outfile> [ <packet#>[-<packet#>] ... ]

<infile> and <outfile> must both be present.
A single packet or a range of packets can be selected.

Packet selection:
-r                               keep the selected packets; default is to delete them.
-A <start time>                 only output packets whose timestamp is after (or equal
                                to) the given time (format as YYYY-MM-DD hh:mm:ss).
-B <stop time>                  only output packets whose timestamp is before the
                                given time (format as YYYY-MM-DD hh:mm:ss).

Duplicate packet removal:
-d                               remove packet if duplicate (window == 5).
-D <dup window>                 remove packet if duplicate; configurable <dup window>
                                Valid <dup window> values are 0 to 1000000.
                                NOTE: A <dup window> of 0 with -v (verbose option) is
                                useful to print MD5 hashes.
-w <dup time window>            remove packet if duplicate packet is found EQUAL TO OR
                                LESS THAN <dup time window> prior to current packet.
                                A <dup time window> is specified in relative seconds
                                (e.g. 0.000001).
-a <framenum>:<comment>         Add or replace comment for given frame number

-I <bytes to ignore>            ignore the specified bytes at the beginning of
                                the frame during MD5 hash calculation
                                Useful to remove duplicated packets taken on
                                several routers(differents mac addresses for
                                example)
                                e.g. -I 26 in case of Ether/IP/ will ignore
                                ether(14) and IP header(20 - 4(src ip) - 4(dst ip)).

```


NOTE: The use of the 'Duplicate packet removal' options with other editcap options except -v may not always work as expected. Specifically the -r, -t or -S options will very likely NOT have the desired effect if combined with the -d, -D or -w.

Packet manipulation:

-s <snaplen> truncate each packet to max. <snaplen> bytes of data.

-C [offset:]<choplen> chop each packet by <choplen> bytes. Positive values chop at the packet beginning, negative values at the packet end. If an optional offset precedes the length, then the bytes chopped will be offset from that value. Positive offsets are from the packet beginning, negative offsets are from the packet end. You can use this option more than once, allowing up to 2 chopping regions within a packet provided that at least 1 choplen is positive and at least 1 is negative.

-L adjust the frame (i.e. reported) length when chopping and/or snapping

-t <time adjustment> adjust the timestamp of each packet; <time adjustment> is in relative seconds (e.g. -0.5).

-S <strict adjustment> adjust timestamp of packets if necessary to insure strict chronological increasing order. The <strict adjustment> is specified in relative seconds with values of 0 or 0.000001 being the most reasonable. A negative adjustment value will modify timestamps so that each packet's delta time is the absolute value of the adjustment specified. A value of -0 will set all packets to the timestamp of the first packet.

-E <error probability> set the probability (between 0.0 and 1.0 incl.) that a particular packet byte will be randomly changed.

-o <change offset> When used in conjunction with -E, skip some bytes from the beginning of the packet. This allows to preserve some bytes, in order to have some headers untouched.

Output File(s):

-c <packets per file> split the packet output to different files based on uniform packet counts with a maximum of <packets per file> each.

-i <seconds per file> split the packet output to different files based on uniform time intervals with a maximum of <seconds per file> each.

-F <capture type> set the output file type; default is pcapng. An empty "-F" option will list the file types.

-T <encap type> set the output file encapsulation type; default is the same as the input file. An empty "-T" option will list the encapsulation types.

Miscellaneous:

-h display this help and exit.

-v verbose output.

If -v is used with any of the 'Duplicate Packet Removal' options (-d, -D or -w) then Packet lengths and MD5 hashes are printed to standard-error.

Capture file types available from editcap -F .

```
$ editcap -F
editcap: option requires an argument -- 'F'
editcap: The available capture file types for the "-F" flag are:
  5views - InfoVista 5View capture
  btsnoop - Symbian OS btsnoop
  commview - TamoSoft CommView
  dct2000 - Catapult DCT2000 trace (.out format)
  erf - Endace ERF capture
  eyesdn - EyeSDN USB S0/E1 ISDN trace format
  k12text - K12 text file
  lanalyzer - Novell LANalyzer
```

```

logcat - Android Logcat Binary format
logcat-brief - Android Logcat Brief text format
logcat-long - Android Logcat Long text format
logcat-process - Android Logcat Process text format
logcat-tag - Android Logcat Tag text format
logcat-thread - Android Logcat Thread text format
logcat-threadtime - Android Logcat Threadtime text format
logcat-time - Android Logcat Time text format
modlibpcap - Modified tcpdump - libpcap
netmon1 - Microsoft NetMon 1.x
netmon2 - Microsoft NetMon 2.x
nettl - HP-UX nettl trace
ngsniffer - Sniffer (DOS)
ngwsniffer_1_1 - NetXray, Sniffer (Windows) 1.1
ngwsniffer_2_0 - Sniffer (Windows) 2.00x
niobserver - Network Instruments Observer
nokialibpcap - Nokia tcpdump - libpcap
nseclibpcap - Wireshark - nanosecond libpcap
nstrace10 - NetScaler Trace (Version 1.0)
nstrace20 - NetScaler Trace (Version 2.0)
nstrace30 - NetScaler Trace (Version 3.0)
nstrace35 - NetScaler Trace (Version 3.5)
pcap - Wireshark/tcpdump/... - pcap
pcapng - Wireshark/... - pcapng
rf5 - Tektronix K12xx 32-bit .rf5 format
rh6_libpcap - RedHat 6.1 tcpdump - libpcap
snoop - Sun snoop
suse6_3libpcap - SuSE 6.3 tcpdump - libpcap
visual - Visual Networks traffic capture

```

Encapsulation types available from editcap.

```

$ editcap -T
editcap: option requires an argument -- 'T'
editcap: The available encapsulation types for the "-T" flag are:
  ap1394 - Apple IP-over-IEEE 1394
  arcnet - ARCNET
  arcnet_linux - Linux ARCNET
  ascend - Lucent/Ascend access equipment
  atm-pdus - ATM PDUs
  atm-pdus-untruncated - ATM PDUs - untruncated
  atm-rfc1483 - RFC 1483 ATM
  ax25 - Amateur Radio AX.25
  ax25-kiss - AX.25 with KISS header
  bacnet-ms-tp - BACnet MS/TP
  bacnet-ms-tp-with-direction - BACnet MS/TP with Directional Info
  ber - ASN.1 Basic Encoding Rules
  bluetooth-bredr-bb-rf - Bluetooth BR/EDR Baseband RF
  bluetooth-h4 - Bluetooth H4
  bluetooth-h4-linux - Bluetooth H4 with linux header
  bluetooth-hci - Bluetooth without transport layer
  bluetooth-le-ll - Bluetooth Low Energy Link Layer
  bluetooth-le-ll-rf - Bluetooth Low Energy Link Layer RF
  bluetooth-linux-monitor - Bluetooth Linux Monitor
  can20b - Controller Area Network 2.0B
  chdlc - Cisco HDLC
  chdlc-with-direction - Cisco HDLC with Directional Info
  cosine - CoSine L2 debug log
  dbus - D-Bus
  dct2000 - Catapult DCT2000
  docsis - Data Over Cable Service Interface Specification
  dpnss_link - Digital Private Signalling System No 1 Link Layer
  dvbci - DVB-CI (Common Interface)
  enc - OpenBSD enc(4) encapsulating interface
  epon - Ethernet Passive Optical Network
  erf - Extensible Record Format
  ether - Ethernet

```

```
ether-nettl - Ethernet with nettl headers
fc2 - Fibre Channel FC-2
fc2sof - Fibre Channel FC-2 With Frame Delimiter
fddi - FDDI
fddi-nettl - FDDI with nettl headers
fddi-swapped - FDDI with bit-swapped MAC addresses
flexray - FlexRay
frelay - Frame Relay
frelay-with-direction - Frame Relay with Directional Info
gcom-serial - GCOM Serial
gcom-tie1 - GCOM TIE1
gprs-llc - GPRS LLC
gsm_um - GSM Um Interface
hhdllc - HiPath HDLC
i2c - I2C
ieee-802-11 - IEEE 802.11 Wireless LAN
ieee-802-11-airopeek - IEEE 802.11 plus AiroPeek radio header
ieee-802-11-avs - IEEE 802.11 plus AVS radio header
ieee-802-11-netmon - IEEE 802.11 plus Network Monitor radio header
ieee-802-11-prism - IEEE 802.11 plus Prism II monitor mode radio header
ieee-802-11-radio - IEEE 802.11 Wireless LAN with radio information
ieee-802-11-radiotap - IEEE 802.11 plus radiotap radio header
ieee-802-16-mac-cps - IEEE 802.16 MAC Common Part Sublayer
infiniband - InfiniBand
ios - Cisco IOS internal
ip-over-fc - RFC 2625 IP-over-Fibre Channel
ip-over-ib - IP over Infiniband
ipfix - IPFIX
ipmb - Intelligent Platform Management Bus
ipmi-trace - IPMI Trace Data Collection
ipnet - Solaris IPNET
irda - IrDA
isdn - ISDN
ixveriwave - IxVeriWave header and stats block
jfif - JPEG/JFIF
json - JavaScript Object Notation
juniper-atm1 - Juniper ATM1
juniper-atm2 - Juniper ATM2
juniper-chdlc - Juniper C-HDLC
juniper-ether - Juniper Ethernet
juniper-frelay - Juniper Frame-Relay
juniper-ggsn - Juniper GGSN
juniper-mlfr - Juniper MLFR
juniper-mlppp - Juniper MLPPP
juniper-ppp - Juniper PPP
juniper-pppoe - Juniper PPPoE
juniper-svcs - Juniper Services
juniper-vp - Juniper Voice PIC
k12 - K12 protocol analyzer
lapb - LAPB
lapd - LAPD
layer1-event - EyeSDN Layer 1 event
lin - Local Interconnect Network
linux-atm-clip - Linux ATM CLIP
linux-lapd - LAPD with Linux pseudo-header
linux-sll - Linux cooked-mode capture
logcat - Android Logcat Binary format
logcat_brief - Android Logcat Brief text format
logcat_long - Android Logcat Long text format
logcat_process - Android Logcat Process text format
logcat_tag - Android Logcat Tag text format
logcat_thread - Android Logcat Thread text format
logcat_threadtime - Android Logcat Threadtime text format
logcat_time - Android Logcat Time text format
loop - OpenBSD loopback
ltalk - Localtalk
mime - MIME
most - Media Oriented Systems Transport
mp2ts - ISO/IEC 13818-1 MPEG2-TS
```

```
mpeg - MPEG
mtp2 - SS7 MTP2
mtp2-with-phdr - MTP2 with pseudoheader
mtp3 - SS7 MTP3
mux27010 - MUX27010
netanalyzer - netANALYZER
netanalyzer-transparent - netANALYZER-Transparent
netlink - Linux Netlink
nfc-llcp - NFC LLCp
nflog - NFLOG
nstrace10 - NetScaler Encapsulation 1.0 of Ethernet
nstrace20 - NetScaler Encapsulation 2.0 of Ethernet
nstrace30 - NetScaler Encapsulation 3.0 of Ethernet
nstrace35 - NetScaler Encapsulation 3.5 of Ethernet
null - NULL/Loopback
packetlogger - PacketLogger
pflog - OpenBSD PF Firewall logs
pflog-old - OpenBSD PF Firewall logs, pre-3.4
pktap - Apple PKTAP
ppi - Per-Packet Information header
ppp - PPP
ppp-with-direction - PPP with Directional Info
pppoes - PPP-over-Ethernet session
raw-icmp-nettl - Raw ICMP with nettl headers
raw-icmpv6-nettl - Raw ICMPv6 with nettl headers
raw-telnet-nettl - Raw telnet with nettl headers
rawip - Raw IP
rawip-nettl - Raw IP with nettl headers
rawip4 - Raw IPv4
rawip6 - Raw IPv6
redback - Redback SmartEdge
rtac-serial - RTAC serial-line
s4607 - STANAG 4607
s5066-dpdu - STANAG 5066 Data Transfer Sublayer PDUs(D_PDU)
sccp - SS7 SCCP
sctp - SCTP
sdh - SDH
sdlc - SDLC
sita-wan - SITA WAN packets
slip - SLIP
socketcan - SocketCAN
symantec - Symantec Enterprise Firewall
tnef - Transport-Neutral Encapsulation Format
tr - Token Ring
tr-nettl - Token Ring with nettl headers
tzsp - Tazmen sniffer protocol
unknown - Unknown
unknown-nettl - Unknown link-layer type with nettl headers
usb - Raw USB packets
usb-linux - USB packets with Linux header
usb-linux-mmap - USB packets with Linux header and padding
usb-usbpcap - USB packets with USBPcap header
user0 - USER 0
user1 - USER 1
user2 - USER 2
user3 - USER 3
user4 - USER 4
user5 - USER 5
user6 - USER 6
user7 - USER 7
user8 - USER 8
user9 - USER 9
user10 - USER 10
user11 - USER 11
user12 - USER 12
user13 - USER 13
user14 - USER 14
user15 - USER 15
v5-ef - V5 Envelope Function
```

```

whd1c - Wellfleet HDLC
wireshark-upper-pdu - Wireshark Upper PDU export
wpan - IEEE 802.15.4 Wireless PAN
wpan-nofcs - IEEE 802.15.4 Wireless PAN with FCS not present
wpan-nonask-phy - IEEE 802.15.4 Wireless PAN non-ASK PHY
x2e-serial - X2E serial line capture
x2e-xoraya - X2E Xoraya
x25-nettl - X.25 with nettl headers

```

8. *mergecap*: Merging multiple capture files into one

Mergecap is a program that combines multiple saved capture files into a single output file specified by the `-w` argument. Mergecap knows how to read libpcap capture files, including those of tcpdump. In addition, Mergecap can read capture files from snoop (including Shomiti) and atmsnoop, LanAlyzer, Sniffer (compressed or uncompressed), Microsoft Network Monitor, AIX's iptrace, NetXray, Sniffer Pro, RADCOM's WAN/LAN analyzer, Lucent/Ascend router debug output, HP-UX's nettl, and the dump output from Toshiba's ISDN routers. There is no need to tell Mergecap what type of file you are reading; it will determine the file type by itself. Mergecap is also capable of reading any of these file formats if they are compressed using `gzip`. Mergecap recognizes this directly from the file; the ".gz" extension is not required for this purpose.

By default, it writes the capture file in pcapng format, and writes all of the packets in the input capture files to the output file. The `-F` flag can be used to specify the format in which to write the capture file; it can write the file in libpcap format (standard libpcap format, a modified format used by some patched versions of libpcap, the format used by Red Hat Linux 6.1, or the format used by SuSE Linux 6.3), snoop format, uncompressed Sniffer format, Microsoft Network Monitor 1.x format, and the format used by Windows-based versions of the Sniffer software.

Packets from the input files are merged in chronological order based on each frame's timestamp, unless the `-a` flag is specified. Mergecap assumes that frames within a single capture file are already stored in chronological order. When the `-a` flag is specified, packets are copied directly from each input file to the output file, independent of each frame's timestamp.

If the `-s` flag is used to specify a snapshot length, frames in the input file with more captured data than the specified snapshot length will have only the amount of data specified by the snapshot length written to the output file. This may be useful if the program that is to read the output file cannot handle packets larger than a certain size (for example, the versions of snoop in Solaris 2.5.1 and Solaris 2.6 appear to reject Ethernet frames larger than the standard Ethernet MTU, making them incapable of handling gigabit Ethernet captures if jumbo frames were used).

If the `-T` flag is used to specify an encapsulation type, the encapsulation type of the output capture file will be forced to the specified type, rather than being the type appropriate to the encapsulation type of the input capture file. Note that this merely forces the encapsulation type of the output file to be the specified type; the packet headers of the packets will not be translated from the encapsulation type of the input capture file to the specified encapsulation type (for example, it will not translate an Ethernet capture to an FDDI capture if an Ethernet capture is read and `-T fddi` is specified).

Help information available from mergecap.

```

Mergecap (Wireshark) 2.1.0 (v2.1.0rc0-502-g328fbc0 from master)
Merge two or more capture files into one.
See https://www.wireshark.org for more information.

Usage: mergecap [options] -w <outfile>|- <infile> [<infile> ...]

Output:
  -a                concatenate rather than merge files.
                   default is to merge based on frame timestamps.
  -s <snaplen>     truncate packets to <snaplen> bytes of data.
  -w <outfile>|-   set the output filename to <outfile> or '-' for stdout.

```

```
-F <capture type> set the output file type; default is pcapng.
                    an empty "-F" option will list the file types.
-I <IDB merge mode> set the merge mode for Interface Description Blocks; default is 'all'.
                    an empty "-I" option will list the merge modes.
```

Miscellaneous:

```
-h                display this help and exit.
-v                verbose output.
```

A simple example merging `dhcp-capture.pcapng` and `imap-1.pcapng` into `outfile.pcapng` is shown below.

Simple example of using mergecap.

```
$ mergecap -w outfile.pcapng dhcp-capture.pcapng imap-1.pcapng
```

9. *text2pcap*: Converting ASCII hexdumps to network captures

There may be some occasions when you wish to convert a hex dump of some network traffic into a libpcap file.

`text2pcap` is a program that reads in an ASCII hex dump and writes the data described into a libpcap-style capture file. `text2pcap` can read hexdumps with multiple packets in them, and build a capture file of multiple packets.

`text2pcap` is also capable of generating dummy Ethernet, IP and UDP headers, in order to build fully processable packet dumps from hexdumps of application-level data only.

`text2pcap` understands a hexdump of the form generated by `od -A x -t x1`. In other words, each byte is individually displayed and surrounded with a space. Each line begins with an offset describing the position in the file. The offset is a hex number (can also be octal - see `-o`), of more than two hex digits. Here is a sample dump that `text2pcap` can recognize:

```
000000 00 e0 1e a7 05 6f 00 10 .....
000008 5a a0 b9 12 08 00 46 00 .....
000010 03 68 00 00 00 00 0a 2e .....
000018 ee 33 0f 19 08 7f 0f 19 .....
000020 03 80 94 04 00 00 10 01 .....
000028 16 a2 0a 00 03 50 00 0c .....
000030 01 01 0f 19 03 80 11 01 .....
```

There is no limit on the width or number of bytes per line. Also the text dump at the end of the line is ignored. Bytes/hex numbers can be uppercase or lowercase. Any text before the offset is ignored, including email forwarding characters '>'. Any lines of text between the bytestring lines is ignored. The offsets are used to track the bytes, so offsets must be correct. Any line which has only bytes without a leading offset is ignored. An offset is recognized as being a hex number longer than two characters. Any text after the bytes is ignored (e.g. the character dump). Any hex numbers in this text are also ignored. An offset of zero is indicative of starting a new packet, so a single text file with a series of hexdumps can be converted into a packet capture with multiple packets. Multiple packets are read in with timestamps differing by one second each. In general, short of these restrictions, `text2pcap` is pretty liberal about reading in hexdumps and has been tested with a variety of mangled outputs (including being forwarded through email multiple times, with limited line wrap etc.)

There are a couple of other special features to note. Any line where the first non-whitespace character is `#` will be ignored as a comment. Any line beginning with `#TEXT2PCAP` is a directive and options can be inserted after this command to be processed by `text2pcap`. Currently there are no directives implemented; in the future, these may be used to give more fine grained control on the dump and the way it should be processed e.g. timestamps, encapsulation type etc.

`text2pcap` also allows the user to read in dumps of application-level data, by inserting dummy L2, L3 and L4 headers before each packet. Possibilities include inserting headers such as Ethernet, Ethernet + IP, Ethernet + IP + UDP, or Ethernet + IP + TCP before each packet. This allows Wireshark or any other full-packet decoder to handle these dumps.

Help information available from `text2pcap`.

```
Text2pcap (Wireshark) 2.1.0 (v2.1.0rc0-502-g328fbc0 from master)
Generate a capture file from an ASCII hexdump of packets.
See https://www.wireshark.org for more information.

Usage: text2pcap [options] <infile> <outfile>

where <infile> specifies input filename (use - for standard input)
      <outfile> specifies output filename (use - for standard output)

Input:
  -o hex|oct|dec      parse offsets as (h)ex, (o)ctal or (d)ecimal;
                      default is hex.
  -t <timefmt>        treat the text before the packet as a date/time code;
                      the specified argument is a format string of the sort
                      supported by strptime.
                      Example: The time "10:15:14.5476" has the format code
                      "%H:%M:%S."
                      NOTE: The subsecond component delimiter, '.', must be
                      given, but no pattern is required; the remaining
                      number is assumed to be fractions of a second.
                      NOTE: Date/time fields from the current date/time are
                      used as the default for unspecified fields.
  -D                  the text before the packet starts with an I or an O,
                      indicating that the packet is inbound or outbound.
                      This is only stored if the output format is PCAP-NG.
  -a                  enable ASCII text dump identification.
                      The start of the ASCII text dump can be identified
                      and excluded from the packet data, even if it looks
                      like a HEX dump.
                      NOTE: Do not enable it if the input file does not
                      contain the ASCII text dump.

Output:
  -l <typenum>        link-layer type number; default is 1 (Ethernet). See
                      http://www.tcpdump.org/linktypes.html for a list of
                      numbers. Use this option if your dump is a complete
                      hex dump of an encapsulated packet and you wish to
                      specify the exact type of encapsulation.
                      Example: -l 7 for ARCNet packets.
  -m <max-packet>    max packet length in output; default is 65535

Prepend dummy header:
  -e <l3pid>          prepend dummy Ethernet II header with specified L3PID
                      (in HEX).
                      Example: -e 0x806 to specify an ARP packet.
  -i <proto>          prepend dummy IP header with specified IP protocol
                      (in DECIMAL).
                      Automatically prepends Ethernet header as well.
                      Example: -i 46
  -4 <srcip>,<destip> prepend dummy IPv4 header with specified
                      dest and source address.
                      Example: -4 10.0.0.1,10.0.0.2
  -6 <srcip>,<destip> replace IPv6 header with specified
                      dest and source address.
                      Example: -6 fe80:0:0:0:202:b3ff:fe1e:8329,2001:0db8:85a3:0000:0000:8a2e:0370:7334
  -u <srcp>,<destp>  prepend dummy UDP header with specified
                      source and destination ports (in DECIMAL).
                      Automatically prepends Ethernet & IP headers as well.
                      Example: -u 1000,69 to make the packets look like
                      TFTP/UDP packets.
```

```

-T <srcp>,<destp>      prepend dummy TCP header with specified
                        source and destination ports (in DECIMAL).
                        Automatically prepends Ethernet & IP headers as well.
                        Example: -T 50,60
-s <srcp>,<dstp>,<tag>  prepend dummy SCTP header with specified
                        source/dest ports and verification tag (in DECIMAL).
                        Automatically prepends Ethernet & IP headers as well.
                        Example: -s 30,40,34
-S <srcp>,<dstp>,<ppi>  prepend dummy SCTP header with specified
                        source/dest ports and verification tag 0.
                        Automatically prepends a dummy SCTP DATA
                        chunk header with payload protocol identifier ppi.
                        Example: -S 30,40,34

```

Miscellaneous:

```

-h          display this help and exit.
-d          show detailed debug of parser states.
-q          generate no output at all (automatically disables -d).
-n          use PCAP-NG instead of PCAP as output format.

```

10. *reordercap*: Reorder a capture file

`reordercap` lets you reorder a capture file according to the packets timestamp.

Help information available from `reordercap`.

```

Reordercap (Wireshark) 2.1.0 (v2.1.0rc0-502-g328fbc0 from master)
Reorder timestamps of input file frames into output file.
See https://www.wireshark.org for more information.

```

```
Usage: reordercap [options] <infile> <outfile>
```

Options:

```

-n          don't write to output file if the input file is ordered.
-h          display this help and exit.

```

Sources

- https://www.wireshark.org/docs/wsug_html/#AppTools

Programme de formation SIP

- [Sommaire du chapitre](#)
- [Programme de formation](#)
 - [Architecture de SIP](#)
 - [Processus d'enregistrement](#)
 - [Session SIP](#)
 - [Les extensions SIP](#)
 - [SIP et la sécurité](#)

Sommaire du chapitre

- [1. Architecture SIP](#)
- [2. Aperçu des opérations SIP](#)
- [3. INVITE SIP UAC/UAS](#)
- [4. Réponses SIP](#)
- [5. SDP](#)
- [6. Enregistrement REGISTER](#)
- [7. Proxy SIP UDP](#)
- [8. B2BUA](#)
- [9. Flux SIP Trapéziodal](#)
- [10. Extensions SIP](#)
- [11. Sécurité SIP](#)
- [12. RFCs SIP](#)

Programme de formation

Le canevas de ce support de formation est le suivant.

Architecture de SIP

Rappel sur la téléphonie classique

- Le réseau PSTN
- Les composants d'un réseau VoIP
- Les clients
- Les passerelles voix

La signalisation

- Comparaison à H.323
- Les serveurs d'application
- Les composants spécifiques au protocole SIP
 - Les UA
 - UAC
 - UAS
- Les proxy
- Les registrar

Les messages SIP

- INVITE
- REGISTER
- ACK
- BYE
- CANCEL
- UPDATE

Les codes de statuts

- Codes provisionnels
- Codes de réussite
- Codes de redirection
- Code d'échec client
- Code d'échec serveur
- Code d'échec global

Le protocole SDP

La qualité de service

Processus d'enregistrement

- Enregistrement basique
- Messages de notification
- Les WMI
- Configuration du registrar
- Configuration des clients
- Validation de l'enregistrement

Session SIP

- Initiation du dialogue
- Requête client
- Réponse du serveur
- Routage de l'appel
- Perte du routage
- Routage strict
- Modification de la session SIP
- Termination de la session SIP
- Utilisation d'un proxy
- Connexion au réseau PSTN
- Gestion de la présence

Les extensions SIP

- Principe et rôles des extensions
 - Info
 - Comet
 - Subscribe
 - Publish
 - Notify
 - Message

- Refer
- Prack

SIP et la sécurité

- Support du NAT
- Utilisation de Firewall
- Les types d'attaques sur les réseaux convergents
- Le hijacking
- Dénî de service
- Amplification
- Bots
- Les mesures de sécurité
- Contrôle des accès
- Cryptage
- Authentification
- Gestion des autorisations
- Systèmes de détection d'intrusion

Architecture SIP

- 1. Protocole SIP
- 2. La boîte-à-outils SIP
- 3. AOR
- 4. Rôles SIP
 - User Agents (UA)
 - Proxy
 - Serveur de redirection
 - B2BUA
 - REGISTRAR Server et Location Server
 - SBC
 - Gateways
- 5. Requêtes (Méthodes) SIP
- 6. Réponses (Status Codes) SIP
- 7. Messages SIP
 - Exemple d'une transaction BYE-200
- 8. Scénarios SIP
 - Processus d'enregistrement
 - Flux d'appel SIP entre UA et serveurs de redirection entre proxys et UAs
 - Flux d'appel B2BUA
- 9. UA en ligne de commande
 - Installation PJSUA
 - Exemple d'usage de PJSUA
 - Installation du dépôt Linphone
 - linphonecsh
 - linphonec

Dans cette section, on trouvera une description du Protocole SIP comme boîte-à-outils. On décrira le concept d'AOR mais aussi on donnera une idée des rôles (UA, proxies, B2BUA), des requêtes (méthodes), des réponses et des messages du protocole SIP. Enfin, on envisagera quelques scénarios classiques à étudier.

1. Protocole SIP

Session Initiation Protocol (SIP) est un protocole TCP/IP de couche application normalisé et standardisé par l'IETF ([RFC 3261](#)). Il a été conçu pour établir, modifier et terminer des sessions multimédia. Il prend en charge l'authentification et la localisation de multiples participants. S'il se charge de la négociation des médias, il laisse le soin à d'autres protocoles de transporter du texte, de la voix ou de la vidéo.

SIP fonctionne aussi bien avec IPv4 qu'avec IPv6. SIP est supporté par TCP ou UDP sur le port 5060 par défaut. La version sécurisée SIP-TLS utilise par défaut le port TCP 5061.

SIP prend en charge cinq facettes de l'établissement et de la terminaison de communications multimédia :

- **Localisation de l'utilisateur** : détermination du système terminal à utiliser pour la communication ;
- **Disponibilité de l'utilisateur** : détermination de la volonté de l'appelé à s'engager dans une communication ;
- **Capacités de l'utilisateur** : détermination du support et des paramètres de support à utiliser ;
- **Etablissement de session** : "sonnerie", établissement des paramètres de session à la fois chez l'appelant et l'appelé ;
- **Gestion de session** : y compris le transfert et la terminaison des sessions, la modification des paramètres de

session, et l'invocation des services.

SIP n'est pas un système de communications intégré verticalement. SIP est plutôt un composant qui peut être utilisé avec d'autres protocoles de l'IETF pour construire une architecture multimédia complète.

SIP ne fournit pas de services. Plus justement, SIP fournit des primitives qui peuvent être utilisées pour mettre en œuvre différents services. Par exemple, SIP peut localiser un utilisateur et livrer un objet opaque à l'endroit où il se trouve. Une seule primitive est normalement utilisée pour fournir plusieurs services différents.

La nature des services fournis rend la sécurité particulièrement importante. A cette fin, SIP fournit une série de services de sécurité, qui comporte la prévention du déni de service, l'authentification (à la fois d'usager à usager et de mandataire à usager), la protection de l'intégrité, et de services de chiffrement et de confidentialité.

2. La boîte-à-outils SIP

SIP est donc un protocole de l'IETF (il aura la préférence du marché) qui est une véritable boîte-à-outils (primitives SIP) pour établir des communications à travers l'Internet. Ses illustrations les plus immédiates sont celles de la téléphonie IP, de la voix sur IP et puis des systèmes de messageries tels que Skype, WhatsApp, etc. Mais on peut y trouver d'autres cas d'usage où des communications vocales ou vidéos peuvent intervenir dans le cadre de services de support, de fourniture de connectivité ou encore dans l'IoT ou la domotique.

Une expérience SIP pleinement vécue ne se passe pas d'autres protocoles TCP/IP, de logiciels et de leurs librairies qui le mettent en œuvre. Aussi, SIP n'est plus nécessairement le seul protocole ou l'élément central des solutions commerciales qui nous sont offertes sur le marché.

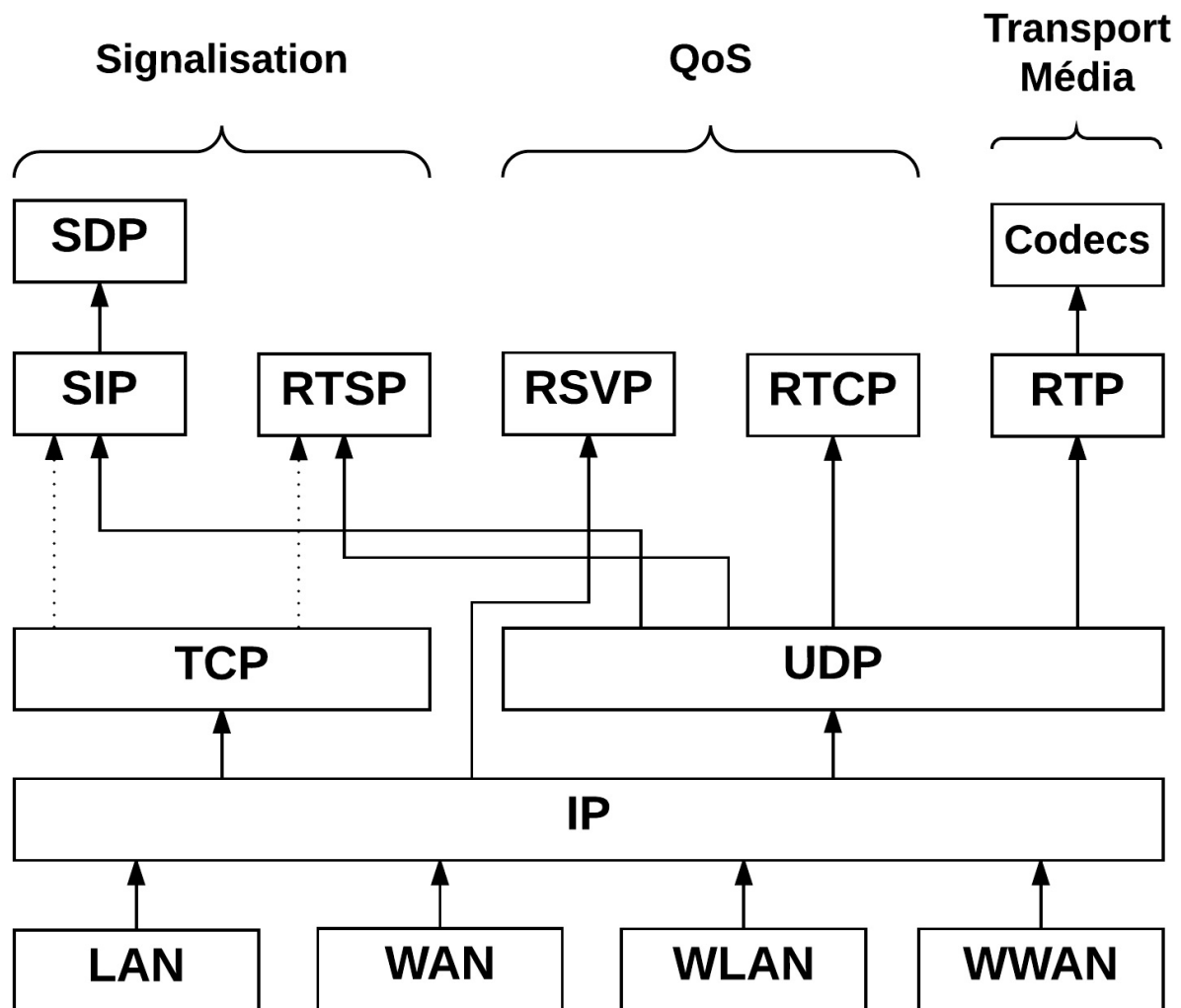


Figure : Protocoles VoIP

A cet égard, la mise en oeuvre du protocole peut provoquer des difficultés d'interconnexion entre des périphériques non supportés entre eux ; les passerelles Cisco Systems a la faveur du marché et des opérateurs pour éviter cet écueil. Egalement, on prendra garde de ne pas confondre le message SIP rendu par l'application et l'erreur de l'application qui génère le message d'erreur. Enfin, comme boîte-à-outils, SIP aura notre préférence dans sa version open-source.

Comme protocole IETF, SIP est normalisé et standardisé IETF. Le document [RFCs SIP](#) reprend les principales RFCs relatives à SIP (il y en a plusieurs centaines) :

- Core SIP Documents
- SDP-Related Documents
- RTP-Related Documents
- HTTP-Related Documents
- MIME-Related Documents
- SIP Standards Track Documents (Options, Extensions, etc.)
- SIP Informational RFCs and BCP Documents
- SIP-Related Documents
- Directory Services Documents

3. AOR

Adresse d'enregistrement (AOR, Address-of-Record) : Une adresse d'enregistrement est un URI SIP ou SIPS qui pointe sur un domaine avec un service de localisation qui peut transposer l'URI en un autre URI où l'utilisateur pourrait être disponible. Normalement, le service de localisation est rempli au moyen des enregistrements. Un AOR est fréquemment vu comme l'"adresse publique" de l'utilisateur.

4. Rôles SIP

On trouvera différents rôles logiques en SIP.

User Agents (UA)

UA qui caractérisent les points d'extrémités de la communication ou un B2BUA.

- **UAC** : Un client est tout élément de réseau qui envoie une demande SIP et reçoit des réponses SIP. Les clients peuvent ou non interagir directement avec un utilisateur humain. Les clients et proxys d'UA sont des clients.
- **UAS** : C'est une entité logique qui génère une réponse à une demande SIP. La réponse accepte, rejette, ou redirige la demande. Ce rôle ne dure que pendant le temps de cette transaction. En d'autres termes, si un logiciel répond à une demande, il agit comme UAS pour la durée de cette transaction. Si il génère plus tard une demande, il assume le rôle d'un UAC pour le traitement de cette transaction.

Proxy

Serveur Proxy (mandataire) : Entité intermédiaire qui agit à la fois comme un serveur et comme un client pour les besoins de l'élaboration de demandes au nom des autres clients.

Un serveur proxy joue principalement un rôle d'acheminement, de routage, ce qui signifie que sa tâche est de s'assurer qu'une demande est envoyée à une autre entité "plus proche" de l'utilisateur cible. En cela il est comparable au routeur IP qui transfère le trafic en fonction de l'adresse IP de destination.

Les proxys sont aussi utiles pour mettre en application la politique de routage des appels (par exemple, s'assurer qu'un utilisateur est autorisé à effectuer un appel).

Un proxy interprète et, si cela est nécessaire, réécrit des parties spécifiques d'un message de demande avant de le retransmettre.

Outbound Proxy : Proxy qui reçoit des demandes de la part d'un client, même si il peut ne pas être le serveur donné par la résolution de l'URI demandée. Normalement un UA est configuré manuellement avec un mandataire de sortie, ou il peut en acquérir la connaissance au moyen de protocoles d'autoconfiguration.

Les notions de **Stateful Proxy** et de **Stateless Proxy** se distinguent par le maintien d'un état des transactions. Alors que le proxy Stateless ne maintient aucun état, le proxy stateful peut accomplir des tâches plus complexes telles que la duplication des transactions, l'absorption des transactions ou d'autres telles que le transfert d'appel.

Serveur de redirection

C'est un serveur d'agent d'utilisateur (UAS) qui génère des réponses 3xx (Redirection) aux demandes qu'il reçoit, amenant le client à contacter un ensemble d'URI de remplacement.

B2BUA

B2BUA, Back-to-Back User Agent : Un B2BUA est une entité logique qui reçoit une demande et la traite comme UAS. Afin de déterminer comment il devrait être répondu à la demande, il agit comme un UAC et génère des demandes. A la différence d'un serveur mandataire, il maintient l'état du dialogue et doit participer à toutes les

demandes envoyées dans les dialogues qu'il a établi. Comme c'est un enchaînement d'UAC et d'UAS, aucune définition explicite n'est nécessaire pour son comportement.

REGISTRAR Server et Location Server

Un **REGISTRAR Server** un serveur qui gère les requêtes REGISTER envoyées par les Users Agents pour signaler leur emplacement courant. Ces requêtes contiennent donc une adresse IP, associée à une URI, qui seront stockées dans une base de données.

Un service de localisation est utilisé par un Redirect Server ou un serveur proxy pour obtenir des informations sur la ou les localisations possibles d'un appelé. Il contient une liste de liens de clés d'address-of-record pour aucune ou plusieurs adresses de contact. Les liens peuvent être créés et retirés de nombreuses façons.

Les URI SIPs sont très similaires dans leur forme à des adresses email : `sip:utilisateur@domaine.com`

Généralement, des mécanismes d'authentification permettent d'éviter que quiconque puisse s'enregistrer avec n'importe quelle URI.

SBC

Un **SBC (Session border controller)** est placé comme élément intermédiaire pour rendre des services entre les UA et les serveurs SIP en matière de sécurité, de camouflage de topologie, de filtrage ou d'assistance dans du NAT Traversal ou encore de chiffrement du trafic.

Gateways

Les **Gateways (passerelles)** sont des entités logiques qui sont capables d'établir des liaisons vers des destinations non-IP notamment les réseaux PSTN.

5. Requêtes (Méthodes) SIP

Le client envoie des requêtes au serveur, qui lui renvoie une réponse. Les méthodes de base ([RFC 3261](#)) comprises dans ces requêtes sont :

- INVITE : permet à un client de demander une nouvelle session,
- ACK : confirme l'établissement de la session,
- CANCEL : annule un INVITE en suspens,
- BYE : termine une session en cours,
- OPTIONS : permet de récupérer les capacités de gestion des usagers, sans ouvrir de session,
- REGISTER : permet de s'enregistrer auprès d'un serveur d'enregistrement.

D'autres RFC sont venus compléter les capacités du protocole avec de nouvelles méthodes :

- PRACK : "Provisional acknowledgement" ([RFC 3262](#))
- SUBSCRIBE : "Subscribes for an Event of Notification from the Notifier" ([RFC 6665](#))
- NOTIFY : "Notify the subscriber of a new Event" ([RFC 6665](#))
- PUBLISH : "Publishes an event to the Server" ([RFC 3903](#))
- INFO : "Sends mid-session information that does not modify the session state" ([RFC 6086](#))
- REFER : "Asks recipient to issue SIP request (call transfer.)" ([RFC 3515](#))
- MESSAGE : "Transports instant messages using SIP" ([RFC 3428](#))
- UPDATE : "Modifies the state of a session without changing the state of the dialog" ([RFC 3311](#))

6. Réponses (Status Codes) SIP

Les réponses SIP sont présentées dans la page [Réponses SIP](#).

Selon les contextes une réponse est attendue, les réponses possibles sont similaires aux réponses HTTP. Dans le meilleur des cas on obtient un 200 OK.

Une transaction SIP survient entre un client et un serveur et comprend tous les messages depuis la première demande envoyée du client au serveur jusqu'à la réponse finale (non-1xx) envoyée du serveur au client. Si la demande est INVITE et la réponse finale est un non-2xx, la transaction comporte aussi un ACK à la réponse. Le ACK pour une réponse 2xx à une demande INVITE est une transaction distincte.

Un dialogue est une relation SIP d'homologue à homologue qui persiste pendant un certain temps entre deux agents d'utilisateur. Un dialogue est établi par les messages SIP, comme une réponse 2xx à une demande INVITE.

!!! attention Un dialogue est identifié par un identifiant d'appel, une étiquette locale, et une étiquette distante.

- Provisional (1xx): La demande est reçue et est en cours de traitement.
- Success (2xx): L'action a été reçue, comprise et acceptée avec succès.
- Redirection (3xx): Une action supplémentaire doit être prise (par l'appelant) pour compléter la demande.
- Client Error (4xx): La demande comporte une mauvaise syntaxe et ne peut être prise en charge par le serveur.
- Server Error (5xx): Le serveur a échoué remplir une demande apparemment valide.
- Global Failure (6xx): La demande ne peut être prise en charge par aucun serveur.

7. Messages SIP

Un message SIP est composé des éléments suivants :

1. Une ligne de départ (start-line) : Message URI SIP/2.0
2. Un ou plusieurs champs d'en-tête (header fields)
3. Une ligne vide indiquant la fin des champs d'en-tête.
4. Un corps de message optionnel (message body)

Un message SIP est notamment composé de champs d'en-têtes définis dans le [RFC 3261](#) pour la signalisation et le routage des informations entre des entités SIP. SIP utilise le même format que celui qui définit un en-tête HTTP ([RFC 2616](#)). Chaque en-tête consiste en un nom de champs suivant d'un deux-points (:) et d'une valeur.

Les champs d'en-tête (header fields) peuvent être, entre autres :

1. **From** : Il indique l'identité de celui qui initié la requête SIP. Cette valeur est souvent valorisée par l'AOR de l'expéditeur. Il comprend un URI SIP ou SIPS voire un "display name" optionnel.
2. **To** : Cet en-tête indique le destinataire souhaité pour la requête SIP. Il utilise habituellement l'AOR du destinataire.
3. **Call-ID** : Il identifie un dialogue SIP de manière unique. Il est donc identique pour toutes les requêtes et les réponses SIP d'un même dialogue.
4. **Cseq** : Cet en-tête est composé d'une valeur de nombre entier et un nom de méthode. Il identifie, ordonne et séquence les requêtes SIP au sein d'un dialogue. Il permet de différencier les nouveaux messages et les retransmissions.
5. **Via** : Le champ Via indique le chemin pris par la requête et identifie où la réponse doit être envoyée. Il indique aussi le transport utilisé. Cet en-tête doit contenir un paramètre "branch" qui est utilisé pour identifier la transaction créée par la requête. Ce paramètre est utilisé aussi bien par le client que par le serveur. Il doit toujours commencer par un "magic cookie" d'une valeur de "z9hG4bK".
6. **Contact** : Cet en-tête identifie un URI SIP ou SIPS où l'UA veut une nouvelle requête SIP.

7. **Allow** : Cet en-tête liste les méthodes supportées par l'UA qui génère le message.
8. **Supported** : Cet en-tête liste toutes extensions supportées par l'UA autre que celles définies dans le [RFC 3261](#). Les extensions SIP sont représentées comme des étiquettes (tags) option.
9. **Require** : Identique au précédent "Supported" mais obligatoire pour aboutir la transaction.
10. **Content-Type** : Cet en-tête indique le type de corps de message (body)
11. **Content-Length** : Indique la longueur du corps du message en décimal. Il est obligatoire quand les messages SIP sont transportés sur TCP.

Les champs d'en-tête obligatoires dans toutes les demandes SIP sont : To, From, CSeq, Call-ID, Max-Forwards et Via.

Exemple d'une transaction BYE-200

Message	Structure
Méthodes	Nom de la méthode + URI du destinataire + Version SIP <code>BYE sip:201@10.33.6.101:5060 SIP/2.0</code>
Réponses	Version SIP + Status Code + Raison <code>SIP/2.0 200 OK</code>

Requête BYE

```
BYE sip:201@10.33.6.101:5060 SIP/2.0
Via: SIP/2.0/UDP 10.33.6.100;branch=z9hG4bKac795178845
Max-Forwards: 70
From: <sip:101@10.33.6.100;user=phone>;tag=1c782609321
To: <sip:201@10.33.6.101>;tag=1c1450530943
Call-ID: 1450530377152201062221@10.33.6.101
CSeq: 1 BYE
Supported: em,timer,replaces,path,resource-priority
Allow: REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE,UPDATE
User-Agent: IPP/v.6.20A.027.012
Reason: Q.850 ;cause=16 ;text="local"
Content-Length: 0
```

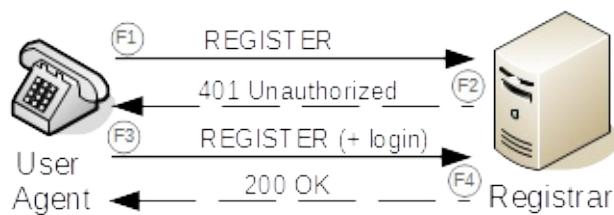
Réponse 200 OK

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 10.33.6.100;branch=z9hG4bKac795178845
From: <sip:101@10.33.6.100;user=phone>;tag=1c782609321
To: <sip:201@10.33.6.101>;tag=1c1450530943
Call-ID: 1450530377152201062221@10.33.6.101
CSeq: 1 BYE
Contact: <sip:201@10.33.6.101:5060>
Supported: em,timer,replaces,path,resource-priority
Allow: REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE,UPDATE
Server: GW/v.6.20A.027.012
Content-Length: 0
```

8. Scénarios SIP

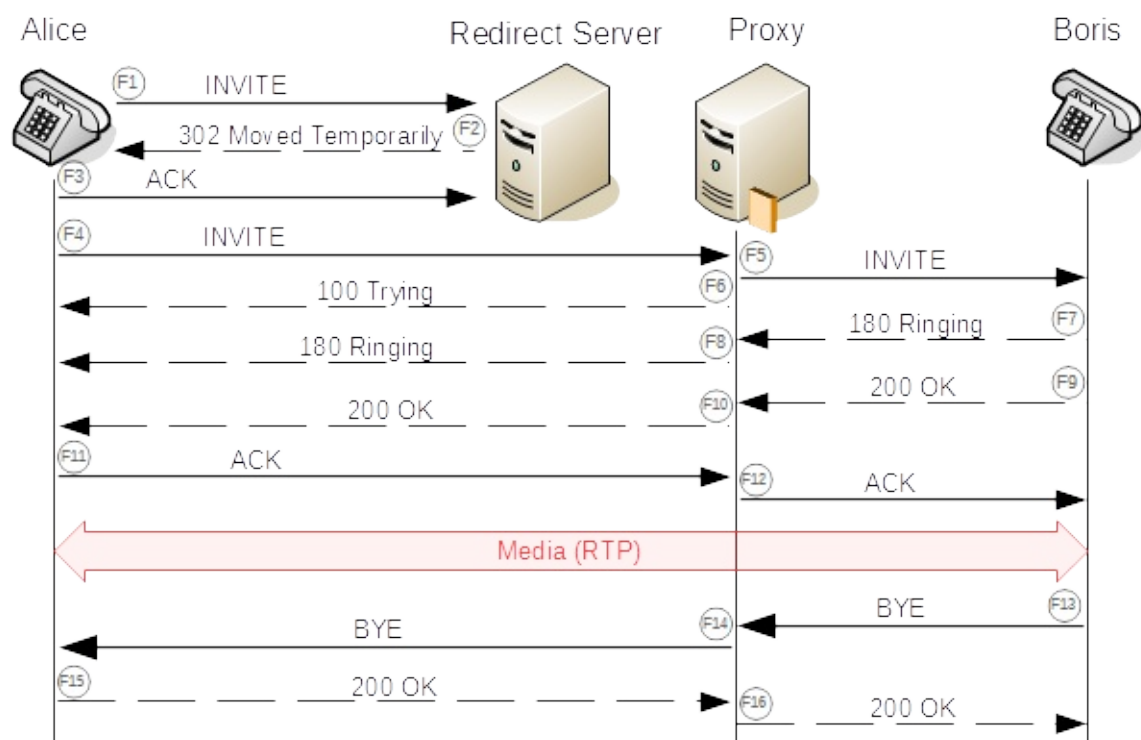
Selon le contexte d'exécution on trouvera probablement plusieurs intervenants dans une session SIP.

Processus d'enregistrement

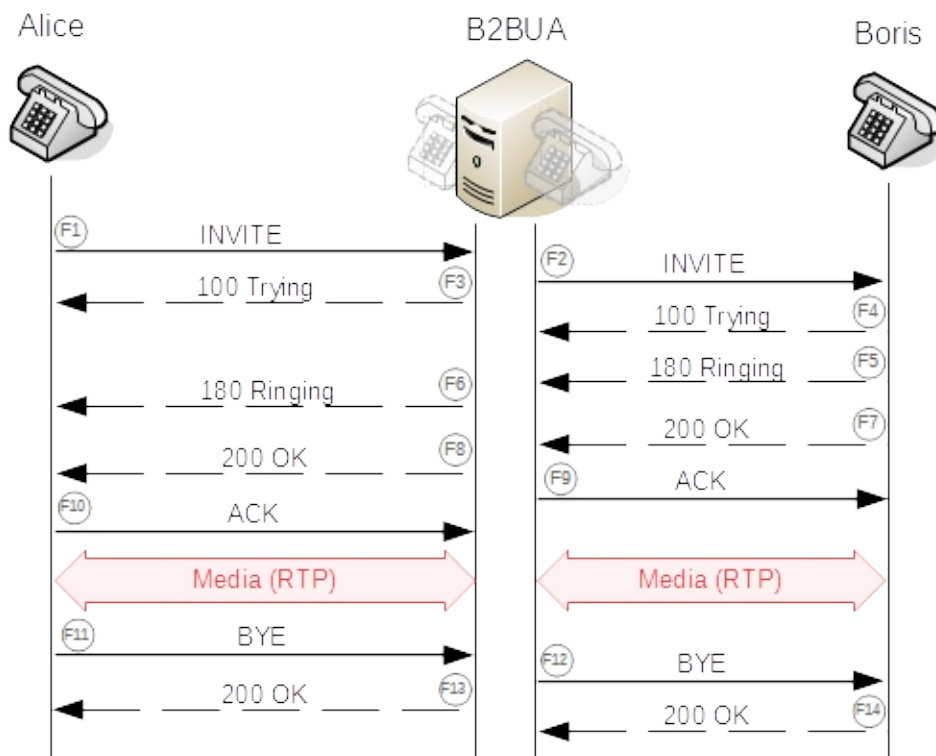


- F1 - initial registration request from SIP User Agent with it's address information (AOR)
- F2 - answer from SIP Registrar with the information about necessary login
- F3 - new registration request with login
- F4 - confirm of successful registration on SIP Registrar

Flux d'appel SIP entre UA et serveurs de redirection entre proxys et UAs



Flux d'appel B2BUA



9. UA en ligne de commande

- PJSUA
- linphonec
- linphonecsh

Installation PJSUA

```
wget http://www.pjsip.org/release/1.12/pjproject-1.12.tar.bz2
tar -xjf pjproject-1.12.tar.bz2
cd pjproject-1.12/
./configure
make dep && make && make install
cp pjsip-apps/bin/pjsua* /usr/local/bin/pjsua
```

Exemple d'usage de PJSUA

Exemple d'un UA qui s'enregistre sur `sip:3302@172.16.98.223`, qui répond automatiquement aux appels en boucle.

```
# cat autoansw.conf
--null-audio
--realm asterisk
--registrar sip:172.16.98.223
--id sip:3302@172.16.98.223
--username 3302
--password test1234
--play-file /root/believe-its-free.wav
--auto-answer 200
--auto-play --auto-loop
--max-calls 10
```

```
# pjsua --config-file autoansw.conf
```

Installation du dépôt Linphone

En Centos 7, ajouter ceci dans le fichier `/etc/yum.repos.d/Belledonne.repo`

```
[Belledonne]
name=Belledonne Communications
baseurl=https://www.linphone.org/snapshots/centos7
enabled=1
gpgcheck=0
priority=1
```

```
# yum update
# yum install linphone-devel
```

linphonecsh

Exemple linphonecsh

```
# linphonecsh init
# linphonecsh register --host 172.16.98.223 --username 3303 --password test1234
# linphonecsh dial "sip:3302@172.16.98.223"
```

linphonec

Exemple linphonec

```
# linphonec -c .linphonerc
```

```
Warning: video is disabled in linphonec, use -V or -C or -D to enable.
linphonec> register sip:3303@172.16.98.223 sip:172.16.98.223 test1234
linphonec> Registration on <sip:172.16.98.223> successful.
linphonec> call sip:3302@172.16.98.223
Establishing call id to <sip:3302@172.16.98.223>, assigned id 1
Contacting <sip:3302@172.16.98.223>
linphonec> Call 1 to <sip:3302@172.16.98.223> in progress.
linphonec> Remote ringing.
linphonec> Remote ringing...
linphonec> Call 1 to <sip:3302@172.16.98.223> ringing.
Remote ringing.
linphonec> Call 1 with <sip:3302@172.16.98.223> connected.
Call answered by <sip:3302@172.16.98.223>.
linphonec> Media streams established with <sip:3302@172.16.98.223> for call 1 (audio).
linphonec> terminate
Call ended
linphonec> Call 1 with <sip:3302@172.16.98.223> ended (No error).
linphonec> quit
Terminating...
Unregistration on <sip:172.16.98.223> done.
```


Aperçu des opérations SIP

Source : [RFC 3261](http://abcdrfc.free.fr/rfc-vf/rfc3261.html), "4 Overview of Operation", traduit par <http://abcdrfc.free.fr/rfc-vf/rfc3261.html> et retravaillé par les soins de l'auteur.

Le premier exemple montre les fonctions de base de SIP : localisation d'un point de terminaison, signal d'un désir de communiquer, négociation des paramètres de session pour établir la session, et suppression de la session une fois établie.

La figure montre un exemple typique d'échange de messages SIP entre deux utilisateurs, Alice et Bob. (Chaque message est étiqueté avec la lettre "F" et un numéro pour référence par le texte.) Dans cet exemple, Alice utilise une application SIP sur son micro ordinateur (auquel on se réfère comme téléphone logiciel) pour appeler Bob sur son téléphone SIP sur l'Internet. Deux serveurs mandataires SIP sont également montrés, qui agissent au nom d'Alice et de Bob pour faciliter l'établissement de la session. Cet arrangement typique est souvent désigné sous le nom "trapézoïde de SIP" comme le montre la forme géométrique de lignes pointillées dans la figure suivante.

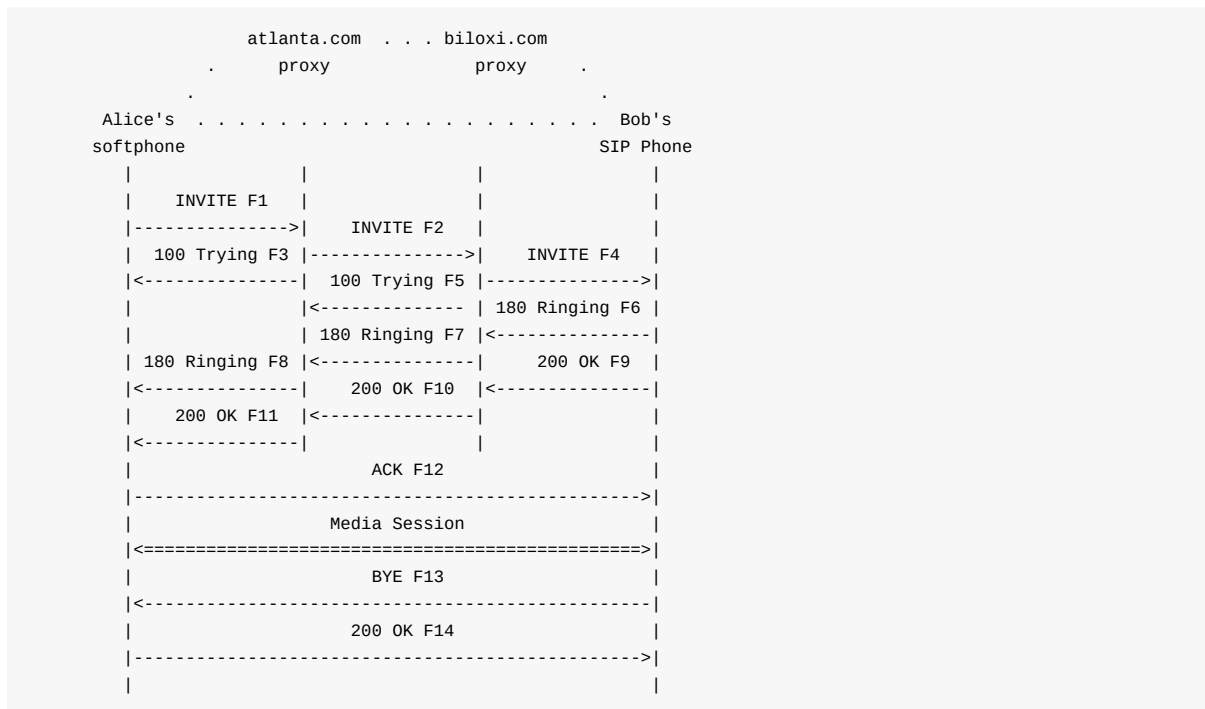


Figure : Exemple d'établissement de session SIP avec le trapézoïde SIP

Alice "appelle" Bob en utilisant son identité SIP, un type d'identifiant de ressource universel (URI, Uniform Resource Identifier) appelé un URI SIP. Ils ont une forme similaire à celle d'une adresse de messagerie électronique, contenant normalement un nom d'utilisateur et un nom d'hôte. Dans ce cas, c'est `sip:bob@biloxi.com`, où `biloxi.com` est le domaine du fournisseur de service SIP de Bob. Alice a un URI SIP qui est `sip:alice@atlanta.com`. Alice pourrait avoir tapé l'URI de Bob ou peut être cliqué sur un lien hypertexte ou sur une entrée dans un carnet d'adresses. SIP fournit aussi des URI sécurisés, appelés URI SIPS. Ce serait par exemple `sips:bob@biloxi.com`. Un appel fait à un URI SIPS garantit qu'un transport sécurisé, chiffré (à savoir TLS) est utilisé pour porter tous les messages SIP de l'appelant au domaine de l'appelé. A partir de là, la demande est envoyée en toute sécurité à l'appelé, mais avec des mécanismes de sécurité qui dépendent de la politique du domaine de l'appelé.

SIP se fonde sur un modèle de transaction de demande/réponse du style HTTP. Chaque transaction consiste en une demande qui implique une méthode ou fonction particulière sur le serveur, et au moins une réponse. Dans cet exemple, la transaction débute avec l'envoi par le téléphone logiciel d'Alice d'une demande INVITE adressée à l'URI

SIP de Bob. INVITE est un exemple de méthode SIP qui spécifie l'action que le demandeur (Alice) veut que fasse le serveur (Bob). La demande INVITE contient un certain nombre de champs d'en-tête. Les champs d'en-tête sont des attributs désignés qui fournissent des informations supplémentaires sur un message. Ceux qui sont présents dans un INVITE incluent un identifiant unique pour l'appel, l'adresse de destination, l'adresse d'Alice, et des informations sur le type de session qu'Alice souhaite établir avec Bob. L'INVITE (message F1 dans la figure) pourrait ressembler à :

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
...
```

(Le SDP d'Alice n'est pas montré)

La première ligne du message en texte codé contient le nom de la méthode (INVITE). Les lignes qui suivent sont une liste de champs d'en-tête. Cet exemple contient l'ensemble minimum requis. Les champs d'en-tête sont brièvement décrits ci-dessous :

- **Via** contient l'adresse (`pc33.atlanta.com`) à laquelle Alice s'attend à recevoir des réponses à cette demande. Il contient aussi un paramètre de branche qui identifie cette transaction.
- **To** (à) contient un nom d'affichage (Bob) et un URI SIP ou SIPS (`sip:bob@biloxi.com`) sur lequel la demande a été dirigée à l'origine. Les noms d'affichage sont décrits dans la RFC 2822 rendu obsolète par le [RFC 5322](#).
- **From** (de) contient aussi un nom d'affichage (Alice) et un URI SIP ou SIPS (`sip:alice@atlanta.com`) qui indique la personne à l'origine de la demande. Ce champs d'en-tête a aussi un paramètre d'étiquette contenant une chaîne aléatoire (`1928301774`) qui a été ajoutée à l'URI par le téléphone logiciel. Elle sert pour des besoins d'identification.
- **Call-ID** (identifiant d'appel) contient un identifiant unique au monde pour cet appel, généré par la combinaison d'une chaîne aléatoire et du nom d'hôte ou de l'adresse IP du téléphone logiciel. La combinaison de **l'étiquette To**, de **l'étiquette From**, et du **Call-ID** (identifiant d'appel) définit de façon complète une relation SIP d'homologue à homologue entre Alice et Bob qu'on appelle un **dialogue**.
- **CSeq** ou Command Sequence (séquence de commande) contient un entier et un nom de méthode. Le nombre CSeq est incrémenté pour chaque nouvelle demande d'un dialogue et c'est un numéro de séquence traditionnel.
- **Contact** contient un URI SIP ou SIPS qui représente une route directe pour contacter Alice, habituellement composé d'un nom d'utilisateur sur un nom de domaine pleinement qualifié (FQDN, fully qualified domain name). Alors qu'un FQDN est préféré, de nombreux systèmes d'extrémité n'ont pas de noms de domaine enregistrés, de sorte que les adresses IP sont permises. Alors que le champ d'en-tête Via indique les autres éléments sur le lieu où envoyer la réponse, le champ d'en-tête Contact indique d'autres éléments sur où envoyer des demandes futures.
- **Max-Forwards** (retransmissions maximales) sert à limiter le nombre de sauts que peut faire une demande sur son chemin vers sa destination. Il consiste en un entier qui est désincrémenté d'un à chaque saut.
- **Content-Type** (type de contenu) contient une description du corps de message (non montré).
- **Content-Length** (longueur du contenu) contient un compte d'octets du corps de message.

Les détails de la session, comme le type de support, codec ou débit d'échantillonnage ne sont pas décrits à l'aide de SIP. Le corps d'un message SIP contient plutôt une description de la session, codée dans un autre format de protocole. Un de ces formats est le protocole de description de session (SDP, Session Description Protocol) ([RFC 4566](#)). Ce message SDP (non montré dans l'exemple) est porté par le message SIP d'une façon analogue à celle dont un document joint est porté par un message électronique, ou dont une page web est portée par un message HTTP.

Comme le téléphone logiciel ne connaît pas la localisation de Bob ou le serveur SIP dans le domaine biloxi.com, le téléphone logiciel envoie l'INVITE au serveur SIP qui dessert le domaine d'Alice, atlanta.com. L'adresse du serveur SIP atlanta.com pourrait avoir été configurée dans le téléphone logiciel d'Alice, ou elle aurait pu être découverte par DHCP par exemple.

Le serveur SIP atlanta.com est un type de serveur SIP connu sous le nom de serveur mandataire (proxy). Un serveur mandataire reçoit des demandes SIP et les transmet au nom du demandeur. Dans cet exemple, le serveur mandataire reçoit la demande INVITE et renvoie une réponse 100 (En essai) au téléphone logiciel d'Alice. La réponse 100 (En essai) indique que l'INVITE a été reçu et que le mandataire travaille en son nom pour acheminer l'INVITE à la destination. Les réponses dans SIP utilisent un code à trois chiffres suivi par une phrase descriptive. **Cette réponse contient les mêmes paramètres To, From, Call-ID, CSeq et branch dans le Via que dans l'INVITE, ce qui permet au téléphone logiciel d'Alice de corréler cette réponse avec l'INVITE envoyé.** Le serveur mandataire atlanta.com localise le serveur mandataire à biloxi.com, éventuellement en effectuant un type particulier de recherche de DNS (service de nom de domaine) pour trouver le serveur SIP qui dessert le domaine biloxi.com. En résultat, il obtient l'adresse IP du serveur mandataire biloxi.com et y transmet ou mandate la demande INVITE. Avant de transmettre la demande, le serveur mandataire atlanta.com **ajoute une valeur de champ d'en-tête Via supplémentaire** qui contient sa propre adresse (l'INVITE contient déjà l'adresse d'Alice dans le premier Via). Le serveur mandataire biloxi.com reçoit l'INVITE et répond par une réponse 100 (En essai) au serveur mandataire atlanta.com pour indiquer qu'il a reçu l'INVITE et traite la demande. Le serveur mandataire consulte une base de données, appelée de façon générique un service de localisation, qui contient l'adresse IP courante de Bob. (Nous verrons dans la prochaine section comment on peut remplir cette base de données.) Le serveur mandataire biloxi.com ajoute une autre valeur de champ d'en-tête Via avec sa propre adresse pour l'INVITE et l'envoie par procuration au téléphone SIP de Bob.

Le téléphone SIP de Bob reçoit l'INVITE et alerte Bob de l'appel entrant provenant d'Alice, de sorte que Bob peut décider s'il va répondre à l'appel, c'est à dire, le téléphone de Bob sonne. Le téléphone SIP de Bob indique cela dans une réponse 180 (Sonnerie), qui est acheminée en retour aux deux mandataires dans les directions opposées. Chaque mandataire utilise le champ d'en-tête Via pour déterminer où il faut envoyer la réponse et retire sa propre adresse du haut. En résultat, bien que les examens DNS et de service de localisation soient nécessaires pour acheminer l'INVITE initiale, la réponse 180 (Sonnerie) peut être retournée à l'appelant sans boucles ou sans que des états soient maintenus dans les mandataires. Cela a aussi la propriété avantageuse que chaque mandataire qui voit l'INVITE va aussi voir toutes les réponses à l'INVITE.

Lorsque le téléphone logiciel d'Alice reçoit la réponse 180 (Sonnerie), il passe cette information à Alice, peut-être en utilisant une tonalité de retour d'appel audio ou en affichant un message sur l'écran d'Alice.

Dans cet exemple, Bob décide de répondre à l'appel. Lorsqu'il décroche le combiné, son téléphone SIP envoie une réponse 200 (OK) pour indiquer que l'appel a reçu une réponse. Le 200 (OK) contient un corps de message avec la description de support SIP du type de session que Bob veut établir avec Alice. En résultat, il y a un échange de messages SDP en deux phases : Alice envoie un à Bob, et Bob en renvoie un à Alice. Cet échange en deux phases donne une capacité de négociation de base et il est fondé sur un modèle simple d'offre/réponse d'échange SDP. Si Bob ne souhaite pas répondre à l'appel ou qu'il est occupé sur une autre ligne, une réponse d'erreur aurait été envoyée à la place du 200 (OK), et il en aurait résulté qu'aucune session de support n'aurait été établie. La liste complète des codes de réponse SIP figure à la page [Réponses SIP](#). Le 200 (OK) (message F9 dans la Figure 1) pourrait ressembler à cela car Bob a envoyé :

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bKnashds8;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhd8 ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
```

```
Content-Type: application/sdp
Content-Length: 131
...
```

(le SDP de Bob n'est pas montré)

La première ligne de la réponse contient le code de réponse (200) et la phrase de cause (OK). Les lignes restantes contiennent des champs d'en-tête. **Les champs d'en-tête Via, To, From, Call-ID, et CSeq sont copiés de la demande INVITE.** (Il y a trois valeurs de champ d'en-tête Via - une ajoutée par le téléphone SIP d'Alice, une ajoutée par le mandataire atlanta.com, et une ajoutée par le mandataire biloxi.com.) Le téléphone SIP de Bob a ajouté un paramètre d'étiquette au champs d'en-tête. Cette étiquette sera incorporée dans le dialogue par les deux points d'extrémité et sera incluse dans toutes les futures demandes et réponses de cet appel. Le champ d'en-tête Contact contient un URI auquel Bob peut être atteint directement sur son téléphone SIP. Le Content-Type (type de contenu) et Content-Length (longueur du contenu) se réfèrent au corps de message (non montré) qui contient les informations de support SDP de Bob.

En plus de l'examen de DNS et de service de localisation montré dans cet exemple, les serveurs mandataires peuvent prendre des "décisions d'acheminement" souples pour décider d'envoyer une demande. Par exemple, si le téléphone SIP de Bob a retourné une réponse 486 (Occupé ici), le serveur mandataire biloxi.com pourrait mandater l'INVITE au serveur de messagerie vocale de Bob. Un serveur mandataire peut aussi envoyer un INVITE à un certain nombre de localisations en même temps. Ce type de recherche parallèle est connu sous le nom de fourchage (forking).

Dans ce cas, le 200 (OK) est ré-acheminé à travers les deux mandataires et est reçu par le téléphone logiciel d'Alice, qui arrête alors la tonalité de retour d'appel et indique que l'appel a reçu une réponse. Finalement, le téléphone logiciel d'Alice envoie un message d'accusé de réception, ACK, au téléphone SIP de Bob pour confirmer la réception de la réponse finale (200 (OK)). Dans cet exemple, le ACK est envoyé directement du téléphone logiciel d'Alice au téléphone SIP de Bob, outrepassant les deux mandataires. Ceci survient parce que les points d'extrémité ont appris leurs adresses respectives d'après les champs d'en-tête Contact à travers l'échange INVITE/200 (OK), qui n'étaient pas connues lorsque l'INVITE initial a été envoyé. Les examens effectués par les deux mandataires ne sont plus nécessaires, de sorte que les mandataires abandonnent le flux d'appel. Cela termine la prise de contact à trois INVITE/200/ACK utilisée pour établir les sessions SIP.

La session media d'Alice et Bob a maintenant commencé, et ils envoient des paquets vocaux/vidéo en utilisant le format sur lequel ils se sont mis d'accord dans l'échange de SDP. En général, ces paquets de bout en bout prennent un chemin différent de celui des messages de signalisation SIP.

Durant la session, Alice ou Bob peut décider de changer les caractéristiques de la session média. Ceci est accompli en envoyant un re-INVITE contenant une nouvelle description de support. Ce re-INVITE fait référence au dialogue existant de sorte que l'autre partie sait que c'est pour modifier une session existante et non pour établir une nouvelle session. L'autre partie envoie un 200 (OK) pour accepter le changement. Le demandeur répond au 200 (OK) par un ACK. Si l'autre partie n'accepte pas le changement, elle envoie une réponse d'erreur comme 488 (Non acceptable ici), qui reçoit aussi un ACK. Cependant, l'échec du re-INVITE ne cause pas de défaillance de l'appel existant - la session continue en utilisant les caractéristiques précédemment négociées.

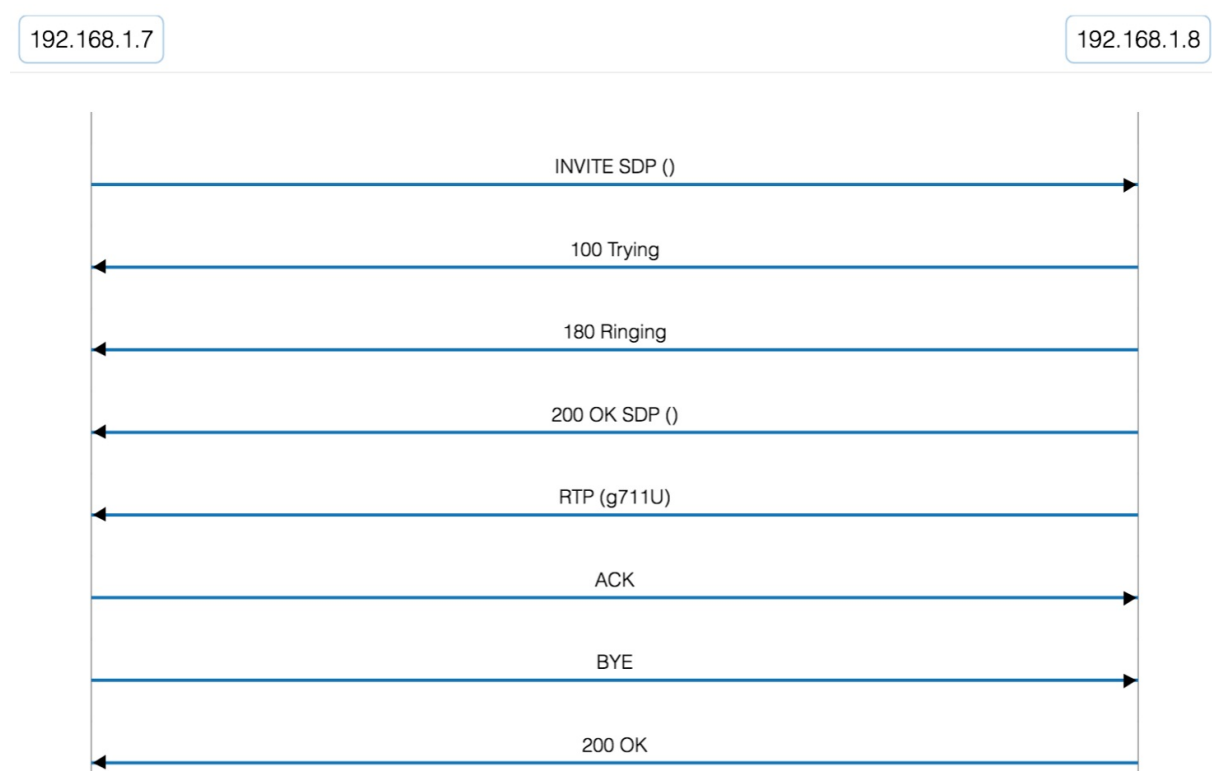
A la fin de l'appel, Bob déconnecte (raccroche) d'abord et génère un message BYE. Ce BYE est acheminé directement au téléphone logiciel d'Alice, outrepassant à nouveau les mandataires. Alice confirme la réception du BYE par une réponse 200 (OK), qui termine la session et la transaction BYE. Aucun ACK n'est envoyé - un ACK n'est envoyé qu'en réponse à une réponse à une demande INVITE. Les raisons de ce traitement spécial pour INVITE seront exposées plus loin mais se rapportent à des mécanismes de fiabilité dans SIP, à la durée qu'il faut pour répondre à un téléphone qui sonne et au fourchage. Pour cette raison, le traitement de la demande dans SIP est souvent classé comme INVITE ou non-INVITE, se référant à toutes les autres à côté de INVITE.

INVITE SIP UAC/UAS

- 1. Requête INVITE SDP
- 2. Réponse provisionnelle 100 Trying
- 3. Réponse provisionnelle 180 Ringing
- 4. Réponse 200 OK SDP
- 5. ACK
- 6. Transfert des medias
- 7. Requête BYE
- 8. Réponse OK

On peut éprouver les capacités de SIP dans un contexte UAC vers UAS sans serveur proxy. Le plus simple est d'installer un téléphone logiciel comme Linphone ou CSIPsimple ou encore PJSUA sur deux ordinateurs du réseau local.

Ici la destination est encodée directement dans le logiciel : `sip:192.168.1.8`. Aucun paramètre n'est configuré sur le logiciel.



Source : <https://www.cloudshark.org/captures/f22c8443205b>

Après avoir pu recevoir une ou plusieurs réponses provisoires, l'UAC va obtenir une ou plusieurs réponses 2xx ou une réponse finale non-2xx. A cause de la durée prolongée que peut prendre la réception des réponses finales à INVITE, les mécanismes de fiabilité des transactions INVITE diffèrent de celles des autres demandes (comme OPTIONS). Une fois qu'il reçoit une réponse finale, l'UAC doit envoyer un ACK pour chaque réponse finale qu'il reçoit. La procédure d'envoi de cet ACK dépend du type de réponse. Pour les réponses finales entre 300 et 699, le traitement de l'ACK est effectué dans la couche transaction et suit un ensemble de règles (voir la Section 17). Pour les réponses 2xx, le ACK est généré par l'UAC central.

Une réponse 2xx à un INVITE établit une session, et elle crée aussi un dialogue entre l'agent d'utilisateur qui a produit l'INVITE et l'agent d'utilisateur qui a généré la réponse 2xx. Donc, lorsque plusieurs réponses 2xx sont reçues en provenance de différents agents d'utilisateur distants (à cause d'un INVITE "fourchu"), chaque 2xx établit un dialogue différent. Tous ces dialogues font partie du même appel.

Source : [RFC 3261 13.1. Initiating a Session Overview](#)

1. Requête INVITE SDP

Le premier message est une requête INVITE décrite comme suit

```
Session Initiation Protocol (INVITE)
Request-Line: INVITE sip:192.168.1.8 SIP/2.0
Message Header
  Via: SIP/2.0/UDP 192.168.1.7:5060;branch=z9hG4bK.zQpMKMKmd;rport
  From: <sip:linphone@192.168.1.7>;tag=zDTUMmqVS
  To: sip:192.168.1.8
  CSeq: 20 INVITE
  Call-ID: dJpMERl3hb
  Max-Forwards: 70
  Supported: outbound
  Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO, UPDATE
  Content-Type: application/sdp
  Content-Length: 585
  Contact: <sip:linphone@192.168.1.7>;+sip.instance="urn:uuid:aa9274b7-03d6-4a98-b5f3-474e7603a6d0>"
  User-Agent: Linphone/3.9.1 (belle-sip/1.4.2)
Message Body
```

2. Réponse provisionnelle 100 Trying

```
Status-Line: SIP/2.0 100 Trying
Message Header
  Via: SIP/2.0/UDP 192.168.1.7:5060;rport=5060;received=192.168.1.7;branch=z9hG4bK.zQpMKMKmd
  Call-ID: dJpMERl3hb
  From: <sip:linphone@192.168.1.7>;tag=zDTUMmqVS
  To: <sip:192.168.1.8>
  CSeq: 20 INVITE
  Content-Length: 0
```

3. Réponse provisionnelle 180 Ringing

```
Status-Line: SIP/2.0 180 Ringing
Message Header
  Via: SIP/2.0/UDP 192.168.1.7:5060;rport=5060;received=192.168.1.7;branch=z9hG4bK.zQpMKMKmd
  Call-ID: dJpMERl3hb
  From: <sip:linphone@192.168.1.7>;tag=zDTUMmqVS
  To: <sip:192.168.1.8>;tag=2ALrvgnIqfZqo1h4Cs5PAG9HS3b4GQNz
  CSeq: 20 INVITE
  Contact: <sip:192.168.1.8:5060>
  Allow: PRACK, INVITE, ACK, BYE, CANCEL, UPDATE, INFO, SUBSCRIBE, NOTIFY, REFER, MESSAGE, OPTIONS
  Content-Length: 0
```

4. Réponse 200 OK SDP

```
Status-Line: SIP/2.0 200 OK
Message Header
  Via: SIP/2.0/UDP 192.168.1.7:5060;rport=5060;received=192.168.1.7;branch=z9hG4bK.zQpMKMKmd
  Call-ID: dJpMERl3hb
  From: <sip:linphone@192.168.1.7>;tag=zDTUMmqVS
  To: <sip:192.168.1.8>;tag=2ALrvgnIqfZq01h4Cs5PAG9HS3b4GQNz
  CSeq: 20 INVITE
  Allow: PRACK, INVITE, ACK, BYE, CANCEL, UPDATE, INFO, SUBSCRIBE, NOTIFY, REFER, MESSAGE, OPTIONS
  Contact: <sip:192.168.1.8:5060>
  Supported: replaces, 100rel, timer, norefersub
  Content-Type: application/sdp
  Content-Length: 300
Message Body
  Session Description Protocol
```

5. ACK

```
Request-Line: ACK sip:192.168.1.8:5060 SIP/2.0
Message Header
  Via: SIP/2.0/UDP 192.168.1.7:5060;rport;branch=z9hG4bK.fPDRQHIBm
  From: <sip:linphone@192.168.1.7>;tag=zDTUMmqVS
  To: <sip:192.168.1.8>;tag=2ALrvgnIqfZq01h4Cs5PAG9HS3b4GQNz
  CSeq: 20 ACK
  Call-ID: dJpMERl3hb
  Max-Forwards: 70
```

6. Transfert des medias

Le transfert de la voix est assuré par RTP codé en G.711 PCMU. Le format a été négocié par SDP dans la requête INVITE et la réponse 200 OK.

```
Request-Line: INVITE sip:192.168.1.8 SIP/2.0
Message Header
Message Body
  Session Description Protocol
    Session Description Protocol Version (v): 0
    Owner/Creator, Session Id (o): linphone 1108 2378 IN IP4 192.168.1.7
    Session Name (s): Talk
    Connection Information (c): IN IP4 192.168.1.7
    Time Description, active time (t): 0 0
    Session Attribute (a): rtcp-xr:rcvr-rtt=all:10000 stat-summary=loss,dup,jitt,TTL voip-metrics
    Media Description, name and address (m): audio 7078 RTP/AVP 96 97 98 99 0 8 101 100 102
      Media Type: audio
      Media Port: 7078
      Media Protocol: RTP/AVP
      Media Format: DynamicRTP-Type-96
      Media Format: DynamicRTP-Type-97
      Media Format: DynamicRTP-Type-98
      Media Format: DynamicRTP-Type-99
      Media Format: ITU-T G.711 PCMU
      Media Format: ITU-T G.711 PCMA
      Media Format: DynamicRTP-Type-101
      Media Format: DynamicRTP-Type-100
      Media Format: DynamicRTP-Type-102
    Media Attribute (a): rtpmap:96 opus/48000/2
    Media Attribute (a): fmp:96 useinbandfec=1
    Media Attribute (a): rtpmap:97 SILK/16000
    Media Attribute (a): rtpmap:98 speex/16000
    Media Attribute (a): fmp:98 vbr=on
    Media Attribute (a): rtpmap:99 speex/8000
    Media Attribute (a): fmp:99 vbr=on
```

```
Media Attribute (a): rtpmap:101 telephone-event/48000
Media Attribute (a): rtpmap:100 telephone-event/16000
Media Attribute (a): rtpmap:102 telephone-event/8000
Media Description, name and address (m): video 9078 RTP/AVP 96 97
Media Attribute (a): rtpmap:96 VP8/90000
Media Attribute (a): rtpmap:97 H264/90000
Media Attribute (a): fmp:97 profile-level-id=42801F
```

```
Status-Line: SIP/2.0 200 OK
Message Header
Message Body
  Session Description Protocol
    Session Description Protocol Version (v): 0
    Owner/Creator, Session Id (o): - 3671180764 3671180766 IN IP4 192.168.1.8
    Session Name (s): pjmedia
    Connection Information (c): IN IP4 192.168.1.8
    Time Description, active time (t): 0 0
    Media Description, name and address (m): audio 4006 RTP/AVP 0 102
    Connection Information (c): IN IP4 192.168.1.8
    Media Attribute (a): rtcp:4007 IN IP4 192.168.1.8
    Media Attribute (a): sendrecv
    Media Attribute (a): rtpmap:0 PCMU/8000
    Media Attribute (a): rtpmap:102 telephone-event/8000
    Media Attribute (a): fmp:102 0-16
    Media Description, name and address (m): video 0 RTP/AVP
    Connection Information (c): IN IP4 192.168.1.8
```

7. Requête BYE

```
Request-Line: BYE sip:192.168.1.8:5060 SIP/2.0
Message Header
  Via: SIP/2.0/UDP 192.168.1.7:5060;branch=z9hG4bK.UKDJAE8~q;rport
  From: <sip:linphone@192.168.1.7>;tag=zDTUMmqVS
  To: <sip:192.168.1.8>;tag=2ALrvgnIqfZq01h4Cs5PAG9HS3b4GQNz
  CSeq: 21 BYE
  Call-ID: dJpMERl3hb
  Max-Forwards: 70
  User-Agent: Linphone/3.9.1 (belle-sip/1.4.2)
```

8. Réponse OK

```
Status-Line: SIP/2.0 200 OK
Message Header
  Via: SIP/2.0/UDP 192.168.1.7:5060;rport=5060;received=192.168.1.7;branch=z9hG4bK.UKDJAE8~q
  Call-ID: dJpMERl3hb
  From: <sip:linphone@192.168.1.7>;tag=zDTUMmqVS
  To: <sip:192.168.1.8>;tag=2ALrvgnIqfZq01h4Cs5PAG9HS3b4GQNz
  CSeq: 21 BYE
  Content-Length: 0
```

Réponses SIP

- Informational (1xx)
 - 100 Trying
 - 180 Ringing
 - 181 Call is Being Forwarded
 - 182 Call Queued
 - 183 Session Progress
- Success (2xx)
 - 200 OK
 - 202 Accepted
- Redirection (3xx)
 - 300 Multiple Choices
 - 301 Moved Permanently
 - 302 Moved Temporarily
 - 305 Use Proxy
 - 380 Alternative Service
- Client Error (4xx)
 - 400 Bad Request
 - 401 Unauthorized
 - 403 Forbidden
 - 404 Not Found
 - 405 Method Not Allowed
 - 406 Not Acceptable
 - 407 Proxy Authentication Required
 - 408 Request Timeout
 - 422 Session Timer Interval Too Small
 - 423 Interval Too Brief
 - 480 Temporarily Unavailable
 - 481 Dialog/Transaction Does Not Exist
 - 483 Too Many Hops
 - 486 Busy Here
 - 487 Request Terminated
- Server Failure (5xx)
 - 500 Server Internal Error
 - 501 Not Implemented
 - 502 Bad Gateway
 - 503 Service Unavailable
 - 504 Gateway Timeout
 - 505 Version Not Supported
 - 513 Message Too Large
 - 580 Preconditions Failure
- Global Error (6xx)
 - 600 Busy Everywhere
 - 603 Decline
 - 604 Does Not Exist Anywhere
 - 606 Not Acceptable

Il existe six classes de réponses SIP identifiées par la valeur du premier numéro du code de réponse.

- 1xx Provisional : Requête reçue et traitement de la requête.
- 2xx Success : L'action a été reçue, comprise et acceptée avec succès.
- 3xx Redirection : Une action plus éloignée est nécessaire pour achever la requête.
- 4xx Client Error : La requête contient une mauvaise syntaxe ou ne peut pas être prise en charge par le serveur.
- 5xx Server Error : Le serveur a échoué à prendre en charge une requête valide.
- 6xx Global Failure : La requête ne peut être prise en charge par aucun serveur.

On accordera une attention particulière aux erreurs "4xx Client Error" :

- 400 Bad Request
- 401 Unauthorized
- 403 Forbidden
- 404 Not Found
- 405 Method Not Allowed
- 406 Not Acceptable
- 407 Proxy Authentication Required
- 408 Request Timeout
- 422 Session Timer Interval Too Small
- 423 Interval Too Brief
- 480 Temporarily Unavailable
- 481 Dialog/Transaction Does Not Exist
- 483 Too Many Hops
- 486 Busy Here
- 487 Request Terminated

Informational (1xx)

Les réponses Informational (1xx) sont utilisées pour indiquer la progression des appels. Normalement, les réactions sont bout à bout (à l'exception de 100 Trying). L'objectif principal des réponses d'information est d'arrêter la retransmission de INVITER demandes.

Les réponses d'information comprennent les réponses suivantes.

100 Trying

Cette réponse est seulement une demande hop-by-hop.

Elle n'est jamais transférée et peut contenir un corps de message.

Il est utilisé pour éviter la retransmission de messages INVITE.

180 Ringing

Cette réponse est utilisée pour indiquer qu'une invitation a été reçue par l'agent utilisateur et que l'alerte est en cours.

181 Call is Being Forwarded

Cette réponse est utilisée pour indiquer que l'appel a été transmis à un autre point de terminaison.

Elle est envoyée lorsque l'information peut être utile à l'appelant.

Elle donne le statut de l'appelant, comme par exemple une opération de transfert allonge le temps de réponse de l'appel.

182 Call Queued

Cette réponse est utilisée pour indiquer que le message INVITE a été reçu et est traitée dans une file d'attente.

183 Session Progress

La réponse "183 Session Progress" indique que des informations sur l'état d'avancement d'une session peut être présent dans un corps de message ou flux média.

Contrairement à une réponse "100 Trying", une réponse "183 Session Progress" est une réponse de bout en bout et établit un dialogue.

Une utilisation typique de cette réponse est de permettre un UAC d'entendre une sonnerie, une tonalité d'occupation, ou l'annonce enregistrée dans les appels via une passerelle vers le réseau PSTN.

Success (2xx)

Cette classe de réponses sert à indiquer qu'une demande a été acceptée. Elle comprend les réponses suivantes.

200 OK

200 OK signifie qu'une demande a été acceptée. Il indique une réussite de la demande.

202 Accepted

202 Accepted indique qu'un UAS a reçu et compris la demande, mais la demande ne peut pas avoir été autorisée ou traitée par le serveur.

Ce code est couramment utilisé dans les réponses aux méthodes SUBSCRIBE et REFER.

Redirection (3xx)

En général, cette classe de réponses est envoyée par les serveurs de redirection en réponse à un INVITE. Ils sont aussi appelés classe de réponses redirigées. Elle comprend les réponses suivantes.

300 Multiple Choices

Ce message contient plusieurs champs d'en-tête de contact pour indiquer que le service de localisation a rendu plusieurs emplacements possibles pour l'URI SIP dans la requête URI.

301 Moved Permanently

Cette réponse de redirection contient un champ d'en-tête `Contact` avec la nouvelle URI permanente de la partie appelée.

L'adresse peut être enregistrée et utilisée dans les futures demandes INVITE.

302 Moved Temporarily

Cette réponse de redirection contient un URI qui est en cours de validité, mais elle n'est pas permanente. L'emplacement est valable pour la durée du temps spécifié.

305 Use Proxy

Cette réponse contient un URI qui pointe vers un serveur proxy ayant des informations faisant autorité sur la partie appelante.

Cette réponse pourrait être envoyée par un UAS utilisant un proxy pour le filtrage d'appel entrant.

380 Alternative Service

Cette réponse renvoie un URI qui indique le type de service que la partie appelée souhaite.

Par exemple, un appel pourrait être redirigé vers un serveur de messagerie vocale.

Client Error (4xx)

Les réponses d'erreur client indiquent que la demande ne peut être satisfaite et que des erreurs sont identifiées du côté UAC. Les codes de réponse sont généralement envoyés par la UAS. Lors de la réception d'un message d'erreur, le client doit renvoyer la demande en la modifiant en fonction de la réponse. Voici quelques-unes des réponses importantes 4XX.

400 Bad Request

Ce code de réponse indique que la demande n'a pas été comprise par le serveur.

La demande (Request) peut manquer de champs d'en-tête nécessaires tels que `To` , `From` , `Call-ID` OU `CSeq` .

401 Unauthorized

Ce message indique que la demande exige que l'utilisateur réalise une authentification.

401 Unauthorized est normalement envoyée par un serveur d'enregistrement de la demande REGISTER.

La réponse contient champ d'en-tête `WWW-Authenticate` qui demande des informations d'identification correcte de l'agent utilisateur (UA) appelant.

403 Forbidden

Le code 403 est envoyé lorsque le serveur a compris la demande, a trouvé que la demande soit formulée correctement, mais qu'il ne pourra pas traiter la demande.

Cette réponse n'est pas utilisé lorsque l'autorisation est nécessaire.

404 Not Found

404 Not Found indique que l'utilisateur identifié par l'URI SIP dans la requête ne peut pas être localisé par le serveur.

405 Method Not Allowed

405 Method Not Allowed indique que le serveur ou l'agent utilisateur (UA) a reçu et compris une demande mais n'est pas prêt à y répondre.

Exemple: Une demande REGISTER peut être envoyée à un agent utilisateur.

Un champ `Allow` doit être présent pour informer l'UAC quelles méthodes sont acceptables.

406 Not Acceptable

Cette réponse indique que la demande ne peut pas être traitée en raison d'une exigence dans le message de demande.

Le champ d'en-tête `Accept` dans la demande ne contenait aucune des options prises en charge par le UAS.

407 Proxy Authentication Required

Cette demande est envoyée par un mandataire (proxy) qui indique que l'UAC doit d'abord s'authentifier auprès du mandataire avant que la demande puisse être traitée.

La réponse devrait contenir des informations sur le type d'informations d'identification requises par le proxy dans un champ d'en-tête `Proxy-Authenticate`.

La demande peut être présentée à nouveau avec les informations d'identification appropriées dans un champ d'en-tête `Proxy-Authorization`.

408 Request Timeout

Cette réponse est envoyée lorsqu'un champ d'en-tête `Expires` est présent en une requête INVITE et que la période de temps spécifiée est dépassée.

Ce message peut être envoyé par un proxy ou un fork agent utilisateur.

La demande peut être relancée à tout moment par l'UAC.

422 Session Timer Interval Too Small

La réponse est utilisée pour rejeter une demande contenant un champ d'en-tête `Session-Expires`.

L'intervalle minimum autorisé est indiqué dans le champ d'en-tête requis `Min-SE`.

L'initiateur de l'appel peut réessayer la demande sans le champ d'en-tête ou avec une valeur `Session-Expires` inférieure ou égale au minimum spécifié.

423 Interval Too Brief

La réponse est renvoyée par un serveur d'enregistrement (REGISTRAR) qui rejette une demande d'enregistrement parce que le délai d'expiration demandé sur un ou plusieurs Contacts est trop court.

La réponse doit contenir un champ d'en-tête `Min-Expires` liste l'intervalle d'expiration minimum que le REGISTRAR acceptera.

480 Temporarily Unavailable

Cette réponse indique que la demande a atteint la bonne destination, mais que la partie appelée n'est pas disponible pour une raison quelconque.

La réponse doit contenir un en-tête de `Retry-After` avoir indiqué lorsque la demande peut être prise en charge.

481 Dialog/Transaction Does Not Exist

Cette réponse indique qu'une réponse faisant référence à un appel existant ou d'une transaction a été reçue pour laquelle le serveur n'a pas de session (record) ou d'information d'état.

483 Too Many Hops

Cette réponse indique que la demande a été transférée un nombre maximal de fois fixé par l'en-tête `Max-Forwards` dans la demande.

Ceci est indiqué par la réception d'un `Max-Forward` fixé à la valeur 0 dans l'en-tête de la demande.

486 Busy Here

Cela indique l'agent utilisateur est occupé et qu'il ne peut pas accepter l'appel.

487 Request Terminated

Cette réponse peut être envoyée par un UA qui a reçu une demande CANCEL pour une demande INVITE en attente.

Une réponse 200 OK est envoyée pour confirmer (acknowledge) le CANCEL et une réponse 487 est envoyée pour annuler la transaction INVITE.

Server Failure (5xx)

Cette classe de réponse est utilisée pour indiquer que la demande ne peut pas être traitée en raison d'une erreur avec le serveur. Le serveur n'a pas réussi à satisfaire une demande apparemment valide. La réponse peut contenir un champ d'en-tête `Retry-After`. La demande peut être essayée à d'autres endroits, car il n'y a pas d'erreurs indiquées dans la demande. Certaines des réponses importantes de défaillance du serveur sont présentées ci-dessous.

500 Server Internal Error

500 indique que le serveur a connu une erreur qui empêche de traiter la demande.

C'est une sorte de défaillance typique du serveur qui indique au client de réessayer la demande à nouveau au bout de quelques secondes.

501 Not Implemented

Ce code indique que le serveur est incapable de traiter la demande car il est pas pris en charge.

Cette réponse peut être utilisée pour refuser une demande contenant une méthode inconnue.

502 Bad Gateway

Cette réponse est envoyée par un mandataire (proxy) qui agit comme une passerelle vers un autre réseau.

Il indique qu'un problème dans d'autres réseaux empêche la demande d'être traitée.

503 Service Unavailable

Cette réponse indique que le service est temporairement indisponible à ce moment-là.

La demande peut être relancée après quelques secondes, ou après l'expiration du champ d'en-tête `Retry-After`.

Une erreur fatale de transport rapportée par la couche transport (généralement due à des erreurs fatales d'ICMP dans UDP ou des défaillances de connexion dans TCP) dans la demande implique une réponse *503 Service Unavailable*.

504 Gateway Timeout

Cette réponse intervient quand la demande a échoué en raison d'un délai d'attente qui a eu lieu dans un autre réseau auquel la passerelle se connecte.

C'est une classe d'erreur de serveur, car l'appel échoue en raison d'une défaillance d'accès du serveur aux ressources en dehors du réseau SIP.

505 Version Not Supported

Le serveur refuse une demande quand il est livré avec un autre numéro de version SIP. Le refus est indiqué dans ce message.

À l'heure actuelle la version SIP 2.0 est la seule version implémentée.

513 Message Too Large

Cette réponse est utilisée par un UAS pour indiquer que la taille de la demande était trop grande être traitée.

580 Preconditions Failure

Cette réponse permet de rejeter une offre SDP qui exigerait des conditions préalables qui ne peuvent être satisfaites.

Global Error (6xx)

Cette classe de réponse indique que le serveur sait que la demande échouera où qu'elle soit tentée . En conséquence, la demande ne devrait pas être envoyé à d'autres endroits.

Seul un serveur ayant des connaissances définitives de l'utilisateur identifié par la requête devrait envoyer tous les cas possibles une réponse globale de la classe d'erreur. Dans le cas contraire, une réponse de classe d'erreur client doit être envoyé.

Un champ d'en-tête Retry-After peut être utilisé pour indiquer que la demande pourrait être couronnée de succès. Certaines des réponses importantes sont présentées ci-dessous.

600 Busy Everywhere

Ce code de réponse indique que l'appel dans la requête spécifiée devrait trouver réponse auprès d'autres endroits.

603 Decline

Cette réponse pourrait indiquer la partie appelée est occupée, ou simplement ne veut pas accepter l'appel.

604 Does Not Exist Anywhere

Cette réponse est semblable à la réponse 404 Not Found, mais indique que l'utilisateur dans la requête ne peut être trouvé nulle part.

Cette réponse devrait seulement être envoyée par un serveur ayant accès à toutes les informations sur l'utilisateur.

606 Not Acceptable

Cette réponse indique que certains aspects de la session souhaitée n'est pas acceptable pour l'UAS, et par conséquent, la session ne peut pas être établie.

La réponse peut contenir un champ d'en-tête `Warning` avec un code numérique décrivant exactement ce qui était pas acceptable.

La demande peut être relancée avec différentes informations de sessions médias.

Protocole SDP

- 1. Introduction
- 2. Sections SDP
 - Description de la session
 - Description du timing (obligatoire)
 - Description du média (si présent)
- 3. Interprétation

1. Introduction

Le protocole Session Description Protocol (SDP) est un protocole de communication de description de paramètres d'initialisation d'une session de diffusion en flux (streaming). L'Internet Engineering Task Force (IETF) a publié une première spécification comme une proposition de standard Internet en avril 1998 et a publié une spécification révisée comme le standard Internet [RFC 4566](#) en juillet 2006.

Le protocole SDP a été créé pour décrire des sessions de communication multimédia, par exemple pour l'annonce de la session, l'invitation à une session et la négociation de paramètres. Le protocole SDP ne livre pas le média lui-même. Il est utilisé par l'émetteur et le destinataire pour la négociation du type et du format du média, et les propriétés associées. L'ensemble des paramètres d'une session est souvent appelé un profil de session. Le SDP a été conçu pour être extensible et soutenir les nouveaux types et formats de médias.

SDP a commencé comme une composante du Session Announcement Protocol (SAP), mais a trouvé d'autres utilisations en conjonction avec le protocole Real-time Transport Protocol (RTP), le protocole Real-Time Streaming Protocol (RTSP), le protocole Session Initiation Protocol (SIP) et même comme un format autonome pour décrire des sessions de Multicast.

Source : https://fr.wikipedia.org/wiki/Session_Description_Protocol

SDP avec SIP fonctionne selon un modèle Offre/Réponse où chacun des protagonistes d'un dialogue indique ses capacités en terme de codage du media utilisé (voix, vidéo, ...) et choisissent les paramètres qui leur correspondent pour le placement du trafic RTP.

Une charge SDP est embarquée dans le corps d'un message INVITE SIP ou un Re-INVITE SIP et dans la réponse 200 OK consécutive.

2. Sections SDP

SDP est composé d'une série de lignes `<caractère>=<valeur>`, où `<caractère>` est un caractère alphabétique sensible à la casse et `<valeur>` est un texte structuré.

SDP se compose de trois sections principales:

- la session,
- le timing
- et les descriptions des médias.

Chaque message peut comporter plusieurs descriptions de synchronisation et de média, mais seulement une description de session.

La définition de ces sections et leurs contenus possibles sont les suivantes.

Toutes ces entrées peuvent être présentes dans un message SDP.

Description de la session

- v= (numéro de version du protocole, actuellement seulement 0)
- o= (identificateur d'origine et de session: nom d'utilisateur, identifiant, numéro de version, adresse réseau)
- s= (nom de la session: obligatoire avec au moins un caractère codé UTF-8)
- i=* (titre de la session ou information courte)
- u=* (URI de description)
- e=* (zéro ou plus adresse e-mail avec le nom facultatif des contacts)
- p=* (zéro ou plus numéro de téléphone avec le nom optionnel des contacts)
- c=* (informations de connexion - pas nécessaire si inclus dans tous les médias)
- b=* (zéro ou plus de lignes d'information de bande passante)
- Une ou plusieurs descriptions de temps (lignes "t=" et "r=" , voir ci-dessous)
- z= * (ajustements du fuseau horaire)
- k=* (clé de chiffrement)
- a=* (zéro ou plus de lignes d'attribut de session)
- Zéro ou plus de descriptions de média (chacune commençant par une ligne "m =", voir ci-dessous)

Description du timing (obligatoire)

- t= (time the session is active)
- r=* (zéro ou plus de répétition)

Description du média (si présent)

- m= (nom du média et adresse du transport)
- i=* (titre du média ou champ d'information)
- c=* (informations de connexion - facultatif si inclus au niveau de la session)
- b=* (zéro ou plus de lignes d'information de bande passante)
- k=* (clé de chiffrement)
- a=* (zéro ou plus de lignes d'attribut multimédia - surpassant les lignes d'attributs)

3. Interprétation

```
Session Description Protocol
  Session Description Protocol Version (v): 0
  Owner/Creator, Session Id (o): linphone 1108 2378 IN IP4 192.168.1.7
  Session Name (s): Talk
  Connection Information (c): IN IP4 192.168.1.7
  Time Description, active time (t): 0 0
  Session Attribute (a): rtcp-xr:rcvr-rtt=all:10000 stat-summary=loss,dup,jitt,TTL voip-metrics
```

```

Media Description, name and address (m): audio 7078 RTP/AVP 96 97 98 99 0 8 101 100 102
  Media Type: audio
  Media Port: 7078
  Media Protocol: RTP/AVP
  Media Format: DynamicRTP-Type-96
  Media Format: DynamicRTP-Type-97
  Media Format: DynamicRTP-Type-98
  Media Format: DynamicRTP-Type-99
  Media Format: ITU-T G.711 PCMU
  Media Format: ITU-T G.711 PCMA
  Media Format: DynamicRTP-Type-101
  Media Format: DynamicRTP-Type-100
  Media Format: DynamicRTP-Type-102
Media Attribute (a): rtpmap:96 opus/48000/2
Media Attribute (a): fmp:96 useinbandfec=1
Media Attribute (a): rtpmap:97 SILK/16000
Media Attribute (a): rtpmap:98 speex/16000
Media Attribute (a): fmp:98 vbr=on
Media Attribute (a): rtpmap:99 speex/8000
Media Attribute (a): fmp:99 vbr=on
Media Attribute (a): rtpmap:101 telephone-event/48000
Media Attribute (a): rtpmap:100 telephone-event/16000
Media Attribute (a): rtpmap:102 telephone-event/8000
Media Description, name and address (m): video 9078 RTP/AVP 96 97
Media Attribute (a): rtpmap:96 VP8/90000
Media Attribute (a): rtpmap:97 H264/90000
Media Attribute (a): fmp:97 profile-level-id=42801F

```

Source : <https://www.cloudshark.org/captures/f22c8443205b>

On retiendra trois lignes qui ont leur importance dans une analyse SDP :

- **c=** indique l'adresse du User Agent à utiliser pour le transport de média ici `IN IP4 192.168.1.7`.
- **m=** indique le type de media (voix, vidéo, ...), une ligne pour chaque media suivie de lignes **a=** d'attribut (codec supporté, entre autres).
- **a=rtpmap:**<payload type> <encoding name>/<clock rate> [/<encoding parameters>] indique ligne par ligne pour un type de media le codec supporté (voir [SDP Attributes Section 6 du RFC 4666](#)).

Ces différentes charges, leur codes et leurs RFC de référence sont recensés par l'IANA en tant que paramètres RTP à la page <https://www.iana.org/assignments/rtp-parameters/rtp-parameters.xhtml>.

Enregistrement

- 1. REGISTER simple
- 2. Annuler un REGISTER
- 3. REGISTER avec Authentification MD5
 - Procédure REGISTER avec Authentification MD5
- 4. REGISTER avec utilisateur erroné
- 5. REGISTER avec mot de passe erroné
- 6. Découverte de mots de passe

Les transactions non-INVITE n'utilisent pas de ACK. Ce sont de simples interactions demande-réponse.

Source : [RFC 3261 17.1.2.1 Overview of the non-INVITE Transaction](#)

SIP offre une capacité de localisation. Si un utilisateur veut initier une session avec un autre utilisateur, SIP doit découvrir le ou les hôtes auxquels l'utilisateur de destination est couramment joignable. Ce processus de découverte est fréquemment accompli par les éléments de réseau SIP tels que des serveurs mandataires et les serveurs de redirection qui sont chargés de recevoir une demande, de déterminer où l'envoyer sur la base de la connaissance de la localisation de l'utilisateur, et ensuite de l'y envoyer. Pour ce faire, les éléments de réseau SIP consultent un service abstrait connu sous le nom de service de localisation, qui fournit les liens d'adresse pour un domaine particulier. Ces liens d'adresse transposent un URI SIP ou SIPS entrant, `sip:bob@biloxi.com`, par exemple, en un ou plusieurs URI qui sont en quelque sorte "plus proches" de l'utilisateur désiré, `sip:bob@engineering.biloxi.com`, par exemple. Finalement, un mandataire va consulter un service de localisation qui fait correspondre un URI reçu à l'agent ou aux agents d'utilisateur dans lesquels le receveur désiré réside actuellement.

L'enregistrement crée des liens dans un service de localisation pour un domaine particulier qui associe un URI AOR avec une ou plusieurs adresses de contact. Et donc, lorsqu'un mandataire pour ce domaine reçoit une requête dont le "Request-URI" correspond à l'AOR, le mandataire va transmettre la demande aux adresses de contact enregistrées à cette AOR.

L'enregistrement a pour conséquence l'envoi d'une requête REGISTER à un type particulier d'UAS connu sous le nom de REGISTRAR. Un REGISTRAR agit comme extrémité frontale du service de localisation pour un domaine, lisant et écrivant les transpositions sur la base du contenu des demandes REGISTER. Ce service de localisation est alors normalement consulté par un serveur mandataire qui est chargé d'acheminer les demandes pour ce domaine. Dans un contexte avec un PABX ou un fournisseur de service, c'est tout simplement le serveur de téléphonie (Mandataire ou B2BUA) qui remplit ce rôle.

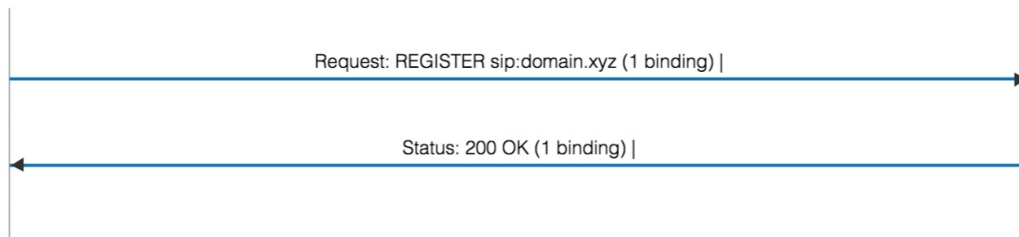
Source : [RFC 3261, 10.1. Registrations, Overview](#)

1. REGISTER simple

Entre l'UAC 172.16.98.182 et l'UAS 172.16.98.102.

172.16.98.182

172.16.98.102



Source : <https://www.cloudshark.org/captures/de9c2cf75368>

```

REGISTER sip:domain.xyz SIP/2.0
Via: SIP/2.0/UDP 172.16.98.182:5060;rport;branch=z9hG4bK1013779528
From: <sip:telephone1@domain.xyz>;tag=1182049044
To: <sip:telephone1@domain.xyz>
Call-ID: 1077245679
CSeq: 1 REGISTER
Contact: <sip:telephone1@172.16.98.182;line=f0d50acfece3520>
Max-Forwards: 70
User-Agent: Linphone/3.6.1 (eXosip2/4.0.0)
Expires: 3600
Content-Length: 0
  
```

```

SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.16.98.182:5060;rport=5060;branch=z9hG4bK1013779528
Contact: <sip:telephone1@172.16.98.182;line=f0d50acfece3520>;expires=3600
To: <sip:telephone1@domain.xyz>;tag=2021e220
From: <sip:telephone1@domain.xyz>;tag=1182049044
Call-ID: 1077245679
CSeq: 1 REGISTER
User-Agent: repro 1.9.7
Content-Length: 0
  
```

2. Annuler un REGISTER

Normalement, les données d'enregistrement sont abandonnées tant qu'il n'y a pas eu de rafraîchissement. Pour annuler explicitement un enregistrement, une nouvelle requête REGISTER avec un champ `Expires: 0` dans :

```

REGISTER sip:domain.xyz SIP/2.0
Via: SIP/2.0/UDP 172.16.98.182:5060;rport;branch=z9hG4bK410623604
From: <sip:telephone1@domain.xyz>;tag=1182049044
To: <sip:telephone1@domain.xyz>
Call-ID: 1077245679
CSeq: 2 REGISTER
Contact: <sip:telephone1@172.16.98.182;line=f0d50acfece3520>
Max-Forwards: 70
User-Agent: Linphone/3.6.1 (eXosip2/4.0.0)
Expires: 0
Content-Length: 0
  
```

Note : si le champ `Expires` était absent, l'effet de "dés-enregistrement" serait identique. Si le champs `Expires` utilisé peut être celui de l'en-tête, l'information pourrait se trouver comme paramètre du champ `Contact` .

```

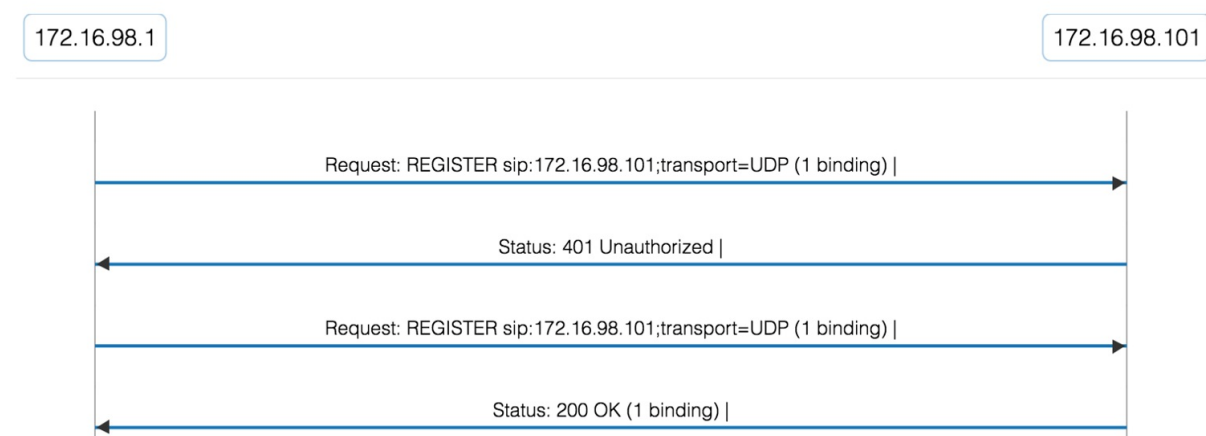
SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.16.98.182:5060;rport=5060;branch=z9hG4bK410623604
To: <sip:telephone1@domain.xyz>;tag=09697f6c
From: <sip:telephone1@domain.xyz>;tag=1182049044
  
```

```
Call-ID: 1077245679
CSeq: 2 REGISTER
User-Agent: repro 1.9.7
Content-Length: 0
```

3. REGISTER avec Authentification MD5

Le [RFC 3261](#) recommande que les requêtes SIP soient authentifiées comme indiqué dans le [RFC 2617](#) (HTTP Authentication: Basic and Digest Access Authentication).

Entre l'UAC 172.16.98.1 et 172.16.98.101.



Source : <https://www.cloudshark.org/captures/423ab1d45e27>

Procédure REGISTER avec Authentification MD5

```
REGISTER sip:172.16.98.101;transport=UDP SIP/2.0
Via: SIP/2.0/UDP 172.16.98.1:42952;branch=z9hG4bK-d8754z-531c0fb072273b86-1---d8754z-
Max-Forwards: 70
Contact: <sip:telephone1@172.16.98.1:42952;rinstance=564a2ed14798bc07;transport=UDP>
To: <sip:telephone1@172.16.98.101;transport=UDP>
From: <sip:telephone1@172.16.98.101;transport=UDP>;tag=0976af15
Call-ID: MzlhNDk0Nzg0MjE0MTEyNzRlM2VhNGYyYjgzYzc0MzA.
CSeq: 1 REGISTER
Expires: 3600
Allow: INVITE, ACK, CANCEL, BYE, NOTIFY, REFER, MESSAGE, OPTIONS, INFO, SUBSCRIBE
Supported: replaces, norefersub, extended-refer, timer, X-cisco-serviceuri
User-Agent: Z 3.3.21933 r21903
Allow-Events: presence, kpm1
Content-Length: 0
```

Le serveur SIP répond avec **401 Unauthorized** et indique un champ nécessaire à l'authentification `WWW-Authenticate: Digest algorithm=MD5, realm="asterisk", nonce="26235be0"`

```
SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP 172.16.98.1:42952;branch=z9hG4bK-d8754z-531c0fb072273b86-1---d8754z-;received=172.16.98.1
From: <sip:telephone1@172.16.98.101;transport=UDP>;tag=0976af15
To: <sip:telephone1@172.16.98.101;transport=UDP>;tag=as108b6b97
Call-ID: MzlhNDk0Nzg0MjE0MTEyNzRlM2VhNGYyYjgzYzc0MzA.
CSeq: 1 REGISTER
Server: Asterisk PBX 11.7.0~dfsg-1ubuntu1
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH
Supported: replaces, timer
WWW-Authenticate: Digest algorithm=MD5, realm="asterisk", nonce="26235be0"
Content-Length: 0
```

Une nouvelle requête (voir Cseq: 2) fournit les paramètres adéquat : Authorization: Digest

```
username="telephone1",realm="asterisk",nonce="26235be0",uri="sip:172.16.98.101;transport=UDP",response="f76185db0be3a8ba2494f5fc4ea99eed",algorithm=MD5
```

```
REGISTER sip:172.16.98.101;transport=UDP SIP/2.0
Via: SIP/2.0/UDP 172.16.98.1:42952;branch=z9hG4bK-d8754z-8de228c832cb6988-1---d8754z-
Max-Forwards: 70
Contact: <sip:telephone1@172.16.98.1:42952;rinstance=564a2ed14798bc07;transport=UDP>
To: <sip:telephone1@172.16.98.101;transport=UDP>
From: <sip:telephone1@172.16.98.101;transport=UDP>;tag=0976af15
Call-ID: MzlhNDk0Nzg0MjE0MTEyNzRlM2VhNGYyYjgzYzc0MZA.
CSeq: 2 REGISTER
Expires: 3600
Allow: INVITE, ACK, CANCEL, BYE, NOTIFY, REFER, MESSAGE, OPTIONS, INFO, SUBSCRIBE
Supported: replaces, norefersub, extended-refer, timer, X-cisco-serviceuri
User-Agent: Z 3.3.21933 r21903
Authorization: Digest username="telephone1",realm="asterisk",nonce="26235be0",uri="sip:172.16.98.101;transport=
UDP",response="f76185db0be3a8ba2494f5fc4ea99eed",algorithm=MD5
Allow-Events: presence, kpml
Content-Length: 0
```

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.16.98.1:42952;branch=z9hG4bK-d8754z-8de228c832cb6988-1---d8754z-;received=172.16.98.1
From: <sip:telephone1@172.16.98.101;transport=UDP>;tag=0976af15
To: <sip:telephone1@172.16.98.101;transport=UDP>;tag=as108b6b97
Call-ID: MzlhNDk0Nzg0MjE0MTEyNzRlM2VhNGYyYjgzYzc0MZA.
CSeq: 2 REGISTER
Server: Asterisk PBX 11.7.0~dfsg-1ubuntu1
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH
Supported: replaces, timer
Expires: 3600
Contact: <sip:telephone1@172.16.98.1:42952;rinstance=564a2ed14798bc07;transport=UDP>;expires=3600
Date: Wed, 11 May 2016 11:55:19 GMT
Content-Length: 0
```

4. REGISTER avec utilisateur erroné

Dans cet exemple, l'UA tente de s'authentifier sous un nom d'utilisateur (telephone0) inexistant sur le serveur SIP. Le message d'erreur est "403 Forbidden".

Source : <https://www.cloudshark.org/captures/10222a758905>

5. REGISTER avec mot de passe erroné

Dans cet exemple, l'UA tente de s'authentifier avec un mot de passe erroné. Le message d'erreur est aussi dans ce cas "403 Forbidden".

Source : <https://www.cloudshark.org/captures/784f68cbb09a>

6. Découverte de mots de passe

Voir la page [Sécurité VoIP / SIP](#)

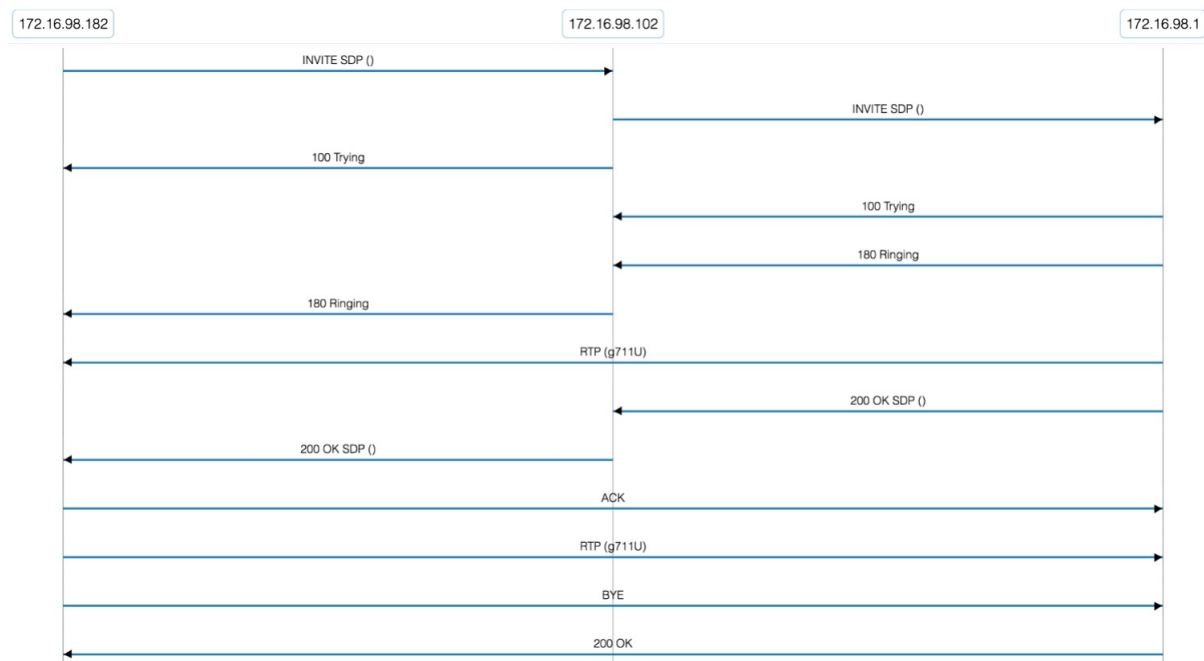
Proxy SIP UDP

- Scénario
- 1. INVITE (telephone1 → Proxy)
- 2. INVITE (Proxy → telephone2)
- 3. 100 Trying, 180 Ringing
- 4. 200 OK

Scénario

Dans ce scénario, on analyse un appel entre (sip:telephone1@domain.xyz) 172.16.98.182 et 172.16.98.1 (sip:telephone2@domain.xyz) à travers un proxy domain.xyz (172.16.98.102) dont le service est rendu en UDP.

- UAC : 172.16.98.182
- UAS : 172.16.98.1
- Proxy : 172.16.98.102



Source : <https://www.cloudshark.org/captures/5881e936996f>

Dans cette analyse, on sera attentif aux champs "Via", "Max-Forwards" et "Contact".

1. INVITE (telephone1 → Proxy)

2. INVITE (Proxy → telephone2)

Le champ "Via" est ajouté

3. 100 Trying, 180 Ringing

Réponses routées à travers le saut noté dans le deuxième champ "Via". La réponse "180 Ringing" du correspondant ajoute un *tag* au champ "To:".

4. 200 OK

La combinaison des champs To-tag From-Tag et Call-ID identifie le dialogue entre `telephone1` et `telephone2`

B2BUA

- [Conversation entre un Asterisk et deux téléphones](#)

Conversation entre un Asterisk et deux téléphones

Avec une topologie comprenant un B2BUA, le serveur établit un dialogue SIP respectivement avec chaque intervenant impliqué dans une session multimédia. Il est donc le seul à parler en SIP avec les différents UA.

Dans cet exemple l'utilisateur `telephone1` (`172.16.98.1`) tente de joindre le numéro de téléphone 2302. Cet appel est "routé" jusqu'au périphérique `telephone2` (`172.16.98.145`) par le B2BUA (`172.16.98.101`). Le B2BUA établit un dialogue avec chacun des deux UA qui doivent s'échanger une session multimédia. Dans ce cas, le B2BUA est UAS pour le terminal appelant et il est UAC pour le destinataire.

Il s'agit ici de constater la relation unique que chacun entretient avec le B2BUA (`172.16.98.101`).

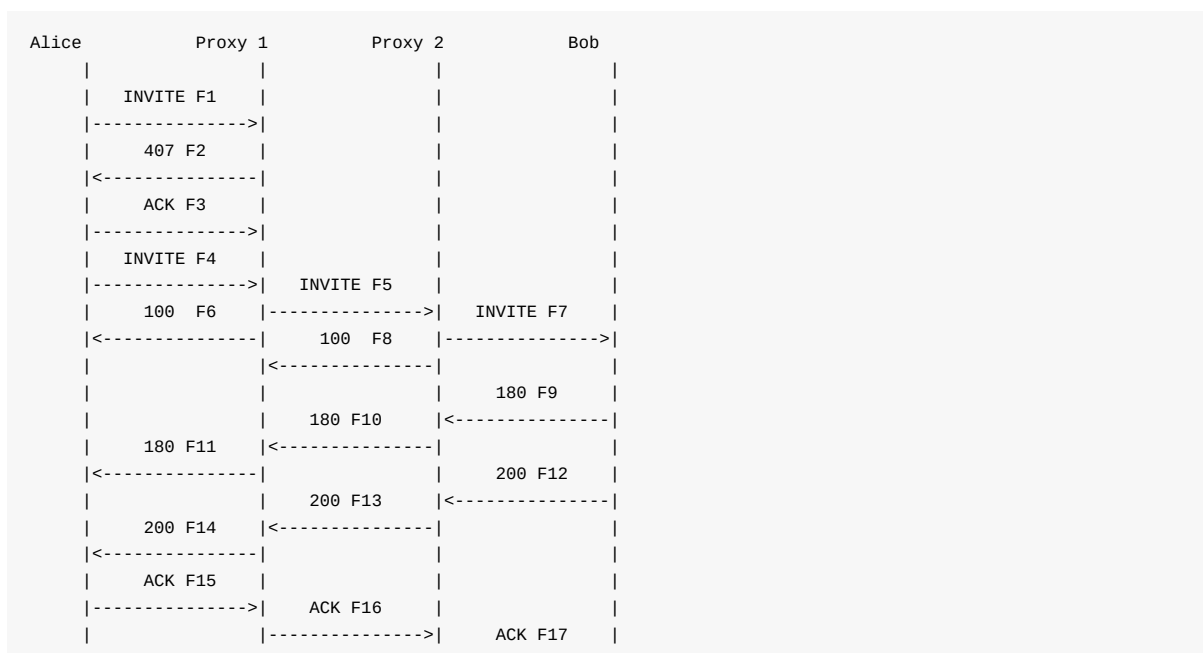
Source : <https://www.cloudshark.org/captures/b89478f0b2d8> (sip-2phones-B2BUA-Asterisk.pcapng)

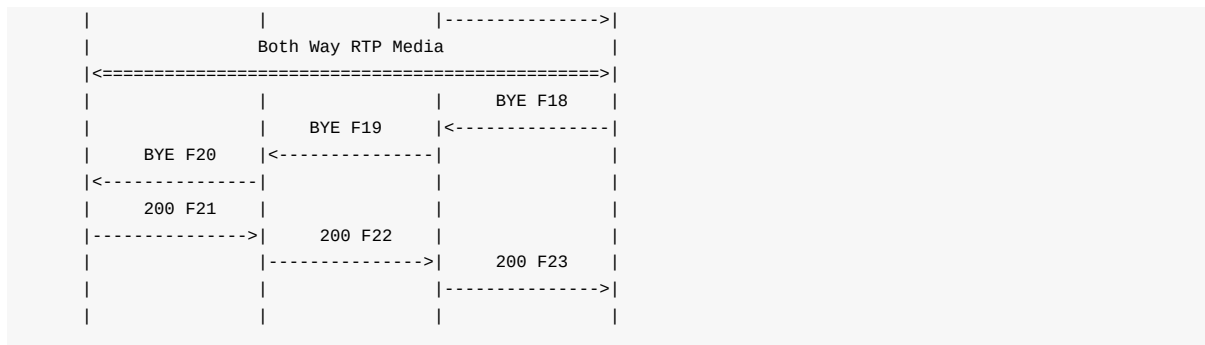
Flux SIP Trapéziodal

- Etablissement de session à travers deux proxy
 - F1 INVITE Alice -> Proxy 1
 - F2 407 Proxy Authorization Required Proxy 1 -> Alice
 - F3 ACK Alice -> Proxy 1
 - F4 INVITE Alice -> Proxy 1
 - F5 INVITE Proxy 1 -> Proxy 2
 - F6 100 Trying Proxy 1 -> Alice
 - F7 INVITE Proxy 2 -> Bob
 - F8 100 Trying Proxy 2 -> Proxy 1
 - F9 180 Ringing Bob -> Proxy 2
 - F10 180 Ringing Proxy 2 -> Proxy 1
 - F11 180 Ringing Proxy 1 -> Alice
 - F12 200 OK Bob -> Proxy 2
 - F13 200 OK Proxy 2 -> Proxy 1
 - F14 200 OK Proxy 1 -> Alice
 - F15 ACK Alice -> Proxy 1
 - F16 ACK Proxy 1 -> Proxy 2
 - F17 ACK Proxy 2 -> Bob
 - F18 BYE Bob -> Proxy 2
 - F19 BYE Proxy 2 -> Proxy 1
 - F20 BYE Proxy 1 -> Alice
 - F21 200 OK Alice -> Proxy 1
 - F22 200 OK Proxy 1 -> Proxy 2
 - F23 200 OK Proxy 2 -> Bob

Etablissement de session à travers deux proxy

Source : <https://tools.ietf.org/html/rfc3665#section-3.2>





Dans ce scénario, Alice réalise un appel à Bob en utilisant deux mandataires Proxy 1 et Proxy 2. Le message "INVITE" initial (F1) contient un en-tête "Route" pré-configuré avec l'adresse des deux mandataires Proxy 1 et Proxy 2 (Proxy 1 est configuré comme mandataire sortant (*outbound proxy*) par défaut pour Alice). La demande ne contient pas les crédits "Authorization" requis par Proxy 1, envoyant une réponse "407 Proxy Authorization" avec le "challenge" d'authentification.

Un nouvel invite (F4) est ensuite envoyé avec les crédits correctes et l'appel se réalise. L'appel se termine quand Bob se déconnecte en initiant un message BYE.

Proxy 1 insère un en-tête "Record-Route" au message "INVITE" pour s'assurer qu'il sera présent dans tous les messages et échanges associés.

Le "ACK" (F15) et le "BYE" (F18) ont tous deux un en-tête "Route".

F1 INVITE Alice -> Proxy 1

```

INVITE sip:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74b43
Max-Forwards: 70
Route: <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
Contact: <sip:alice@client.atlanta.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=alice 2890844526 2890844526 IN IP4 client.atlanta.example.com
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
  
```

F2 407 Proxy Authorization Required Proxy 1 -> Alice

Proxy 1 envoie un "challenge" à Alice pour l'authentification.

```

SIP/2.0 407 Proxy Authorization Required
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74b43
;received=192.0.2.101
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>;tag=3flal12sf
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 INVITE
Proxy-Authenticate: Digest realm="atlanta.example.com", qop="auth",
nonce="f84f1cec41e6cbe5aea9c8e88d359",
  
```

```
opaque="", stale=FALSE, algorithm=MD5
Content-Length: 0
```

F3 ACK Alice -> Proxy 1

Alice répond avec ses crédits en renvoyant un nouvel "INVITE".

```
ACK sip:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74b43
Max-Forwards: 70
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>;tag=3flal12sf
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 ACK
Content-Length: 0
```

F4 INVITE Alice -> Proxy 1

Proxy 1 accepte les crédits et transfère le message "INVITE" au Proxy 2. Le logiciel client d'Alice se prépare à recevoir des données sur le 49172.

```
INVITE sip:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 INVITE
Contact: <sip:alice@client.atlanta.example.com;transport=tcp>
Proxy-Authorization: Digest username="alice",
  realm="atlanta.example.com",
  nonce="wf84f1ceczx41ae6cbe5aea9c8e88d359", opaque="",
  uri="sip:bob@biloxi.example.com",
  response="42ce3cef44b22f50c6a6071bc8"
Content-Type: application/sdp
Content-Length: 151

v=0
o=alice 2890844526 2890844526 IN IP4 client.atlanta.example.com
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F5 INVITE Proxy 1 -> Proxy 2

```
INVITE sip:bob@biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 INVITE
Contact: <sip:alice@client.atlanta.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: 151
```

```

v=0
o=alice 2890844526 2890844526 IN IP4 client.atlanta.example.com
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

```

F6 100 Trying Proxy 1 -> Alice

```

SIP/2.0 100 Trying
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 INVITE
Content-Length: 0

```

F7 INVITE Proxy 2 -> Bob

```

INVITE sip:bob@client.biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP ss2.biloxi.example.com:5060;branch=z9hG4bK721e4.1
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 68
Record-Route: <sip:ss2.biloxi.example.com;lr>,
<sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 INVITE
Contact: <sip:alice@client.atlanta.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

```

```

v=0
o=alice 2890844526 2890844526 IN IP4 client.atlanta.example.com
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

```

F8 100 Trying Proxy 2 -> Proxy 1

```

SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 INVITE
Content-Length: 0

```

F9 180 Ringing Bob -> Proxy 2

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP ss2.biloxi.example.com:5060;branch=z9hG4bK721e4.1
    ;received=192.0.2.222
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
    ;received=192.0.2.111
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
    ;received=192.0.2.101
Record-Route: <sip:ss2.biloxi.example.com;lr>,
    <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=314159
Call-ID: 3848276298220188511@atlanta.example.com
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
CSeq: 2 INVITE
Content-Length: 0
```

F10 180 Ringing Proxy 2 -> Proxy 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
    ;received=192.0.2.111
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
    ;received=192.0.2.101
Record-Route: <sip:ss2.biloxi.example.com;lr>,
    <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=314159
Call-ID: 3848276298220188511@atlanta.example.com
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
CSeq: 2 INVITE
Content-Length: 0
```

F11 180 Ringing Proxy 1 -> Alice

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
    ;received=192.0.2.101
Record-Route: <sip:ss2.biloxi.example.com;lr>,
    <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=314159
Call-ID: 3848276298220188511@atlanta.example.com
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
CSeq: 2 INVITE
Content-Length: 0
```

F12 200 OK Bob -> Proxy 2

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss2.biloxi.example.com:5060;branch=z9hG4bK721e4.1
    ;received=192.0.2.222
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
    ;received=192.0.2.111
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
    ;received=192.0.2.101
Record-Route: <sip:ss2.biloxi.example.com;lr>,
    <sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=314159
```

```

Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 INVITE
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: 147

v=0
o=bob 2890844527 2890844527 IN IP4 client.biloxi.example.com
s=-
c=IN IP4 192.0.2.201
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

```

F13 200 OK Proxy 2 -> Proxy 1

```

SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss2.biloxi.example.com;lr>,
<sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=314159
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 INVITE
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: 147

v=0
o=bob 2890844527 2890844527 IN IP4 client.biloxi.example.com
s=-
c=IN IP4 192.0.2.201
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

```

F14 200 OK Proxy 1 -> Alice

```

SIP/2.0 200 OK
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss2.biloxi.example.com;lr>,
<sip:ss1.atlanta.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
To: Bob <sip:bob@biloxi.example.com>;tag=314159
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 INVITE
Contact: <sip:bob@client.biloxi.example.com;transport=tcp>
Content-Type: application/sdp
Content-Length: 147

v=0
o=bob 2890844527 2890844527 IN IP4 client.biloxi.example.com
s=-
c=IN IP4 192.0.2.201
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

```

F15 ACK Alice -> Proxy 1


```
ACK sip:bob@client.biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74b76
Max-Forwards: 70
Route: <sip:ss1.atlanta.example.com;lr>,
      <sip:ss2.biloxi.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>;tag=314159
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 ACK
Content-Length: 0
```

F16 ACK Proxy 1 -> Proxy 2

```
ACK sip:bob@client.biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74b76
      ;received=192.0.2.101
Max-Forwards: 69
Route: <sip:ss2.biloxi.example.com;lr>
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>;tag=314159
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 ACK
Content-Length: 0
```

F17 ACK Proxy 2 -> Bob

```
ACK sip:bob@client.biloxi.example.com SIP/2.0
Via: SIP/2.0/TCP ss2.biloxi.example.com:5060;branch=z9hG4bK721e4.1
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
      ;received=192.0.2.111
Via: SIP/2.0/TCP client.atlanta.example.com:5060;branch=z9hG4bK74b76
      ;received=192.0.2.101
Max-Forwards: 68
From: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
To: Bob <sip:bob@biloxi.example.com>;tag=314159
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 2 ACK
Content-Length: 0
```

Les flux RTP sont établis entre Alice et Bob.

F18 BYE Bob -> Proxy 2

Bob raccroche.

Notez que champ "CSeq" **n'est pas** 3. Alice et Bob maintiennent leur propre numéros "CSeq".

```
BYE sip:alice@client.atlanta.example.com SIP/2.0
Via: SIP/2.0/TCP client.biloxi.example.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
Route: <sip:ss2.biloxi.example.com;lr>,
      <sip:ss1.atlanta.example.com;lr>
From: Bob <sip:bob@biloxi.example.com>;tag=314159
To: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 BYE
Content-Length: 0
```

F19 BYE Proxy 2 -> Proxy 1

```
BYE sip:alice@client.atlanta.example.com SIP/2.0
Via: SIP/2.0/TCP ss2.biloxi.example.com:5060;branch=z9hG4bK721e4.1
Via: SIP/2.0/TCP client.biloxi.example.com:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
Max-Forwards: 69
Route: <sip:ss1.atlanta.example.com;lr>
From: Bob <sip:bob@biloxi.example.com>;tag=314159
To: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 BYE
Content-Length: 0
```

F20 BYE Proxy 1 -> Alice

```
BYE sip:alice@client.atlanta.example.com SIP/2.0
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ss2.biloxi.example.com:5060;branch=z9hG4bK721e4.1
;received=192.0.2.222
Via: SIP/2.0/TCP client.biloxi.example.com:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
Max-Forwards: 68
From: Bob <sip:bob@biloxi.example.com>;tag=314159
To: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 BYE
Content-Length: 0
```

F21 200 OK Alice -> Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.atlanta.example.com:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ss2.biloxi.example.com:5060;branch=z9hG4bK721e4.1
;received=192.0.2.222
Via: SIP/2.0/TCP client.biloxi.example.com:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
From: Bob <sip:bob@biloxi.example.com>;tag=314159
To: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 BYE
Content-Length: 0
```

F22 200 OK Proxy 1 -> Proxy 2

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss2.biloxi.example.com:5060;branch=z9hG4bK721e4.1
;received=192.0.2.222
Via: SIP/2.0/TCP client.biloxi.example.com:5060;branch=z9hG4bKnashds7
;received=192.0.2.101
From: Bob <sip:bob@biloxi.example.com>;tag=314159
To: Alice <sip:alice@atlanta.example.com>;tag=9fxced76sl
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 BYE
Content-Length: 0
```

F23 200 OK Proxy 2 -> Bob

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP client.biloxi.example.com:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
From: Bob <sip:bob@biloxi.example.com>;tag=314159
To: Alice <sip:alice@atlanta.example.com>;tag=9fxced76s1
Call-ID: 3848276298220188511@atlanta.example.com
CSeq: 1 BYE
Content-Length: 0
```

Extensions SIP

- [Introduction](#)
- [Méthode SUBSCRIBE](#)
- [Méthode PUBLISH](#)
- [Méthode NOTIFY](#)
 - [MWI](#)
 - [A1 : Subscriber -> Notifier](#)
 - [A2 : Notifier -> Subscriber](#)
 - [A3 Notifier -> Subscriber](#)
 - [A4: Subscriber -> Notifier](#)
 - [A5: Notifier -> Subscriber](#)
 - [A6: Subscriber -> Notifier](#)
 - [A7: Subscriber -> Notifier](#)
 - [A8: Notifier -> Subscriber](#)
 - [A9: Notifier -> Subscriber](#)
 - [A10: Subscriber -> Notifier](#)
 - [A11: Subscriber -> Notifier](#)
 - [A12: Notifier -> Subscriber](#)
 - [A13: Notifier -> Subscriber](#)
 - [A14: Subscriber -> Notifier](#)
- [Méthode REFER](#)
 - [Premier message \(F1\)](#)
 - [Second message \(F2\)](#)
 - [Troisième message \(F3\)](#)
 - [Quatrième message \(F4\)](#)
 - [Cinquième message \(F5\)](#)
 - [Sixième message \(F6\)](#)
- [Méthode MESSAGE](#)
- [Méthode INFO](#)
- [Méthode PRACK](#)
- [Méthode UPDATE](#)

Introduction

- Publication et notification d'événements, MWI (SUBSCRIBE, NOTIFY, PUBLISH)
- Transfert d'appel (REFER)
- Messagerie Instantannée (MESSAGE)

Le [RFC 5359](#) intitulé "Session Initiation Protocol Service Examples" fournit une suite d'exemples d'applications SIP qui utilisent les méthodes SUBSCRIBE, NOTIFY et REFER.

On trouvera ici une description succincte de ces messages étendus au-delà du RFC 3261 principal.

Enfin, on citera aussi les méthodes MESSAGE, INFO, PRACK et UPDATE.

Méthode SUBSCRIBE

[RFC 6665](#) (SIP-Specific Event Notification)

La méthode SUBSCRIBE est utilisée pour créer un abonnement entre une application cliente qui désire obtenir des informations d'un service et le service qui fournit ces informations. Par exemple, un téléphone SIP peut souscrire à un service de messagerie vocale afin d'allumer un MWI (Message Waiting Indicator) en rouge lorsqu'un message est en attente et de l'éteindre quand tous les messages sont lus. Un autre exemple pourrait être un premier intervenant souscrivant à un système d'appel d'urgence pour prendre connaissance de l'appelant de ce service d'urgence. Enfin, une application sur le smartphone d'un médecin peut souscrire à un électrocardiogramme pour savoir quand la lecture de l'ECG indique un problème significatif pour le patient.

Méthode PUBLISH

[RFC 3903](#) (Session Initiation Protocol (SIP) Extension for Event State Publication)

La méthode PUBLISH est utilisée lorsqu'un service a quelque chose à signaler. Dans notre exemple précédent, un téléphone SIP s'abonne à un service de messagerie vocale. Lorsque ce service de messagerie vocale veut signaler un nouveau message, il envoie un message PUBLISH qui contient des informations sur la messagerie vocale et l'appelant qui l'a quitté. De manière semblable, le service d'urgence enverrait un message PUBLISH lorsque quelqu'un a composé le numéro d'urgence. Ce message de publication contiendrait des informations de base comme le numéro de téléphone de l'appelant, mais il pourrait également contenir ses coordonnées GPS ainsi que l'emplacement d'une caméra vidéo en circuit fermé près de lui. Un message de publication pour l'électrocardiogramme peut contenir le nom du patient, le numéro de sa chambre et la lecture ECG.

Méthode NOTIFY

Lorsqu'un utilisateur, un périphérique ou une application s'abonne, il n'envoie pas le message SUBSCRIBE au service auquel il s'inscrit. A la place, il l'envoie à un fournisseur (*broker*) (par exemple, un Presence Server) qui gère les abonnements de tous les utilisateurs, les périphériques ou les applications peuvent créer pour ce service ou tout autre. Par conséquent, lorsqu'une entité SIP envoie un message PUBLISH, elle n'envoie pas ce message directement à l'abonné. Il parle également au fournisseur d'abonnement. Il incombe au fournisseur d'accepter les messages SUBSCRIBE pour créer des liens entre les abonnés et les services. Lorsque le fournisseur reçoit une publication, il recherche dans sa table de liens pour informer qui s'intéresse aux informations contenues dans le message PUBLISH et envoie les messages NOTIFY aux utilisateurs, aux applications ou aux périphériques. Cela permet à un seul message PUBLISH de générer plusieurs messages de notification lorsque plus d'un abonnement existe pour ce service.

Subscriber	Notifier
-----SUBSCRIBE----->	Request state subscription
<-----200-----	Acknowledge subscription
<-----NOTIFY-----	Return current state information
-----200----->	
<-----NOTIFY-----	Return current state information
-----200----->	

MWI

Les MWI (Message Waiting Indicator [RFC 3842](#)), sont un signal audio ou visuel qui réagit en fonction d'un événement.

Dans cet exemple, téléphone IP d'Alice s'abonne aux messages de statut d'Alice.

Subscriber	Notifier
A1: SUBSCRIBE (new)	
----->	

```

| A2: 200 OK |
|<-----|
| A3: NOTIFY (sync) |
|<-----|
| A4: 200 OK |
|----->|
| |
| A5: NOTIFY (change) |
|<-----|
| A6: 200 OK |
|----->|
| |
| A7: (re)SUBSCRIBE |
|----->|
| A8: 200 OK |
|<-----|
| A9: NOTIFY (sync) |
|<-----|
| A10: 200 OK |
|----->|
| |
| A11: (un)SUBSCRIBE |
|----->|
| A12: 200 OK |
|<-----|
| A13: NOTIFY (sync) |
|<-----|
| A14: 200 OK |
|----->|

```

A1 : Subscriber -> Notifier

"Subscriber" (téléphone d'Alice) -> "Notifier" (Passerelle de boîte vocale d'Alice)

Abonnement par Alice aux messages de statut d'Alice pendant 1 jour.

```

SUBSCRIBE sip:alice@vmail.example.com SIP/2.0
To: <sip:alice@example.com>
From: <sip:alice@example.com>;tag=78923
Date: Mon, 10 Jul 2000 03:55:06 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 4 SUBSCRIBE
Contact: <sip:alice@alice-phone.example.com>
Event: message-summary
Expires: 86400
Accept: application/simple-message-summary
Content-Length: 0

```

A2 : Notifier -> Subscriber

"Notifier" -> "Subscriber"

```

SIP/2.0 200 OK
To: <sip:alice@example.com>;tag=4442
From: <sip:alice@example.com>;tag=78923
Date: Mon, 10 Jul 2000 03:55:07 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 4 SUBSCRIBE

```

```
Expires: 86400
Content-Length: 0
```

A3 Notifier -> Subscriber

"Notifier" -> "Subscriber"

Synchronisation immédiate de l'état courant : 2 nouveau messages et 8 anciens dont 2 urgents.

```
NOTIFY sip:alice@alice-phone.example.com SIP/2.0
To: <sip:alice@example.com>;tag=78923
From: <sip:alice@example.com>;tag=4442
Date: Mon, 10 Jul 2000 03:55:07 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 20 NOTIFY
Contact: <sip:alice@vmail.example.com>
Event: message-summary
Subscription-State: active
Content-Type: application/simple-message-summary
Content-Length: 99

Messages-Waiting: yes
Message-Account: sip:alice@vmail.example.com
Voice-Message: 2/8 (0/2)
```

A4: Subscriber -> Notifier

```
SIP/2.0 200 OK
To: <sip:alice@example.com>;tag=78923
From: <sip:alice@example.com>;tag=4442
Date: Mon, 10 Jul 2000 03:55:08 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 20 NOTIFY
Content-Length: 0
```

A5: Notifier -> Subscriber

"Notifier" -> "Subscriber"

Voici la notification d'un nouveau message.

Des en-têtes sont adaptés pour chaque nouveau message ajouté.

```
NOTIFY sip:alice@alice-phone.example.com SIP/2.0
To: <sip:alice@example.com>;tag=78923
From: <sip:alice@example.com>;tag=4442
Date: Mon, 10 Jul 2000 04:28:53 GMT
Contact: <sip:alice@vmail.example.com>
Call-ID: 1349882@alice-phone.example.com
CSeq: 31 NOTIFY
Event: message-summary
Subscription-State: active
Content-Type: application/simple-message-summary
Content-Length: 503

Messages-Waiting: yes
Message-Account: sip:alice@vmail.example.com
Voice-Message: 4/8 (1/2)

To: <alice@atlanta.example.com>
From: <bob@biloxi.example.com>
Subject: carpool tomorrow?
```

```
Date: Sun, 09 Jul 2000 21:23:01 -0700
Priority: normal
Message-ID: 13784434989@vmail.example.com

To: <alice@example.com>
From: <cathy-the-bob@example.com>
Subject: HELP! at home ill, present for me please
Date: Sun, 09 Jul 2000 21:25:12 -0700
Priority: urgent
Message-ID: 13684434990@vmail.example.com
```

A6: Subscriber -> Notifier

"Subscriber" -> "Notifier"

```
SIP/2.0 200 OK
To: <sip:alice@example.com>;tag=78923
From: <sip:alice@example.com>;tag=4442
Date: Mon, 10 Jul 2000 04:28:53 GMT
Call-ID: 1349882@alice-phone.example.com
CSeq: 31 NOTIFY
Content-Length: 0
```

A7: Subscriber -> Notifier

"Subscriber" -> "Notifier"

Rafraîchissement de l'abonnement.

```
SUBSCRIBE sip:alice@vmail.example.com SIP/2.0
To: <sip:alice@example.com>;tag=4442
From: <sip:alice@example.com>;tag=78923
Date: Mon, 10 Jul 2000 15:55:06 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 8 SUBSCRIBE
Contact: <sip:alice@alice-phone.example.com>
Event: message-summary
Expires: 86400
Accept: application/simple-message-summary
Content-Length: 0
```

A8: Notifier -> Subscriber

"Notifier" -> "Subscriber"

```
SIP/2.0 200 OK
To: <sip:alice@example.com>;tag=4442
From: <sip:alice@example.com>;tag=78923
Date: Mon, 10 Jul 2000 15:55:07 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 8 SUBSCRIBE
Contact: <sip:alice@alice-phone.example.com>
Expires: 86400
Content-Length: 0
```

A9: Notifier -> Subscriber

Notifier -> Subscriber

Synchronisation immédiate de l'état courant.


```
NOTIFY sip:alice@alice-phone.example.com SIP/2.0
To: <sip:alice@example.com>;tag=78923
From: <sip:alice@example.com>;tag=4442
Date: Mon, 10 Jul 2000 15:55:07 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 47 NOTIFY
Contact: <sip:alice@vmail.example.com>
Event: message-summary
Subscription-State: active
Content-Type: application/simple-message-summary
Content-Length: 99

Messages-Waiting: yes
Message-Account: sip:alice@vmail.example.com
Voice-Message: 4/8 (1/2)
```

A10: Subscriber -> Notifier

```
SIP/2.0 200 OK
To: <sip:alice@example.com>;tag=78923
From: <sip:alice@example.com>;tag=4442
Date: Mon, 10 Jul 2000 15:55:08 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 47 NOTIFY
Contact: <sip:alice@vmail.example.com>
```

A11: Subscriber -> Notifier

"Subscriber" -> "Notifier"

Alice retire son abonnement après sa déconnexion.

```
SUBSCRIBE sip:alice@vmail.example.com SIP/2.0
To: <sip:alice@example.com>;tag=4442
From: <sip:alice@example.com>;tag=78923
Date: Mon, 10 Jul 2000 19:35:06 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 17 SUBSCRIBE
Contact: <sip:alice@alice-phone.example.com>
Event: message-summary
Expires: 0
Accept: application/simple-message-summary
Content-Length: 0
```

A12: Notifier -> Subscriber

"Notifier" -> "Subscriber"

```
SIP/2.0 200 OK
To: <sip:alice@example.com>;tag=4442
From: <sip:alice@example.com>;tag=78923
Date: Mon, 10 Jul 2000 19:35:07 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 17 SUBSCRIBE
Contact: <sip:alice@alice-phone.example.com>
Expires: 0
Content-Length: 0
```

A13: Notifier -> Subscriber

Notifier -> Subscriber

Synchronisation immédiate de l'état courant que le "Subscriber" pourrait désormais ignorer.

```
NOTIFY sip:alice@alice-phone.example.com SIP/2.0
To: <sip:alice@example.com>;tag=78923
From: <sip:alice@example.com>;tag=4442
Date: Mon, 10 Jul 2000 19:35:07 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 56 NOTIFY
Contact: <sip:alice@vmail.example.com>
Event: message-summary
Subscription-State: terminated;reason=timeout
Content-Type: application/simple-message-summary
Content-Length: 99
Messages-Waiting: yes
Message-Account: sip:alice@vmail.example.com
Voice-Message: 4/8 (1/2)
```

A14: Subscriber -> Notifier

"Subscriber" -> "Notifier"

```
SIP/2.0 200 OK
To: <sip:alice@example.com>;tag=78923
From: <sip:alice@example.com>;tag=4442
Date: Mon, 10 Jul 2000 19:35:08 GMT
Call-Id: 1349882@alice-phone.example.com
CSeq: 56 NOTIFY
Event: message-summary
Content-Length: 0
```

Méthode REFER

La méthode REFER ([RFC 3515](#)) peut être utilisée dans un grand nombre d'applications, notamment le transfert d'appel.

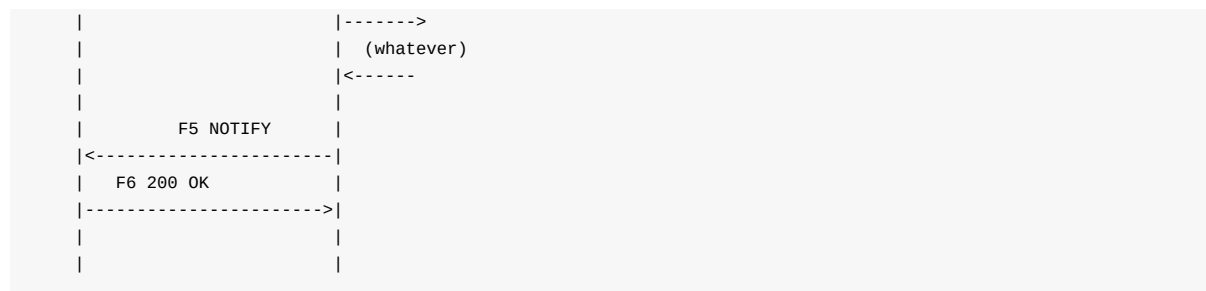
Par exemple si Alice est en conversation avec Bob, elle peut donner instruction à son UA d'envoyer une demande SIP REFER au UA de Bob en lui fournissant les information de "Contact" de Carol. En considérant que Bob donne sa permission, le UA de Bob tentera d'appeler Carol en utilisant le "Contact" transmis. Ensuite, l'UA de Bob informera l'UA d'Alice du succès de la tentative.

La méthode REFER indique au récepteur identifié par un "Request-URI" de contacter une ressource tierce en utilisant les informations de "Contact" dans la requête.

Un message NOTIFY donne l'état de la tentative.

```

Agent A                               Agent B
|                                     |
|   F1 REFER                         |
|----->|
|   F2 202 Accepted                  |
|<-----|
|   F3 NOTIFY                        |
|<-----|
|   F4 200 OK                        |
|----->|
|                                     |
|                                     |
```



Voici les exemples des messages entre l' *Agent A* et l' *Agent B*, faisant référence à quelque chose (whatever) que l' *Agent B* a réussi. Les détails de cet échange indiquent qu'un REFER particulier intervient en dehors de la session (il n'y pas d'étiquette "Tag" dans le champ "To" de la demande REFER). Si le message REFER intervenait dans la session le "To" serait accompagné d'un "Tag".

Premier message (F1)

```

REFER sip:b@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agenta.atlanta.example.com;branch=z9hG4bK2293940223
To: <sip:b@atlanta.example.com>
From: <sip:a@atlanta.example.com>;tag=193402342
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 93809823 REFER
Max-Forwards: 70
Refer-To: (whatever URI)
Contact: sip:a@atlanta.example.com
Content-Length: 0
  
```

Second message (F2)

```

SIP/2.0 202 Accepted
Via: SIP/2.0/UDP agenta.atlanta.example.com;branch=z9hG4bK2293940223
To: <sip:b@atlanta.example.com>;tag=4992881234
From: <sip:a@atlanta.example.com>;tag=193402342
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 93809823 REFER
Contact: sip:b@atlanta.example.com
Content-Length: 0
  
```

Troisième message (F3)

```

NOTIFY sip:a@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9922ef992-25
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993402 NOTIFY
Max-Forwards: 70
Event: refer
Subscription-State: active;expires=(depends on Refer-To URI)
Contact: sip:b@atlanta.example.com
Content-Type: message/sipfrag;version=2.0
Content-Length: 20

SIP/2.0 100 Trying
  
```

Quatrième message (F4)

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9922ef992-25
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993402 NOTIFY
Contact: sip:a@atlanta.example.com
Content-Length: 0
```

Cinquième message (F5)

```
NOTIFY sip:a@atlanta.example.com SIP/2.0
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9323394234
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993403 NOTIFY
Max-Forwards: 70
Event: refer
Subscription-State: terminated;reason=noresource
Contact: sip:b@atlanta.example.com
Content-Type: message/sipfrag;version=2.0
Content-Length: 16

SIP/2.0 200 OK
```

Sixième message (F6)

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP agentb.atlanta.example.com;branch=z9hG4bK9323394234
To: <sip:a@atlanta.example.com>;tag=193402342
From: <sip:b@atlanta.example.com>;tag=4992881234
Call-ID: 898234234@agenta.atlanta.example.com
CSeq: 1993403 NOTIFY
Contact: sip:a@atlanta.example.com
Content-Length: 0
```

Méthode MESSAGE

La méthode MESSAGE ([RFC 3428](#)) propose des fonctionnalités de messagerie instantanée (IM). Cette méthode transporte des contenus en format MIME.

Méthode INFO

La méthode INFO ([RFC 2976](#)) permet d'embarquer des informations de signalisation DTMF, de taxation, signalisation ISDN, etc.

Méthode PRACK

Alors qu'un message ACK fait suite à tous les codes de réponses sauf 1XX (codes provisionnels), PRACK ([RFC 3262](#)) assure des accusées de réception pour des réponse 180 notamment qui peuvent être critiques.

Méthode UPDATE

<https://tools.ietf.org/html/rfc3311>

Permet de mettre à jour des paramètres de session (codecs) avant qu'elle ne débute.

SIP et la sécurité

- 1. Menaces
 - Rapport Shodan
 - Typologie des attaques et des menaces
- 2. Scans SIP
- 3. Attaque ARP Poison Routing (APR)
 - Capture de paquets
 - Capturer des paquets dans un environnement LAN commuté
 - Transfert de trafic dans une attaque APR
 - Détail de l'attaque
 - Outils d'attaque ARP
 - Contre-mesures
- 4. sipdump/sipcrack john
 - Dump d'un REGISTER (sans attaque MiTM)
 - Crack du mot de passe REGISTER avec un dictionnaire pof.
 - Crack du mot de passe REGISTER avec un dictionnaire standard
 - Crack de l'emprunte avec john
- 5. Metasploit
- 6. Autres outils et Frameworks
 - Frameworks
 - Denial Of Service
- 7. Manipulation de trafic VoIP (forges, générateurs)
 - Outils de forge génériques
 - Outils de forge dédiés SIP :
- 8. Protection d'un infrastructure SIP
 - Propositions du RFC 3261
 - Solution en bordure
 - Solution interne
 - Best Practices SBC Avaya
- 9. Session Border Controller (SBC)
- 10. Signalisation et transport sécurisés
 - Sécurité Astersik
 - Sécurité Kamailio
 - Sécurité FreeSWITCH

Ce chapitre attire l'attention du lecteur sur les menaces qui pèsent sur les technologies VoIP/SIP. Il est nécessaire pour les attaques locales de disposer d'un lab de téléphones et de services de téléphonie. Il est fortement recommandé d'isoler entièrement ses essais et de ne cibler que des ressources prévues à cet effet. Enfin, on terminera le propos avec des propositions pour la mise en oeuvre de la sécurité des ces communications et le rôle d'un périphérique que l'on appelle "Session Border Controller" (SBC).

1. Menaces

Rapport Shodan

- <https://www.shodan.io/report/m3PurUVH> : SIP-CH-09042017, Search for port:"5060" country:"CH" returned 209,597 results on 08-04-2017
- <https://www.shodan.io/report/j1kUIYeF> : SIP-BE-04022108 Search for port:"5060" country:"BE" returned 29,506

results on 04-02-2018

- <https://www.shodan.io/report/d3HjLPLn> : SIP-BE-CHA-04022018 Search for port:"5060" country:"BE" city:"Charleroi" returned 96 results on 04-02-2018
- <https://www.shodan.io/report/VdSMYbFZ> : SIP-BE-FPBX-04022018 Search for port:"5060" country:"BE" product:"FPBX" returned 171 results on 04-02-2018
- <https://www.shodan.io/report/YHAVcxdc> : BE-Asterisk-04022108 Search for country:"BE" product:"Asterisk" returned 349 results on 04-02-2018

Typologie des attaques et des menaces

- Information Gathering, Footprinting and Enumeration
- Monitoring Traffic and eavesdropping Phone calls
- Attacking Authentication
- VLAN Hopping.
- Denial of Service / Flooding.
- Spoofing Caller ID
- Toll fraud

2. Scans SIP

- sipvicious : svmap, svcrack, svwar, svcrash : <http://tools.kali.org/sniffingspoofing/sipvicious>, <https://github.com/EnableSecurity/sipvicious>

```
root@kali:~# svmap 172.16.98.0/24
| SIP Device      | User Agent      | Fingerprint |
|-----|
| 172.16.98.224:5060 | Asterisk PBX 13.12.2 | disabled |
```

- nmap (<https://nmap.org/nsedoc/scripts/sip-methods.html>)

```
nmap --script=sip-methods -sU -p 5060 <targets>
```

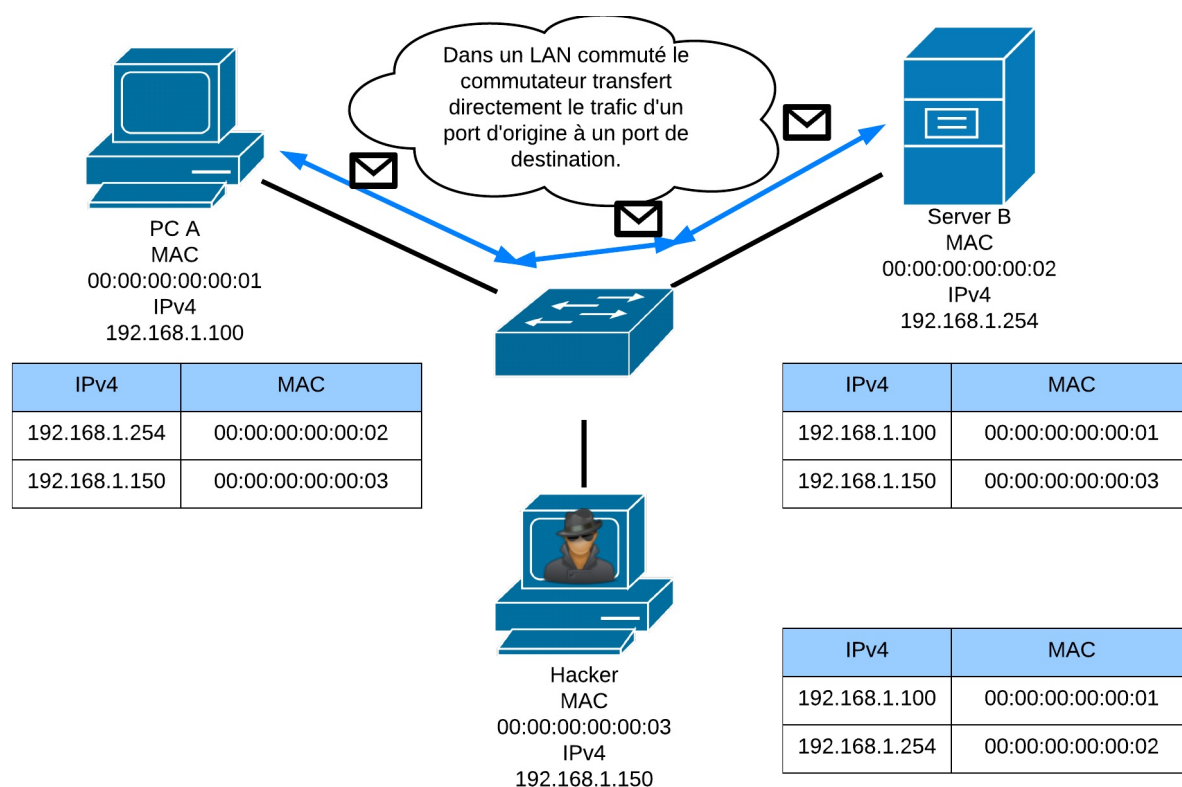
3. Attaque ARP Poison Routing (APR)

Cette attaque permet de s'immiscer dans les communications non protégées au sein d'un réseau local (LAN).

Capture de paquets

Nous avons vu que la capture de paquet ne fournissait des résultats uniquement que pour le trafic livré à l'interface ethernet soit en *unicast* à destination de la machine elle-même, en *broadcast* et en *multicast* transférés d'emblée par les commutateurs.

Dans un environnement LAN commuté, le commutateur transfère directement le trafic en fonction de l'adresse MAC de destination encodée dans les trames Ethernet. Ce sont les hôtes d'origine et de destination qui encodent ces adresses sur base d'un processus ARP.



Capturer des paquets dans un environnement LAN commuté

Si l'on désire capturer tous les paquets du réseau, on placera un port d'un commutateur Cisco en mode "span" soit en mode miroir. Si la pratique est acceptable dans le cadre de la surveillance quotidienne du réseau, elle est moins crédible dans le cadre d'audit d'intrusion.

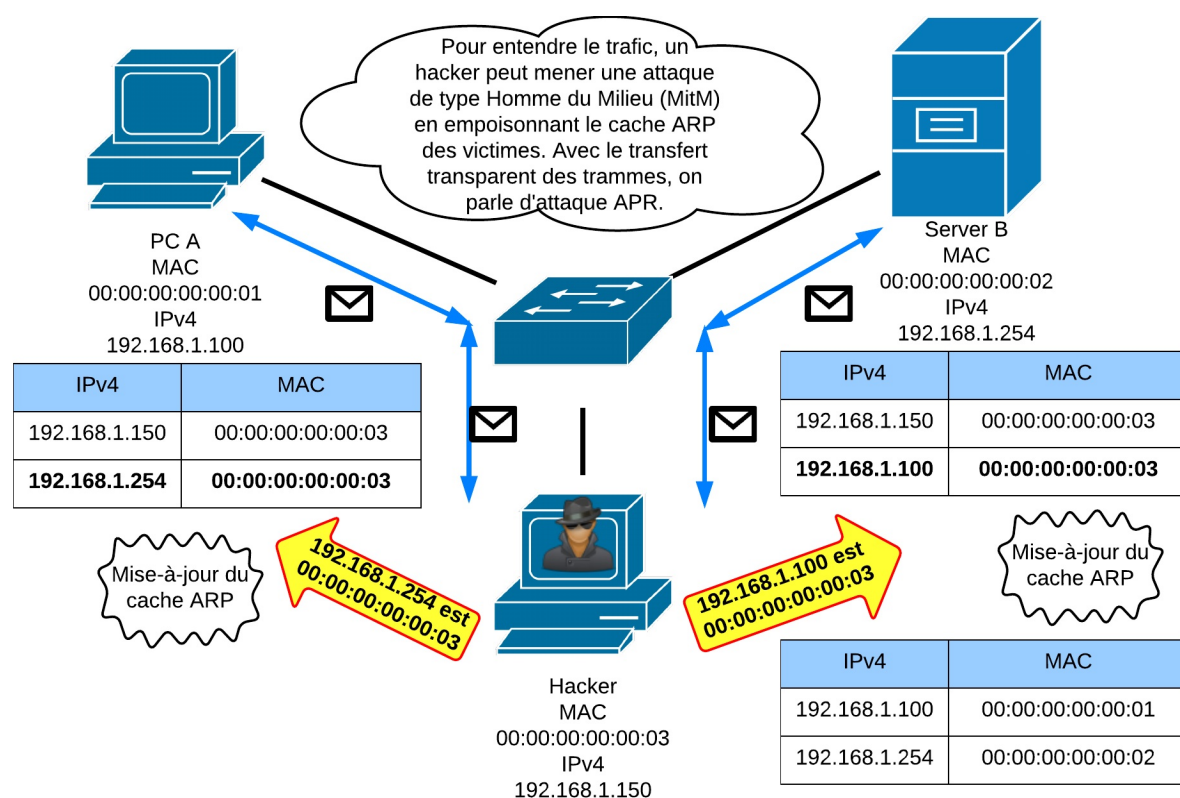
Il pourrait sembler plus simple d'attaquer le commutateur. L'outil `macof` de la suite `dsniff` vise à saturer la table CAM du commutateur avec des entrées factices. L'effet escompté est que le commutateur transfère le trafic par tous ses ports à la manière d'un concentrateur Ethernet (Hub). Toutefois, les commutateurs d'entreprise pourraient supporter cette charge. L'auteur obtiendrait tout au plus un déni de service (DOS) peu discret.

Transfert de trafic dans une attaque APR

Dans un cadre autorisé, on peut s'intéresser à une faiblesse intrinsèque des protocoles TCP/IP sur les réseaux locaux. Elle concerne aussi bien ARP pour IPv4 et Neighbor Discovery pour IPv6. Ici, on s'intéressera uniquement à l'attaque dite APR.

La faiblesse tient au fait que les hôtes TCP/IP ne filtrent pas le trafic ARP. De la même manière aucune authentification de ce trafic n'est aujourd'hui implémentée dans nos réseaux. Les victimes vont accepter du trafic ARP gratuits adressé par le pirate en unicast avec des champs ARP usurpés.

Une attaque APR (ARP Poison Routing) est une attaque d'interception (MiTM) qui consiste pour le pirate à empoisonner le cache ARP des victimes avec sa propre adresse MAC. On peut aussi classer l'attaque comme attaque d'usurpation (spoofing). A condition que le pirate prenne en charge le routage des trames entre les destinations légitimes, la communication ne sera pas interrompue. Le pirate pourra alors observer le trafic entre les victimes de manière transparente. Il est évident que le pirate peut devenir un goulot d'étranglement en fonction du nombre de victimes qu'il usurpe.



L'attaque consiste à empoisonner le cache des victimes avec son adresse MAC en correspondance des adresses IPv4 à usurper et à activer le routage IPv4.

De manière crédule, les victimes vont livrer le trafic au pirate.

Une attaque APR est souvent le préalable d'attaques d'interception (MitM). Elle intervient sur un LAN sur lequel le pirate dispose d'un accès physique tels qu'un réseau Ethernet ou un réseau Wi-Fi ouvert. Si le réseau sans-fil est sécurisé, il sera nécessaire de le casser pour réaliser cette dernière attaque.

Détail de l'attaque

Une analyse préalable comme un scan ARP permet d'identifier les stations (une personnalité de l'organisation, un admin) ou les serveurs victimes (téléphonie, courriel, partages, ...). La passerelle du réseau local est aussi un cible de choix. L'attaque peut trouver son utilité sur des réseaux ouverts comme des hotspots wi-fi.

Par exemple sous Linux avec `arp spoof`, les victimes sont respectivement 192.168.1.100 et 192.168.1.254 :

On installe la suite logicielle `dsniff` et on active le routage IPv4

```
# apt-get install dsniff
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

On empoisonne la table ARP de 192.168.1.100 avec une annonce indiquant l'adresse IPv4 de l'autre victime 192.168.1.254 en relation avec l'adresse MAC de l'interface `eth0` du pirate.

```
# arpspoof -i eth0 -t 192.168.1.100 192.168.1.254
```

Ensuite dans un autre terminal, on empoisonne la table ARP de 192.168.1.254 avec une annonce indiquant l'adresse IPv4 de l'autre victime 192.168.1.100 en relation avec l'adresse MAC de l'interface `eth0` du pirate.

```
# arpspoof -i eth0 -t 192.168.1.254 192.168.1.100
```

Outils d'attaque ARP

On peut réaliser cette attaque sous Linux avec :

- Dsniff (arpspoof)
- Cain sous Windows
- Ethercap
- Packeth
- Scapy
- Yersinia
- ...

Contre-mesures

- Inspection de couche 2
 - DAI (Deep ARP Inspection)
 - DHCP snooping
- Surveillance
 - IDS/IPS
 - arpswatch, arpalert, ndpmon, ntop-ng
- Tuning
 - Désactiver les "ARP gratuits"
 - Des enregistrements statiques ?
- Design réseau
 - Architecture VLAN
 - IEEE 802.1X/Radius/EAP/802.11i
 - HTTPS, SSH, SIPS, ZRTP
 - Infrastructure à clé publique (PKI)

4. sipdump/sipcrack john

Dump d'un REGISTER (sans attaque MiTM)

```
root@kali:~# sipdump -i eth0 test

SIPdump 0.2 ( MaJoMu | www.codito.de )
-----

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

* Dumped login from 172.16.98.224 -> 172.16.98.1 (User: 'telephone1')
* Dumped login from 172.16.98.224 -> 172.16.98.1 (User: 'telephone1')
* Dumped login from 172.16.98.224 -> 172.16.98.1 (User: 'telephone1')
```

Crack du mot de passe REGISTER avec un dictionnaire pof.

```
root@kali:~# echo "1111" > password.lst
root@kali:~# sipcrack -w password.lst test

SIPcrack 0.2 ( MaJoMu | www.codito.de )
-----

* Found Accounts:
```

```

Num      Server      Client      User      Hash|Password

1       172.16.98.1    172.16.98.224    telephone1    aef0698218c9e25f957a6bb7262ff2b4
2       172.16.98.1    172.16.98.224    telephone1    bdebe7e2e51fa2a4a0262ee22c8946c2
3       172.16.98.1    172.16.98.224    telephone1    fd346613dc423e540a1193df1ff02901

* Select which entry to crack (1 - 3): 1

* Generating static MD5 hash... 656b1403125e0d95d6b2c906e49e3d83
* Loaded wordlist: 'password.lst'
* Starting bruteforce against user 'telephone1' (MD5: 'aef0698218c9e25f957a6bb7262ff2b4')
* Tried 1 passwords in 0 seconds

* Found password: '1111'
* Updating dump file 'test'... done

```

Crack du mot de passe REGISTER avec un dictionnaire standard

```

# sipcrack -w /usr/share/john/password.lst test

SIPcrack 0.2 ( MaJoMu | www.codito.de )
-----

* Found Accounts:

Num      Server      Client      User      Hash|Password

1       172.16.98.1    172.16.98.224    telephone1    1111
2       172.16.98.1    172.16.98.224    telephone1    bdebe7e2e51fa2a4a0262ee22c8946c2
3       172.16.98.1    172.16.98.224    telephone1    fd346613dc423e540a1193df1ff02901

* Select which entry to crack (1 - 3): 2

* Generating static MD5 hash... 656b1403125e0d95d6b2c906e49e3d83
* Loaded wordlist: '/usr/share/john/password.lst'
* Starting bruteforce against user 'telephone1' (MD5: 'bdebe7e2e51fa2a4a0262ee22c8946c2')
* Tried 754 passwords in 0 seconds

* Found password: '1111'
* Updating dump file 'test'... done

```

Crack de l'emprunte avec john

```
# john test
```

à vérifier.

5. Metasploit

```

# msfconsole

IIIIII  dTb.dTb      _.-.-.-_
II      4'  v  'B   .'"'. /|\\". "''.
II      6.      .P  :  .' / | \ \ . ' :
II      'T; . .;P'  '.' / | \ \ \ . '
II      'T; ;P'    \ / | | \ \ . '
IIIIII  'YvP'      \-._|_.-'

I love shells --egypt

```

Love leveraging credentials? Check out bruteforcing
in Metasploit Pro -- learn more on <http://rapid7.com/metasploit>

```
=[ metasploit v4.12.22-dev ]
+ -- --[ 1577 exploits - 906 auxiliary - 272 post ]
+ -- --[ 455 payloads - 39 encoders - 8 nops ]
+ -- --[ Free Metasploit Pro trial: http://r-7.co/trymsp ]
```

Recherche d'attaques avec l'occurrence "voip".

```
msf > search voip
```

Matching Modules

=====

Name	Disclosure Date	Rank	Description
auxiliary/voip/asterisk_login		normal	Asterisk Manager Login Utility
auxiliary/voip/cisco_cucdm_call_forward		normal	Viproy CUCDM IP Phone XML Services - Call Forwarding Tool
auxiliary/voip/cisco_cucdm_speed_dials		normal	Viproy CUCDM IP Phone XML Services - Speed Dial Attack Tool
auxiliary/voip/sip_deregister		normal	SIP Deregister Extension
auxiliary/voip/sip_invite_spoof		normal	SIP Invite Spoof
auxiliary/voip/telisca_ips_lock_control	2015-12-17	normal	Telisca IPS Lock Cisco IP Phone Control
post/multi/escalate/allwinner_backdoor	2016-04-30	normal	Allwinner 3.4 Legacy Kernel Local Privilege Escalation

Recherche d'attaques avec le mot clé "sip"

```
msf > search sip
```

Matching Modules

=====

Name	Disclosure Date	Rank	Description
auxiliary/scanner/sip/enumerator		normal	SIP Username Enumerator (UDP)
auxiliary/scanner/sip/enumerator_tcp		normal	SIP Username Enumerator (TCP)
auxiliary/scanner/sip/options		normal	SIP Endpoint Scanner (UDP)
auxiliary/scanner/sip/options_tcp		normal	SIP Endpoint Scanner (TCP)
auxiliary/scanner/sip/sipdroid_ext_enum		normal	SIPDroid Extension Grabber
auxiliary/server/capture/sip		normal	Authentication Capture: SIP
auxiliary/voip/sip_deregister		normal	SIP Deregister Extension
auxiliary/voip/sip_invite_spoof		normal	SIP Invite Spoof
exploit/windows/browser/aol_icq_downloadagent	2006-11-06	excellent	America Online ICQ ActiveX Control Arbitrary File Download and Execute
exploit/windows/local/agnitum_outpost_acs	2013-08-02	excellent	Agnitum Outpost Internet Security Local Privilege Escalation
exploit/windows/sip/aim_triton_cseq	2006-07-10	great	AIM Triton 1.0.4 CSeq Buffer Overflow
exploit/windows/sip/sipxezphone_cseq	2006-07-10	great	SIPfoundry sipXezPhone 0.35a CSeq Field Overflow
exploit/windows/sip/sipxphone_cseq	2006-07-10	great	SIPfoundry sipXphone 2.6.0.27 CSeq Buffer Overflow

Exemple d'attaque.

D'abord activer le module "auxiliary/scanner/sip/options"

```
msf > use auxiliary/scanner/sip/options
```

Ensuite, vérifier les paramètres.

```
msf auxiliary(options) > show options

Module options (auxiliary/scanner/sip/options):

  Name      Current Setting  Required  Description
  ----      -
  BATCHSIZE  256              yes       The number of hosts to probe in each set
  RHOSTS     172.16.98.0/24   yes       The target address range or CIDR identifier
  RPORT      5060             yes       The target port
  THREADS    10              yes       The number of concurrent threads
  TO         nobody           no        The destination username to probe at each hos
```

Ensuite fixer la variable RHOSTS.

```
msf auxiliary(options) > set RHOSTS 172.16.98.0/24
RHOSTS => 172.16.98.0/24
```

Enfin lancer l'attaque.

```
msf auxiliary(options) > run

[*] Sending SIP UDP OPTIONS requests to 172.16.98.0->172.16.98.255 (256 hosts)
[*] 172.16.98.224:5060 udp SIP/2.0 401 Unauthorized: {"Server"=>"Asterisk PBX 13.12.2"}
[*] 172.16.98.224:5060 udp SIP/2.0 401 Unauthorized: {"Server"=>"Asterisk PBX 13.12.2"}
[*] 172.16.98.224:5060 udp SIP/2.0 401 Unauthorized: {"Server"=>"Asterisk PBX 13.12.2"}
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

6. Autres outils et Frameworks

Frameworks

- VIPROY - VoIP Pen-Test Kit for Metasploit Framework : <https://github.com/fozavci/viproxy-voipkit>
- Pentest : <https://github.com/sanvil/vsaudit>
- VLAN Hopping : <http://voiphopper.sourceforge.net/>
- Pentest : <https://github.com/jesusprubio/bluebox-ng>

Denial Of Service

Des plus bêtes et des plus violentes attaques qui ne se tarissent pas.

- Inviteflood
- Rtpflood
- laxflood
- Teardown

7. Manipulation de trafic VoIP (forges, générateurs)

On citera ici quelques outils de référence open-source parmi certainement beaucoup d'autres.

Outils de forge génériques

- [Scapy](#) : voir des exemples sur <https://github.com/unregistered436/scapy> ou sur [SIP OPTIONS with Scapy](#)
- [Ostinato Network Traffic Generator and Analyzer](#)

Outils de forge dédiés SIP :

- [Sipp](#) : *SIPp is a free Open Source test tool / traffic generator for the SIP protocol.*
- [SIPPing](#) : SIPPing is a simple SIP packet forging tool written in pure Python.

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

8. Protection d'un infrastructure SIP

Les éléments fondamentaux de sécurité peuvent être identifiés par l'acronyme *Confidentiality Integrity Authentication*. On peut y ajouter d'autres éléments tels que la non-répudiation, le non-rejeu, etc. mais ces trois Confidentialité, Intégrité et Authentification sont un bon points de départ.

Aussi, il est important d'identifier les zones (leur localisation) soumises aux menaces : l'exposition privée (interne/intra/inter) et publique (des serveurs SIP exposés publiquement).

Certaines menaces demandent des réponses appropriées mais elles sont toujours à évaluer sur base d'une étude de risque.

Enfin, nul n'est bien ou mal intentionné : plus simplement, il n'y aucune raison d'offrir à d'éventuels intrus un occasion d'accéder ou d'endommager les données ou les services que nous gérons.

Si on suit ce point de vue, certes sommaire mais simple quelques mesures par conception (*security by design*) semblent assez évident :

- gestion parcimonieuse des mises-à-jour logicielles (tous périphériques confondus), maintenance, support, ...
- séparation stricte des flux (gestion, voip, data et autres)
- imposer des politiques de mots de passe très fortes (longueur, variabilité, imposé, régularité de modification), voire utiliser des authentification à chiffrement asymétriques
- passer à une infrastructure PKI (Public Key Infrastructure) permettant un déploiement robuste de TLS (pour SIP)
- de tout chiffrer nativement (TLS, SIPS, SRTP, ZRTP, **IAX**) ou à travers des tunnels IPSEC
- de se soucier du filtrage et du comportement sur les interfaces avec des pare-feux (de tout type, voir plus bas) et/ou des systèmes de détection d'intrusion (tel que Snort par exemple)

Propositions du RFC 3261

Le [RFC 3261](#), [section 26](#), énonce une série d'attaques classiques et de contre-mesures dans un cadre où les communications et services SIP sont exposées publiquement sur Internet :

Autres menaces :

- Capture d'enregistrement
- Usurpation du nom d'un serveur
- Altération des corps de message
- Suppression de sessions
- Dénier de service et amplification

Contre-mesures :

- Sécurité des couches Transport (TLS) et Réseau (IPSEC)
- Schéma d'URI SIPS
- Authentification HTTP

- Chiffrement S/MIME

Solution en bordure

Si on ne peut faire autrement que d'exposer SIP sur des adresses publiques, diverses mesures

- Pare-feu
- Pare-feu lié à l'applicatif (de type [fail2ban](#))
- Pare-feu Applicatif
- Un [SBC, Session Border Controller](#)
- Confier le service à un prestataire d'envergure

Solution interne

- Séparation des flux gestion, data, voip, etc. (VLANs)

Best Practices SBC Avaya

[Security Best Practice Checklist](#)

9. Session Border Controller (SBC)

Le [RFC 5853](#) (2010) de type "Informational" intitulé "Requirements from Session Initiation Protocol (SIP) Session Border Control (SBC) Deployments".

Le but du document est de décrire les fonctions implémentées dans les SBC. Une attention particulière est accordée aux pratiques qui sont en conflit d'une manière ou d'une autre avec les principes architecturaux du protocole SIP. Il explore également les exigences sous-jacentes des opérateurs de réseau qui ont conduit à l'utilisation de ces fonctions et pratiques.

Le terme Session Border Controller (SBC) est un terme relativement indéfini dans le sens où il n'est standardisé ni défini nulle part.

Les SBC se situent habituellement entre deux réseaux de fournisseurs de services dans un environnement de "peering" ou entre un réseau d'accès et un réseau de backbone. On est donc très éloigné des moyens et des besoins de petites et moyennes entreprises.

Ils fournissent une variété de fonctions pour activer ou améliorer les services multimédias basés sur des sessions (par exemple, la voix sur IP). Ces fonctions comprennent:

- la défense de périmètre (contrôle d'accès, dissimulation de topologie, prévention et déni de service);
- fonctionnalité non disponible sur les points d'extrémité (NAT Traversal, inter-opérabilité ou réparation de protocole);
- la gestion du trafic (surveillance des médias et qualité de service (QoS)).

Le document "[Market Guide for Enterprise SBC](#)" de Gartner du 29 décembre 2017 sera certainement une aide utile pour évaluer les solutions et les tendances actuelles du marché. En terme de parts de marché, Cisco et Audiocodes augmentent leur parts contre Sonus et Oracle ([Infonetics' quarterly enterprise SBC report, march 2015](#)). Il reste seulement 40 % de parts de marché aux autres acteurs.

On trouvera dans le document de Gartner de informations quant aux offres dont le nombre de connexion simultanées à gérer donnera une fourchette de prix quant à la solution envisagée. Par ailleurs, on notera que Gartner définit précisément l'offre E-SBC avec d'autant plus d'inspiration que le RFC 5853.

Le document [Avaya SBCE 7.0 Security Configuration and Best Practices Guide](#) peut donner une idée des recommandation d'un acteur du marché.

10. Signalisation et transport sécurisés

D'autres sources d'informations plus encyclopédiques ou plus opérationnelles s'évertuent à expliquer les architectures PKI, leurs mécanismes, leurs implications, leurs protocoles et leur déploiement.

Plusieurs éléments sont toutefois à prendre en compte :

- La validité des certificats
- La gestion des certificats
- Le périmètre de déploiement
- La capacité et la configuration des postes terminaux

On peut se référer à quelques guides de déploiement mais on est encore très loin des contraintes pratiques que l'on rencontrera dans la réalité. Il est bien sûr toujours plus facile de partir d'une page blanche ou d'une situation nouvelle technique et culturelle alors que cette réalité nous confronte souvent à des situations hétérogènes difficiles à intégrer. Mais l'heure n'est certainement pas au [renoncement](#) au contraire car ces technologies commencent à se démocratiser.

Deux protocoles sécurisé ont la faveur du marché :

- SIP sur TLS (SIPS) pour la signalisation.
- SRTP pour le transport ([RFC 3711](#)).

Sécurité Astersik

Le [wiki d'Asterisk Secure Calling Tutorial](#) donne une bonne idée des opérations sur un client et un serveur pour déployer TLS en chan_pjsip et chan_sip ainsi qu'en SRTP.

On notera les capacités intéressantes du protocoles IAX de transporter le media et la signalisation nativement dans des tunnels sécurisés en AES. Cette solution open-source traverse particulièrement bien les pare-feux en bordure d'un Internet entre des serveurs protégés.

Sécurité Kamailio

- <https://www.kamailio.org/wiki/tutorials/security/kamailio-security>

Sécurité FreeSWITCH

- <https://freeswitch.org/confluence/display/FREESWITCH/Security>
- <https://freeswitch.org/confluence/display/FREESWITCH/Certificates>
- <https://freeswitch.org/confluence/display/FREESWITCH/Fail2Ban>
- <https://freeswitch.org/confluence/display/FREESWITCH/SIP+TLS>
- <https://freeswitch.org/confluence/display/FREESWITCH/SSL%2C+SSLv2%2C+SSLv3>

RFCs SIP

- [Core SIP Documents](#)
- [SDP-Related Documents](#)
- [RTP-Related Documents](#)
- [HTTP-Related Documents](#)
- [MIME-Related Documents](#)
- [SIP Standards Track Documents \(Options, Extensions, etc.\)](#)
- [SIP Informational RFCs and BCP Documents](#)
- [SIP-Related Documents](#)
- [Directory Services Documents](#)

Liste maintenue par <https://www.packetizer.com/ipmc/sip/standards.html>

Core SIP Documents

RFC Reference	Document Title
RFC 2543	SIP: Session Initiation Protocol (obsolete)
RFC 3261	SIP: Session Initiation Protocol
RFC 3262	Reliability of Provisional Responses
RFC 3263	Locating SIP Servers
RFC 3265	SIP-Specific Event Notification
RFC 5954	Essential Correction for IPv6 ABNF and URI Comparison in RFC 3261

SDP-Related Documents

RFC Reference	Document Title
RFC 2327	Session Description Protocol (SDP) (obsolete: see RFC 4566)
RFC 3264	An Offer/Answer Model with the Session Description Protocol (SDP)
RFC 3266	Support of IPv6 in SDP
RFC 3388	Grouping Media Lines in SDP (obsolete: see RFC 5888)
RFC 3407	Session Description Protocol (SDP) Simple Capability Declaration
RFC 3524	Mapping of Media Streams to Resource Reservation Flows
RFC 3556	SDP Bandwidth Modifiers for RTCP Bandwidth
RFC 3605	Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)
RFC 3890	A Transport Independent Bandwidth Modifier
RFC 4091	An Alternative NAT Semantics for SDP
RFC 4145	TCP-Based Media Transport in the SDP
RFC 4566	Session Description Protocol (SDP)

RFC 4567	Key Management Extensions for SDP and RTSP
RFC 4568	SDP Security Descriptions for Media Streams
RFC 4570	SDP Source Filters
RFC 4572	Connection-Oriented Media Transport over TLS in SDP
RFC 4574	SDP Label Attribute
RFC 4756	FEC Grouping Semantics in SDP
RFC 4796	The SDP Content Attribute
RFC 5027	Security Preconditions for SDP
RFC 5432	QoS Mechanism Selection in SDP
RFC 5547	SDP Offer/Answer Mechanism to Enable File Transfer
RFC 5576	Source-Specific Media Attributes in SDP
RFC 5583	Signaling Media Decoding Dependency in SDP
RFC 5888	The SDP Grouping Framework
RFC 5898	Connectivity Preconditions for SDP Media Streams
RFC 5939	SDP Capability Negotiation
RFC 6064	SDP and RTSP Extensions Defined for 3GPP Packet-Switched Streaming Service and Multimedia Broadcast/Multicast Service (informational)
RFC 6193	Media Description for IKE in SDP
RFC 6164	Media Description for the Internet Key Exchange Protocol (IKE) in SDP (informational)
RFC 6236	Negotiation of Generic Image Attributes in the SDP
RFC 6364	SDP Elements for the Forward Error Correction (FEC) Framework

RTP-Related Documents

RFC Reference	Document Title
RFC 1889	RTP: Transport Protocol for Real-Time Applications (obsolete: see RFC 3550)
RFC 1890	RTP Profile for Audio and Video Conferences with Minimal Control (obsolete: see RFC 3551)
RFC 2198	RTP Payload for Redundant Audio Data
RFC 2733	An RTP Payload Format for Generic Forward Error Correction (obsolete: see RFC 5109)
RFC 2793	RTP Payload for Text Conversation (obsolete: see RFC 4103)
RFC 2833	RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals (obsolete: see RFC 4733)
RFC 2959	Real-Time Transport Protocol Management Information Base
RFC 3389	RTP Payload for Comfort Noise
RFC 3550	RTP: Transport Protocol for Real-Time Applications
RFC 3551	RTP Profile for Audio and Video Conferences with Minimal Control

RFC 3611	RTP Control Protocol Extended Reports (RTCP XR)
RFC 3711	The Secure Real-time Transport Protocol (SRTP)
RFC 3952	RTP Payload Format for internet Low Bit Rate Codec (iLBC) Speech
RFC 4103	RTP Payload for Text Conversation
RFC 4571	Framing RTP and RTCP Packets over Connection-Oriented Transport
RFC 4585	Extended RTP Profile for RTCP-Based Feedback (RTP/AVPF)
RFC 4586	RTP/AVPF: Results of the Timing Rule Simulations
RFC 4588	RTP Retransmission Payload Format
RFC 4629	RTP Payload Format for ITU-T Rec. H.263 Video
RFC 4771	Integrity Transform Carrying Roll-Over for SRTP
RFC 4733	RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals
RFC 4961	Symmetric RTP / RTP Control Protocol
RFC 5109	RTP Payload Format for Generic Forward Error Correction
RFC 5117	RTP Topologies
RFC 5285	A General Mechanism for RTP Header Extensions
RFC 5450	Transmission Time Offsets in RTP Streams
RFC 5506	Support for Reduced-Size RTCP: Opportunities and Consequences
RFC 5761	Multiplexing RTP Data and Control Packets on a Single Port
RFC 6184	RTP Payload Format for H.264 Video
RFC 6642	RTCP Extension for a Third-Party Loss Report
RFC 6679	Explicit Congestion Notification (ECN) for RTP over UDP

HTTP-Related Documents

RFC Reference	Document Title
RFC 2616	Hypertext Transfer Protocol -- HTTP/1.1
RFC 2617	HTTP Authentication: Basic and Digest Access Authentication
RFC 3310	HTTP Digest Authentication Using Authentication and Key Agreement (AKA)

MIME-Related Documents

RFC Reference	Document Title
RFC 1847	Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted
RFC 2045	MIME Part One: Format of Internet Message Bodies
RFC 2046	MIME Part Two: Media Types
RFC 2047	MIME Part Three: Message Header Extensions for Non-ASCII Text
RFC 2048	MIME Part Four: Registration Procedures (obsolete: see RFC 4288 and RFC 4289)

RFC 2633	S/MIME Version 3 Message Specification
RFC 3204	MIME media types for ISUP and QSIG Objects
RFC 3420	Internet Media Type message/sipfrag
RFC 3555	MIME Type Registration of RTP Payload Formats
RFC 4288	Media Type Specifications and Registration Procedures
RFC 4289	Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures

SIP Standards Track Documents (Options, Extensions, etc.)

RFC Reference	Document Title
RFC 2976	The SIP INFO Method (obsolete: see RFC 6086)
RFC 2848	Extensions for IP Access to Telephone Call Services
RFC 3050	CGI for SIP
RFC 3311	UPDATE Method
RFC 3312	Integration of Resource Management and SIP
RFC 3313	Private SIP Extensions for Media Authorization
RFC 3319	DHCPv6 Options for SIP Servers
RFC 3323	A Privacy Mechanism for SIP
RFC 3324	Short Term Requirements for Network Asserted Identity
RFC 3325	Private Extensions to SIP for Asserted Identity within Trusted Networks
RFC 3326	The Reason Header Field
RFC 3327	Extension Header Field for Registering Non-Adjacent Contacts
RFC 3329	Security Mechanism Agreement
RFC 3361	DHCP-for-IPv4 Option for SIP Servers
RFC 3372	SIP for Telephones (SIP-T): Context and Architectures
RFC 3398	ISUP to SIP Mapping
RFC 3428	SIP Extension for Instant Messaging
RFC 3455	Private Header Extensions for 3GPP
RFC 3515	The Session Initiation Protocol (SIP) Refer Method
RFC 3578	Mapping ISUP Overlapped Signalling to SIP
RFC 3581	Extension to SIP for Symmetric Response Routing
RFC 3608	Extension Header Field for Service Route Discovery During Registration
RFC 3680	SIP Event Package for Registrations
RFC 3840	Indicating User Agent Capabilities in SIP
RFC 3841	Caller Preferences for SIP
RFC 3842	Message Summary and Message Waiting Indication Event Package

RFC 3856	Presence Event Package
RFC 3857	A Watcher Information Event Template-Package
RFC 3891	"Replaces" Header
RFC 3892	Referred-By Mechanism
RFC 3893	SIP Authenticated Identity Body (AIB)
RFC 3911	SIP "Join" Header
RFC 3903	Event State Publication
RFC 3959	Early Session Disposition Type
RFC 3960	Early Media and Ringing Tone Generation
RFC 4028	Session Timers in the Session Initiation Protocol (SIP)
RFC 4092	Using SDP Alternative NAT Semantics in SIP (obsolete: see RFC 5245)
RFC 4235	An INVITE-Initiated Dialog Event Package for SIP
RFC 4244	Extension for Request History Information
RFC 4320	Actions Addressing Identified Issues with the SIP Non-INVITE Transaction
RFC 4411	Extending the SIP Reason Header for Preemption Events
RFC 4412	Communications Resource Priority for SIP
RFC 4474	Enhancements for Authenticated Identity Management in SIP
RFC 4479	A Data Model for Presence
RFC 4483	A Mechanism for Content Indirection in SIP
RFC 4488	Suppression of SIP REFER Method Implicit Subscription
RFC 4508	Conveying Feature Tags with the SIP REFER Method
RFC 4575	SIP Event Package for Conference State
RFC 4662	SIP Event Notification Extension for Resource Lists
RFC 4730	Event Package for KPML
RFC 4780	MIB for SIP
RFC 4904	Representing Trunk Groups in tel/sip URIs
RFC 4916	Connected Identity in SIP
RFC 4967	Dial String Parameter for the SIP URI
RFC 4975	Message Session Relay Protocol (MSRP)
RFC 4976	Relay Extension for MSRP
RFC 5079	Rejecting Anonymous Requests in SIP
RFC 5196	SIP User Agent Capability Extension to Presence Information Data Format (PIDF)
RFC 5239	A Framework for Centralized Conferencing
RFC 5245	ICE: A Protocol for NAT Traversal for Offer/Answer Protocols
RFC 5263	SIP Extension for Partial Notification of Presence Information
RFC 5264	Publication of Partial Presence Information

RFC 5362	SIP Pending Additions Event Package
RFC 5363	Framework and Security Considerations for SIP URI-List Services
RFC 5365	Multiple-Recipient MESSAGE Requests in SIP
RFC 5366	Conference Establishment Using Request-Contained Lists in SIP
RFC 5367	Subscriptions to Request-Contained Resource Lists in SIP
RFC 5368	Referring to Multiple Resources in SIP
RFC 5370	SIP Conference Bridge Transcoding Model
RFC 5373	Requesting Answering Modes for SIP
RFC 5393	Addressing an Amplification Vulnerability in SIP Forking Proxies
RFC 5478	IANA Registration of new SIP Resource-Priority Namespaces
RFC 5509	IANA Registration Instant Messaging and Presence DNS SRV RRs for SIP
RFC 5552	SIP Interface to VoiceXML Media Services
RFC 5589	SIP Call Control - Transfer
RFC 5621	Message Body Handling in SIP
RFC 5626	Managing Client-Initiated Connections in SIP
RFC 5627	Obtaining and Using Globally Routable User Agent URIs (GRUUs) in SIP
RFC 5628	Registration Event Package Extension for SIP GRUUs
RFC 5629	A Framework for Application Interaction in SIP
RFC 5630	The Use of the SIPS URI Scheme in SIP
RFC 5631	SIP Session Mobility
RFC 5658	Addressing Record-Route Issues in SIP
RFC 5688	A SIP Media Feature Tag for MIME Application Subtypes
RFC 5768	Indicating Support for Interactive Connectivity Establishment (ICE) in SIP
RFC 5839	An Extension to SIP Events for Conditional Event Notification
RFC 5875	An XML Configuration Access Protocol (XCAP) Diff Event Package
RFC 5922	Domain Certificates in SIP
RFC 5923	Connection Reuse in SIP
RFC 6026	Correct Transaction Handling for 2xx Responses to SIP INVITE Requests
RFC 6035	SIP Event Package for Voice Quality Reporting
RFC 6072	Certificate Management Service for SIP
RFC 6076	Basic Telephony SIP End-to-End Performance Metrics
RFC 6080	A Framework for SIP User Agent Profile Delivery
RFC 6086	SIP INFO Method and Package Framework
RFC 6140	Registration for Multiple Phone Numbers in SIP
RFC 6141	Re-INVITE and Target-Refresh Request Handling in SIP
RFC 6157	IPv6 Transition in SIP

RFC 6223	Indication of Support for Keep-Alive
RFC 6228	SIP Response Code for Indication of Terminated Dialog
RFC 6230	Media Control Channel Framework
RFC 6432	Carrying Q.850 Codes in Reason Header Fields in SIP
RFC 6442	Location Conveyance for SIP
RFC 6446	SIP Event Notification Extension for Notification Rate Control
RFC 6447	Filtering Location Notifications in SIP
RFC 6665	SIP-Specific Event Notification

SIP Informational RFCs and BCP Documents

RFC Reference	Document Title
RFC 3087	Control of Service Context using SIP Request-URI
RFC 3351	User Requirements for SIP in Support of Speech/Hearing Impaired
RFC 3487	Requirements for Resource Priority Mechanisms for SIP
RFC 3603	Private SIP Proxy-to-Proxy Extensions for PacketCable Distributed Call Signaling (obsolete: see RFC 5503)
RFC 3665	SIP Basic Call Flow Examples
RFC 3666	SIP Public Switched Telephone Network (PSTN) Call Flows
RFC 3702	Authentication, Authorization, and Accounting Requirements for SIP
RFC 3824	Using E.164 numbers with SIP
RFC 3968	IANA Registry for SIP Header Field
RFC 3969	IANA Registry for SIP URI
RFC 3976	Interworking SIP and IN Applications
RFC 4117	Transcoding Services Invocation using 3PCC
RFC 4123	SIP-H.323 Interworking Requirements
RFC 4168	SCTP as a transport for SIP
RFC 4189	Requirements for End-to-Middle Security for SIP
RFC 4240	Basic Network Media Services with SIP
RFC 4245	High-Level Requirements for Tightly Coupled SIP Conferencing
RFC 4317	SDP Offer/Answer Examples
RFC 4321	Problems Identified Associated with the SIP Non-INVITE Transaction
RFC 4353	A Framework for Conferencing with SIP
RFC 4354	SIP Event Package and Data Format for Push-to-Talk over Cellular (PoC) Service
RFC 4453	Requirements for Consent-Based Communications in the SIP
RFC 4457	SIP P-User-Database Private-Header (P-Header)

RFC 4458	SIP URIs for Applications such as Voicemail and Interactive Voice Response (IVR)
RFC 4475	SIP Torture Test Messages
RFC 4484	Trait-Based Authorization Requirements for SIP
RFC 4485	Guidelines for Authors of Extensions to SIP
RFC 4504	SIP Telephony Device Requirements and Configuration
RFC 4538	Request Authorization through Dialog Identification in SIP
RFC 4596	Guidelines for Usage of the SIP Caller Preferences Extension
RFC 4579	SIP Call Control - Conferencing for User Agents
RFC 4964	The P-Answer-State Header Extension to SIP
RFC 5002	SIP P-Profile-Key Private Header (P-Header)
RFC 5009	Private Header (P-Header) Extension to SIP for Authorization of Early Media
RFC 5039	SIP and Spam
RFC 5057	Multiple Dialog Usages in SIP
RFC 5118	SIP Torture Test Messages for IPv6
RFC 5194	Framework for Real-Time Text using SIP
RFC 5359	SIP Service Examples
RFC 5390	Requirements for Management of Overload in SIP
RFC 5407	Example Call Flows of Race Conditions in SIP
RFC 5411	A Hitchhiker's Guide to SIP
RFC 5479	Requirements and Analysis of Media Security Management Protocols
RFC 5502	SIP P-Served-User Private-Header (P-Header) for the 3GPP Core Network
RFC 5503	Private SIP Proxy-to-Proxy Extensions for Supporting the PacketCable Distributed Call Signaling Architecture
RFC 5318	The SIP P-Refused-URI-List Private-Header (P-Header)
RFC 5369	Framework for Transcoding with SIP
RFC 5606	Implications of 'retransmission-allowed' for SIP Location Conveyance
RFC 5638	Simple SIP Usage Scenario for Applications in the Endpoints
RFC 5767	User-Agent-Driven Privacy Mechanism for SIP
RFC 5850	A Call Control and Multi-Party Usage Framework for SIP
RFC 5853	Requirements for SIP Session Border Control (SBC) Deployments
RFC 5876	Updates to Asserted Identity in SIP
RFC 5897	Identification of Communications Services in SIP
RFC 5947	Requirements for Multiple Address of Record (AOR) Reachability Information in SIP
RFC 5989	A SIP Event Package for Subscribing to Changes to an HTTP Resource
RFC 6011	SIP User Agent Configuration
RFC 6044	Mapping and Interworking of Diversion Information between Diversion and History-Info Headers in SIP

RFC 6050	A SIP Extension for the Identification of Services
RFC 6216	Example Call Flows Using SIP Security Mechanisms
RFC 6271	Requirements for SIP-Based Session Peering
RFC 6314	NAT Traversal Practices for Client-Server SIP
RFC 6337	SIP Usage of the Offer/Answer Model
RFC 6341	Use Cases and Requirements for SIP-Based Media Recording (SIPREC)
RFC 6357	Design Considerations for SIP Overload Control
RFC 6404	Session PEERing for Multimedia INTERconnect (SPEERMINT) Security Threats and Suggested Countermeasures
RFC 6405	Voice over IP (VoIP) SIP Peering Use Cases
RFC 6406	Session PEERing for Multimedia INTERconnect (SPEERMINT) Architecture
RFC 6567	Problem Statement and Requirements for Transporting User-to-User Call Control Information in SIP

SIP-Related Documents

RFC Reference	Document Title
RFC 3219	Telephony Routing over IP (TRIP) (tutorial)
RFC 3320	Signalling Compression
RFC 3321	Signalling Compression - Extended Operations
RFC 3322	Signalling Compression - Requirements and Assumptions
RFC 3486	Compressing the Session Initiation Protocol (SIP)
RFC 3485	SIP and SDP Static Dictionary for Signaling Compression
RFC 3725	Best Current Practices for 3PCC in SIP
RFC 3761	The E.164 to URI Dynamic Delegation Discovery System Application (ENUM)
RFC 3764	enumservice registration for SIP Addresses-of-Record
RFC 3966	The tel URI for Telephone Numbers
RFC 4077	A Negative Acknowledgement Mechanism for Signaling Compression
RFC 4083	3GPP Release 5 Requirements on SIP
RFC 4465	Signaling Compression (SigComp) Torture Tests
RFC 4497	Interworking between the SIP and QSIG
RFC 4715	The ISDN Subaddress Encoding Type for tel URI
RFC 4740	Diameter SIP Application
RFC 4795	The ENUM Dip Indicator Parameter for the "tel" URI
RFC 5049	Applying Signaling Compression to SIP
RFC 5112	The Presence-Specific Static Dictionary for Signaling Compression
RFC 5115	TRIP attribute for Resource Priority

RFC 5341	The IANA tel URI Parameter Registry
RFC 5806	Diversion Indication in SIP
RFC 5924	Extended Key Usage (EKU) for SIP X.509 Certificates

Directory Services Documents

Standard	Document Title
H.350	Directory Services Architecture for Multimedia Conferencing
H.350.4	Directory Services Architecture for SIP

Programme de formation Asterisk

On trouvera dans ce chapitre une suite d'exercices de mises en oeuvre avec le logiciel Asterisk.

En voici le sommaire :

- [1. Solution FreePBX](#)
- [2. Asterisk Core PABX](#)
- [3. Asterisk Base](#)
- [4. Asterisk Intermédiaire](#)
- [5. Asterisk Avancé](#)

Solution FreePBX

- 1. Introduction
 - 1.1. Fonctionnalités
 - 1.2. Topologie
- 2. Installation
 - 2.1. Procédure d'installation
 - 2.2 Post-installation
 - 2.3. Configuration du PBX
 - 2.4. Configuration des modules
- 3. Connectivité
 - 3.1. Ajout des extensions
 - 3.2. Configuration du compte Anveo
 - 3.3. Configuration du Trunk SIP
 - 3.4. Route sortante
 - 3.5. Route entrante
- 5. Francisation
- 6. Boîtes vocales
- 7. IVR
- 8. Trunk IAX2 intersites
- 9. Sécurité
- 10. Fail2ban
- 11. Support du Fax

On propose ici un exercice d'installation et de configuration d'un IP-PBX avec Astersik/FreePBX dans le but d'illustrer le déploiement VoIP dans une PME. Plus il y a d'utilisateurs et de périphériques, plus le lab est représentatif.

1. Introduction

Source : <https://en.wikipedia.org/wiki/FreePBX>

FreePBX est un GUI basé Web qui gère le serveur de téléphonie Asterisk. FreePBX est sous licence GNU General Public License version 3.

FreePBX a été acquis par Schmooze.com au début 2013 qui a été acquis par Sangoma Technologies Corporation au début 2015.

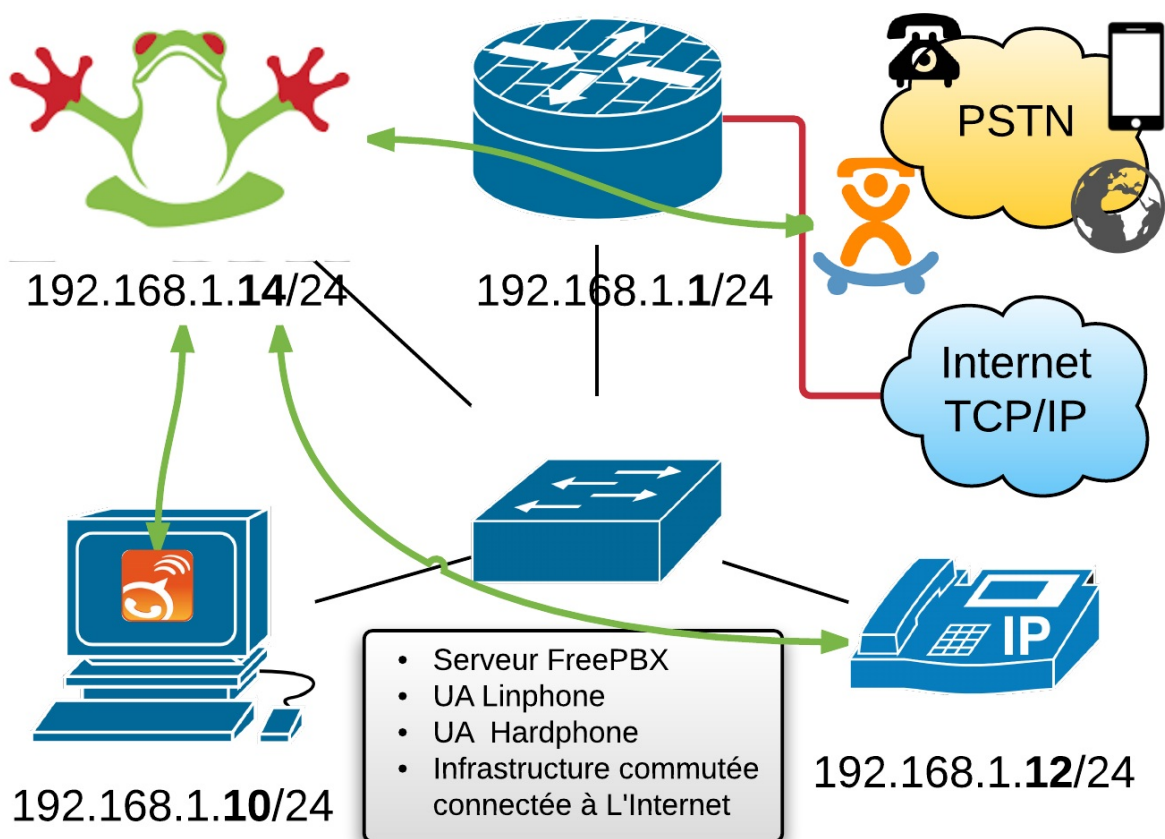
FreePBX peut être exploité :

- A partir d'une installation native combiné au logiciel Asterisk et à une base donnée.
- A partir d'une distribution FreePBX (FreePBX Distro), il existe des variantes comme AsteriskNow, Elastix ou Trixbox.
- A partir d'un matériel dédié et soutenu par les auteurs du projet, en proposant des appliances : <https://www.freepbx.org/store/freepbx-appliances/>, ce qui donne une idée du dimensionnement physique.

1.1. Fonctionnalités

PBX IP graphique complet et à usage avancé.

1.2. Topologie



- Serveur FreePBX/AsteriskNow (2-4Go RAM, 1-2 CPU)
- UA Linphone
- UA Hardphone
- Infrastructure commutée connectée à L'Internet

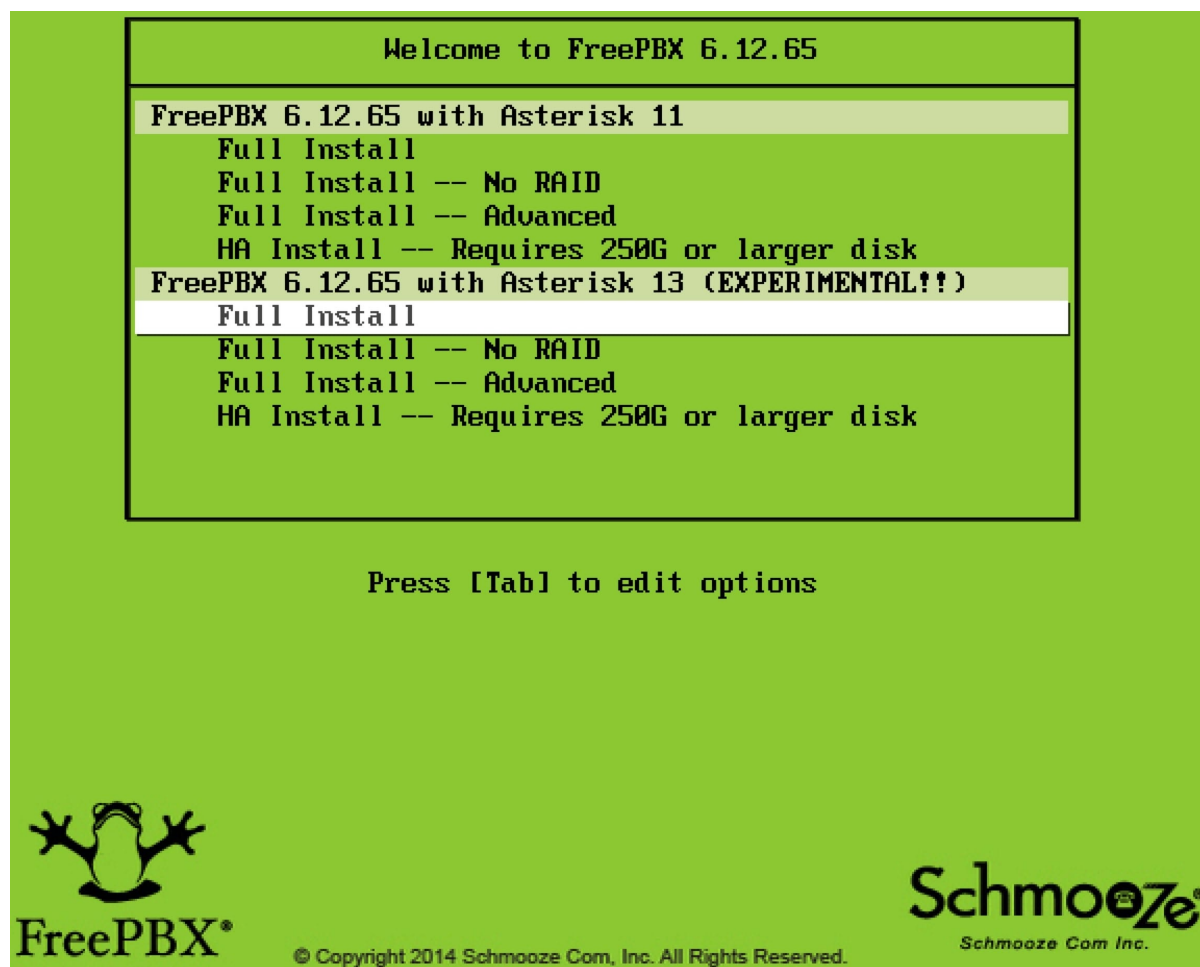
2. Installation

2.1. Procédure d'installation

Source : <https://wiki.asterisk.org/wiki/display/AST/Installing+AsteriskNOW>

Télécharger le fichier `AsteriskNOW-612-current-64.iso` sur <http://www.asterisk.org/downloads/asterisknow>.

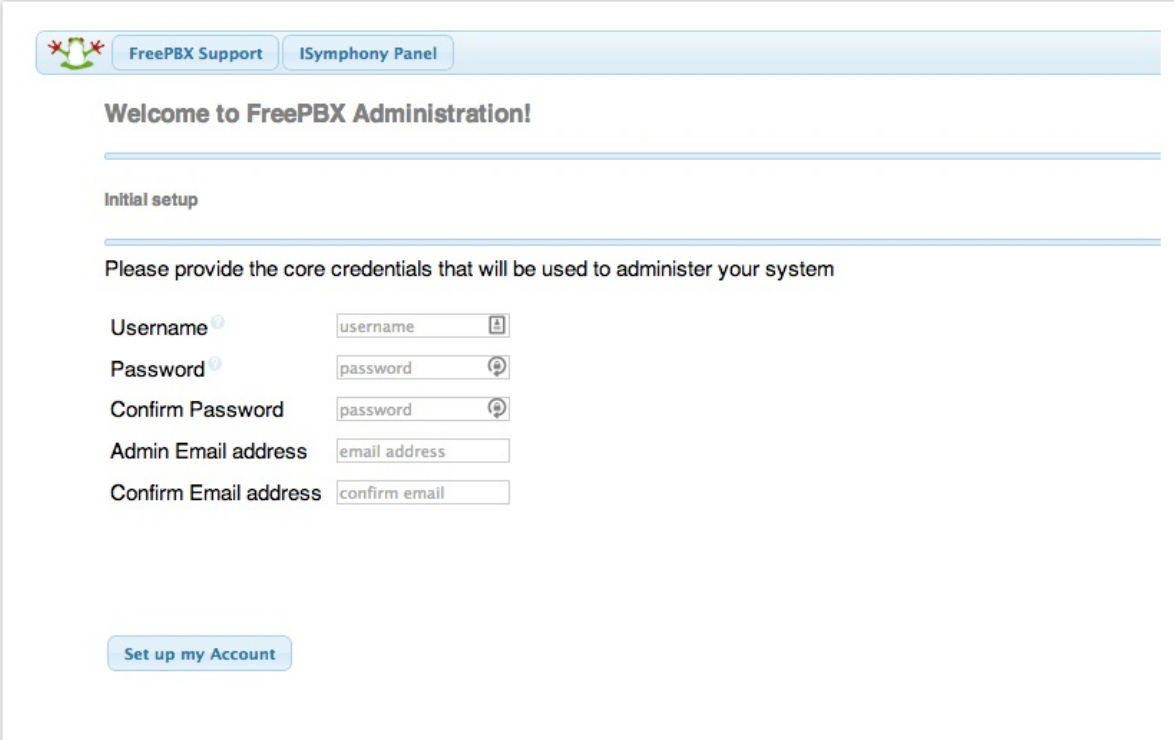
Le choix se porte aujourd'hui sur une installation standard de FreePBX 6.12.65 avec Asterisk 13.



2.2 Post-installation

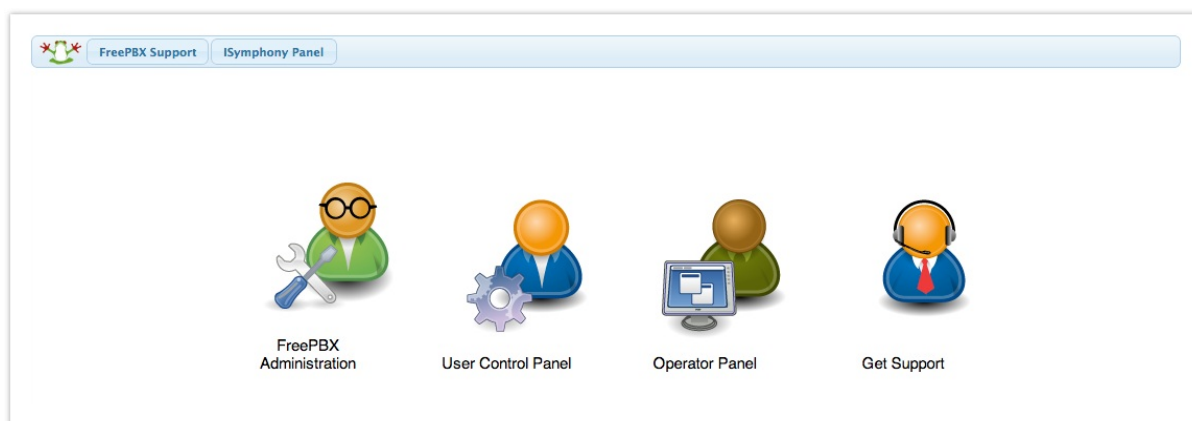
Après redémarrage,

- Prendre connaissance de l'adresse IP du serveur.
- Sur un autre ordinateur, aller à l'adresse http du serveur PBX.
- Remplir le formulaire de création de compte admin



The screenshot shows the 'Initial setup' page of the FreePBX Administration interface. At the top, there is a navigation bar with 'FreePBX Support' and 'iSymphony Panel' links. Below this, a heading reads 'Welcome to FreePBX Administration!'. The main section is titled 'Initial setup' and contains the instruction: 'Please provide the core credentials that will be used to administer your system'. There are five input fields: 'Username' (placeholder 'username'), 'Password' (placeholder 'password'), 'Confirm Password' (placeholder 'password'), 'Admin Email address' (placeholder 'email address'), and 'Confirm Email address' (placeholder 'confirm email'). Each field has a small icon to its right. At the bottom left, there is a blue button labeled 'Set up my Account'.

Le menu principal offre quatre options



1- FreePBX Administration

2- User Control Panel

3- Operator Panel

4- Get Support

- Choisir le menu **FreePBX Administration**
- Passer les étapes d'activation, firewall et SIPStation
- S'attarder sur les offres

2.3. Configuration du PBX

<http://wiki.freepbx.org/display/PPS/FreePBX+Distro+First+Steps+After+Installation>

2.4. Configuration des modules

<http://wiki.freepbx.org/display/FPG/Standard+Modules>

1. ~~Mise à jour des modules~~
2. Installation de nouveaux modules :
3. ~~Languages (voir plus haut)~~
4. Endpoint Manager
5. Time Conditions
6. ~~Ring Groups~~

Paramètres avancés et paramètres SIP

<http://wiki.freepbx.org/display/FPG/Advanced+Settings>

- Adresse : Static
- Adresse IP publique
- Réseau local

<http://wiki.freepbx.org/display/FPG/Asterisk+SIP+Settings> et

<http://wiki.freepbx.org/display/FPG/NAT+Configuration+FreePBX+12>

Changer éventuellement "Strong Passwords"

3. Connectivité

3.1. Ajout des extensions

Softphones SIP/IAX

- Zoiper
- Linphone
- Ekiga
- Jitsi
- Yate
- 3CX Phone
- SJPhone
- Voir https://fr.wikipedia.org/wiki/Liste_des_logiciels_SIP#Clients_SIP

Matériel VoIP

- Polycom SoundPoint IP 321 (3)
- Cisco SPA508G (3)
- Cisco SPA504G (2)
- Cisco SPA922 (6)
- ATA Cisco PAP2T (2)
- Power and Network Devices
- PoE FS108P (4)
- Cisco PoE SF300-24P (1)

Nomenclature des numéros internes

Nom de serveur	Extensions
pbx01	2101, 2102, 2103, 2104
pbx02	2201, 2202, 2203, 2204

pbx03	2301, 2302, 2303, 2304
pbx04	2401, 2402, 2403, 2404
pbx05	2501, 2502, 2503, 2504
pbx06	2601, 2602, 2603, 2604
pbx07	2701, 2702, 2703, 2704
pbx08	2801, 2802, 2803, 2804
pbx09	2901, 2902, 2903, 2904
pbx0a	3001, 3002, 3003, 3004
pbx0b	3101, 3102, 3103, 3104
pbx0c	3201, 3202, 3203, 3204
pbx0d	3301, 3302, 3303, 3304

<http://wiki.freepbx.org/display/F2/Extensions+Module+-+SIP+Extension>

- Application / Extension / Add a SIP Extension
- User Extension
- Display Name
- Devices Options / Secret
- Voicemail : Enabled ...

3.2. Configuration du compte Anveo

Numéros de téléphone en format 322XXXXXXX

https://www.anveo.com/faq.asp?code=sip_freepbx

- Localized
- 32 Belgium
- 00

3.3. Configuration du Trunk SIP

1. Menu Connectivity / Trunks / Add A SIP Trunk
2. Trunk name : Anveo
3. Outbound CallerID : 32XXXXXXX
4. Trunk name : Anveo
5. Outgoing Settings -> PEER Details, remplacer par le numéro de compte et le mot de passe :

```
type=friend
host=sip.de.anveo.com
port=5010
username=ACCOUNT_NUMBER
secret=SIP_PASSWORD
insecure=port,invite
disallow=all
allow=ulaw
context=from-trunk
```

6. Register String : `ACCOUNT_NUMBER:SIP_PASSWORD@sip.de.anveo.com:5010`
7. Submit

3.4. Route sortante

Pour les numéros fixes et cellulaires :

1. Connectivity / Outbound Routes
2. Route Name : Anveo
3. Dial patterns :

```
0XXXXXXX
```

```
0XXXXXXXX
```

- Trunk Sequence for Matched Routes : Anveo
- Submit

3.5. Route entrante

Par numéro de téléphone routé sur le PBX :

1. Connectivity / Inbound Routes
2. DID Number : 32XXXXXXX
3. Set Destinations : p. ex. Extensions/numéro
4. Submit

5. Francisation

A partir de la version 13, de FreePBX, un module se charge des fichiers de langues nécessaires.

Version FreePBX 13 minimum.

1. Admin/Sound Languages
2. Filtrer selon "French"
3. Ajouter les packs de langues FR (alaw)
4. Aller dans "View Custom Languages"
5. Action/edit sur "language fr"
6. Ajouter "French" dans la description et valider
7. Revenir dans Admin/Sound Languages
8. Aller dans change "Global Sound Language"
9. Choisir French et valider

Avec les versions précédentes, 12 et inférieures, il sera nécessaire de placer soi-même les fichiers de langues FR au bon endroit.

On devrait trouver au minimum dans le dossier `/var/lib/asterisk/sounds/fr` le contenu décompressé des fichiers suivants :

- <http://downloads.asterisk.org/pub/telephony/sounds/asterisk-core-sounds-fr-alaw-current.tar.gz>
- <http://downloads.asterisk.org/pub/telephony/sounds/asterisk-extra-sounds-fr-alaw-current.tar.gz>
- <http://downloads.asterisk.org/pub/telephony/sounds/asterisk-moh-opsound-alaw-current.tar.gz>

Plusieurs méthodes sont envisageables pour placer ces fichiers sur le PBX. Le plus facile est de se connecter sur la console du PBX et d'exécuter le script disponible à cette adresse

<https://gist.github.com/goffinet/7835fb38aa1ce29544acc079ecfdb6d>.

Soit, on peut l'exécuter directement dans la console Linux :

```
# wget -q0- https://gist.githubusercontent.com/goffinet/7835fb38aa1ce29544acc079ecfdbed/raw/55513cb8ac796ee8ab  
b3ab133d3b397c619a4f31/astpbx_fr.sh | sh
```

Source du script :

```
#!/bin/sh  
##1. Dans le GUI :  
##Modules Administration/Download and install Languages/Apply Config  
##Asterisk SIP Settings/Advanced General Settings/Language = fr/Apply config  
##2. En console :  
##amportal restart  
##3. Téléchargement des fichiers  
REP=http://downloads.asterisk.org/pub/telephony/sounds  
DIR=/var/lib/asterisk/sounds/fr  
mkdir $DIR  
cd $DIR  
wget $REP/asterisk-core-sounds-fr-alaw-current.tar.gz  
wget $REP/asterisk-extra-sounds-fr-alaw-current.tar.gz  
wget $REP/asterisk-moh-opsound-alaw-current.tar.gz  
gunzip *.gz  
tar xfv asterisk-core-sounds-fr-alaw-current.tar  
tar xfv asterisk-extra-sounds-fr-alaw-current.tar  
tar xfv asterisk-moh-opsound-alaw-current.tar  
rm asterisk-*
```

Ensuite, on pourra définir la langue par défaut ou la langue des utilisateurs

- Modules Administration/Download and install Languages/Apply Config
- Asterisk SIP Settings/Advanced General Settings/Language = fr/Apply

6. Boîtes vocales

Au minimum, on se souciera de la livraison du courrier électronique (messages vocaux).

Sous Debian/Ubuntu :

```
# apt-get remove exim4  
# apt-get install postfix
```

Choisir installation satellite ou Internet smarthost : serveur SMTP par exemple, **smtp.colt.net**, **relay.skynet.be**

Par exemple sous Centos 6 (AsteriskNow), une installation de type smarthost (relay SMTP) :

```
# echo "relayhost = smtp.colt.net" >> /etc/postfix/main.cf  
# service postfix reload
```

On ira consulter la documentation officielle pour la configuration des boîtes vocales.

<http://wiki.freepbx.org/display/F2/Voicemail>

7. IVR

Admin / Feature Codes

<http://wiki.freepbx.org/display/F2/System+Recordings+Module>

<http://wiki.freepbx.org/display/F2/IVR+Module>

8. Trunk IAX2 intersites

Inter-Asterisk eXchange (IAX) est un protocole de couche application natif au logiciel PBX. Il est supporté par quelques autres produits de type systèmes PBX ou téléphones orientés OSS. Le protocole existe aujourd'hui dans sa version 2 et a été formalisé en 2010 dans le [RFC 5456](#) de type *informational (non-standards-track)*. Il remplit aussi bien les fonctions de signalisation que de transport des medias (voix/vidéo). Il utilise un seul port UDP 4569.

IAX remplit les objectifs suivants :

- Réduction de la bande passante utilisée sur les liaisons inter-sites
- Chiffrement AES-128 inclus
- Traversée aisée des routeurs NAT et pare-feux

La documentation ci-dessous suggère une solution fonctionnelle à travers les pare-feux de connexion entre deux PBX distants. Les adresses IP publiques des systèmes téléphoniques doivent être connus.

<http://wiki.freepbx.org/pages/viewpage.action?pageId=4161588>

9. Sécurité

Activité sous Windows

<http://www.oxid.it/cain.html>

1. Capture et crack de mot de passe SIP.
2. Mise en place d'une Attaque MitM ARP Poison Routing entre un PBX et deux téléphones.
3. Capture et reconstitution d'un appel.

10. Fail2ban

Dans AsteriskNow, on peut constater que prison "pbx-gui" vérifie les logs de sécurité d'Asterisk (/var/log/asterisk/freepbx_security.log)

```
# fail2ban-client status
Status
|- Number of jail:      7
`- Jail list:          apache-tcpwrapper, recidive, pbx-gui, apache-badbots, ssh-iptables, asterisk-iptables, vsf
                        tpd-iptables
```

11. Support du Fax

...

Asterisk Core PABX

- 1. Qu'est-ce qu'Asterisk ?
 - 1.1. Asterisk est ...
 - 1.2. Un logiciel Open-Source sur plateformes PC
 - 1.3. Un PBX fonctionnel et plus
 - 1.4. Fonctionnalités avancées
 - 1.5. Société Digium
- 2. Pourquoi Asterisk ?
 - 2.1. Avantages d'Asterisk
 - 2.2. Un coût fortement réduit
 - 2.3. Contrôle et indépendance du système téléphonique
 - 2.4. Environnement de développement rapide et facile
 - 2.5. Riche en fonctionnalités
 - 2.6. Un dial plan flexible et puissant
 - 2.7. Un logiciel open-source testé et maintenu
 - 2.8. Ce qu'en pense l'industrie
- 3. Réticences majeures à l'égard d'Asterisk
 - 3.1. Les parts de marché d'Asterisk sont infimes
 - 3.2. Asterisk gratuit, comment les concepteur en vivent ?
 - 3.3. Difficile de trouver du support
 - 3.4. Asterisk supporte plus de 200 extensions ?
 - 3.5. Seul les « geeks » sont capables de l'installer ?
 - 3.6. Et si le serveur tombe ?
 - 3.7. On n'est pas favorable au logiciel open-source
 - 3.8. Il n'est pas recommandé d'utiliser le CPU pour un PBX
- 4. Documentation Asterisk
 - 4.1. Vidéos YouTube
 - 4.2. Compréhension globale
 - Acronymes et Terminologie
 - Protocoles de téléphonie
 - Linux & Unix
 - Installer et Configurer Asterisk
 - Programmation C
 - 4.3. Où obtenir de l'aide ?
 - Listes de diffusion et IRC
 - Forums Web de discussion
 - Communautés en ligne
 - 4.4. Eviter des informations obsolètes ou incorrectes

1. Qu'est-ce qu'Asterisk ?

1.1. Asterisk est ...

- un projet de développement d'un logiciel [Open Source](#)
- écrit dans le [Le langage de programmation C](#)
- fonctionnant sous [Linux](#) (ou d'autres types de [Unix](#))
- mettant en oeuvre un [Business Telephone Systems](#), soit un [Autocommutateur téléphonique privé](#), soit un PABX.

- connectant différents protocoles de [Telephonie](#)
- c'est aussi une boîte à outils pour fabriquer beaucoup de choses comme
 - un [PABX IP](#) avec beaucoup de fonctionnalités et d'[applications](#)
 - une passerelle VoIP vers des systèmes legacy ([VoIP Gateways](#))
 - un système de conférence ([Conferencing systems](#))
 - un IVR (Interactive Voice Response)
 - un service de file d'attente et/ou de distribution des appels
 - et beaucoup plus
- supportant aussi bien des téléphones VoIP, que [PSTN](#) ou [POTS](#)
- parlant le protocole [SIP](#), le protocole [VoIP](#) le plus répandu, parmi d'autres.

1.2. Un logiciel Open-Source sur plateformes PC

Asterisk est un logiciel PBX open-source qui, une fois installé sur un PC avec les interfaces adéquates, devient un PBX complet

- pour l'utilisateur final à la maison,
- en entreprise,
- chez les fournisseurs de service VoIP,
- pour les sociétés de téléphonie, etc.

Asterisk est à la fois un produit supporté par une communauté open-source et un produit commercial de la société Digium™. Vous êtes libre de modifier le logiciel selon vos besoins.

1.3. Un PBX fonctionnel et plus

Asterisk permet d'établir une connectivité entre des réseaux PSTN et VoIP. Depuis qu'Asterisk est plus qu'un simple PBX, on peut l'utiliser pour de nouveaux usages en téléphonie :

- Connecter à travers Internet les employés qui travaillent de la maison au PBX de l'entreprise; connecter des bureaux distants à travers des réseaux IP;
- Organiser des conférences vocales ou vidéo ;
- donner aux employés d'une entreprise un accès à une boîte vocale intégrée au web et aux courriels;
- Construire des applications de type IVR;
- Donner aux utilisateurs mobiles de l'entreprise un accès au PBX de n'importe où à partir d'une connexion Internet;
- De profiter des meilleurs coûts vers les passerelles opérateurs ;
- D'avoir une téléphonie mobile et unifiée au ROI intéressant ;
- Organiser un service de support en communications unifiées (UC) ;

1.4. Fonctionnalités avancées

Asterisk comprend maintenant des fonctionnalités avancées que l'on trouvait uniquement sur des systèmes coûteux et de très haut niveau :

- De la musique MP3 pour les appelants en attente dans des files d'attentes;
- Des files d'attentes d'appels gérées auxquels une équipe d'agents peut répondre;
- L'intégration du text-to-speech et de la reconnaissance vocale;
- L'enregistrement d'événements aussi dans un fichier texte que dans une base de données SQL;
- La connectivité au réseau PSTN à travers des lignes analogiques et/ou digitales.

1.5. Société Digium

Digium est une société située à Huntsville, dans l'Alabama aux USA qui est le créateur et le premier développeur d'Asterisk. En plus d'être le premier sponsor du développement d'Asterisk, Digium fabrique des cartes d'interfaces téléphoniques et différents matériels pour les PBX Asterisk.

2. Pourquoi Asterisk ?

A condition de dimensionner correctement le CPU du PBX, Asterisk devient une solution extrêmement compétitive grâce à toutes ses fonctionnalités comme l'IVR, l'Unified Messaging, l'enregistrement d'appels, les politiques de filtrage des appels, etc.

2.1. Avantages d'Asterisk

Asterisk comporte pas mal d'avantages :

- Un coût fortement réduit
- Contrôle et indépendance du système téléphonique.
- Environnement de développement rapide et facile
- Riche en fonctionnalités
- Contenu dynamique sur le téléphone
- Un dial plan flexible et puissant
- Un logiciel open-source testé et maintenu

2.2. Un coût fortement réduit

Si l'on compare un PBX traditionnel avec Asterisk sur le plan des interfaces digitales et des téléphones, Asterisk est légèrement moins cher que ceux-ci. Mais Asterisk devient vraiment intéressant quand on ajoute des fonctionnalités avancées comme les boîtes vocales, l'ACD, l'IVR et le CTI, etc. Avec toutes ses fonctionnalités avancées, Asterisk devient considérablement moins cher que les PBX traditionnels toute fonctionnalité étant égale par ailleurs. En fait, comparer Asterisk avec des produits bas de gamme ne convient pas car il propose un grand nombre de fonctions que l'on ne retrouve pas dans ces systèmes.

2.3. Contrôle et indépendance du système téléphonique

Un des grands avantages d'Asterisk est l'indépendance qu'il fournit. A l'heure actuelle, il n'est pas rare de rencontrer des clients qui ne disposent que d'une boîte noire, qui n'ont pas accès à leur système téléphonique ou qui n'ont aucune documentation ou configuration en leur possession. Avec l'approche "do-it-yourself" d'Asterisk, les clients atteignent une liberté totale pour configurer et maintenir leur système de téléphonie.

2.4. Environnement de développement rapide et facile

Asterisk peut être étendu avec des langages de script comme PHP et Perl avec les interfaces AML et AGI. Asterisk est basé open-source et le code original peut être modifié par l'utilisateur. Le code source est écrit en langage ANSI C.

2.5. Riche en fonctionnalités

Asterisk a plusieurs fonctionnalités qui n'existent pas ou qui sont optionnelles dans les PBX traditionnels (par exemple les boîtes vocales, CTI, ACD, IVR, musique d'attente personnalisée et enregistrements). Sur certaines plateformes, le coût de ces fonctionnalités excèdent le coût de la plateforme elle-même.

2.6. Un dial plan flexible et puissant

Un autre avantage d'Asterisk est la capacité de concevoir un dialplan puissant. Dans les PBX traditionnels des fonctionnalités comme le LCR (Last Cost Routing) ne sont pas possibles ou sont optionnels. Avec Asterisk, choisir la meilleure route est facile et propre.

2.7. Un logiciel open-source testé et maintenu

Un dernier avantage d'Asterisk est sa communauté. Différentes sources d'information sont librement disponibles tels que le Wiki d'Asterisk <http://www.voip-info.org>, mais aussi des listes de distribution et des forums. Depuis qu'Asterisk a été largement adopté, tous les bogues découverts ont été corrigés rapidement. Asterisk est probablement le logiciel PBX le plus testé dans le monde. Des versions 1.0 à la 1.2, plus de 3000 changements et bogues dans le code source ont été corrigés, assurant ainsi que le code est à la fois stable et toujours sans erreurs.

2.8. Ce qu'en pense l'industrie

The SIP reference architecture and protocol have operated for enough years now for the business model to become credible for organizations that use third-party network integrators to build, support and plan their telephony and UC environments in small businesses or branches of large enterprises. Digium's product Switchvox is a good fit for companies that want a low-cost turnkey solution suited for small business or small offices of large organizations with a distributed architecture for telephony. Its road map plans for Asterisk, Asterisk Scalable Communications Framework (SCF) and Switchvox to support increasingly larger enterprises.

3. Réticences majeures à l'égard d'Asterisk

3.1. Les parts de marché d'Asterisk sont infimes

On mesure les parts de marché sur base du nombre de PBX vendus. Ces statistiques reprennent les chiffres des plus grands distributions. Asterisk étant un logiciel open-source, il n'apparaît pas dans les chiffres de vente. Des études indépendantes suggèrent plutôt une tendance forte en faveur d'Asterisk. Plus de un million de systèmes fonctionnent sous Asterisk et Digium a vendu plusieurs millions de cartes d'interfaces de téléphonie. ~~La dernière étude de marché Gartner inclut Digium dans sa shortlist parmi Cisco, Microsoft ou encore Avaya.~~

3.2. Asterisk gratuit, comment les concepteur en vivent ?

Digium est une entreprise de développement de logiciels, une communauté, qui a été développé Asterisk depuis 1999. Avec un peu plus d'une centaine d'employés, l'entreprise finance ses activités par la vente d'interface de téléphonie de systèmes PBX comme Switchvox et de logiciels touchant à sept activités. Depuis plusieurs années, elle fait des bénéfices.

3.3. Difficile de trouver du support

Digium fournit un support technique pour ceux qui achètent Asterisk Business Edition. Récemment un support technique pour les versions open-source d'Asterisk est devenu disponible aussi. Des centaines de professionnels sont déjà certifiées et remplissent le rôle de première ligne pour du support et des services professionnels, un peu comme une entreprise IT traditionnelle. De plus en plus de sociétés de service implémentent ou supportent Asterisk. Il y en existe partout dans le monde.

3.4. Asterisk supporte plus de 200 extensions ?

Oui, absolument. Asterisk peuvent être utilisé dans des configurations de plus de 10.000 utilisateurs. Il est franchement évolutif avec des systèmes de répartition de charge et de reprise sur erreur. Il est habituel de trouver des configurations avec un seul serveur pour une centaine d'utilisateurs.

3.5. Seul les « geeks » sont capables de l'installer ?

Avec AsteriskNow et FreePBX, n'importe quel professionnel avec des connaissances limitées en Linux est capable d'installer et de configurer un PBX d'une complexité moyenne. Avec l'aide de d'un GUI, il est possible d'installer et de configurer entièrement un PBX en quelques heures. Les logiciels Open Source disposent d'excellentes interfaces qui facilitent le déploiement et la maintenance de solution VoIP.

3.6. Et si le serveur tombe ?

Un des principaux avantages d'Asterisk et sa capacité à fonctionner sur des systèmes à tolérance de panne. Il est relativement simple et bon marché d'avoir deux serveurs fonctionnant en parallèle. Essayer de faire cela avec un PBX traditionnel.

3.7. On n'est pas favorable au logiciel open-source

Votre entreprise n'utilisera certainement pas de logiciels open-source tant qu'elle ne les aura pas essayés. Beaucoup d'"Appliances" commerciales utilisent Linux comme système d'exploitation. Quoi qu'il en soit, vous restez libres de payer une licence commerciale en achetant les produits Digium.

3.8. Il n'est pas recommandé d'utiliser le CPU pour un PBX

Il est vrai qu'Asterisk utilise le CPU d'un serveur pour traiter la signalisation et la voix tant qu'il n'a pas de DSP dédiés. Mais cette fonctionnalité peut diviser par 5 le coût d'une installation : le système devient alors dépendant des performances du CPU principal. Avec un dimensionnement correct, Asterisk est capable de prendre des volumes importants de trafic. Si vous voulez soulager le processeur principal de ces tâches, vous pouvez aussi utiliser du matériel capable d'annuler l'écho et de codage comme la carte Digium TC400B qui dispose de DSP dédiés.

4. Documentation Asterisk

4.1. Vidéos YouTube

- [Official Asterisk Channel](#)
- [Asterisk 123: Intro to Asterisk](#) de [Astricon 10](#)
- [Asterisk 12 Overview](#) de [Astricon 10](#)

4.2. Compréhension globale

Acronymes et Terminologie

- [Telephony Terminology](#)
- [Asterisk Terms Glossary](#)
- [Telecom Acronyms](#) (very comprehensive)

Protocoles de téléphonie

- [IP Telephony Protocols Overview](#)
- [SIP Overview](#)

- [A Hitchhiker's Guide to SIP](#)

Linux & Unix

- [Linux Newbie Guide](#)
- [Beginner Tutorials](#)
- [Unix Beginner Tutorial](#)

Installer et Configurer Asterisk

- [Asterisk: The Definitive Guide](#) 3rd Edition
- [The Asterisk Wiki](#)

Programmation C

- [C Programming Tutorial](#)
- [Interactive C Tutorial](#)
- [C Programming Quick Guide](#)

4.3. Où obtenir de l'aide ?

Listes de diffusion et IRC

- [Asterisk Mailing List and IRC](#)

Forums Web de discussion

- [Asterisk Community Forums](#)

Communautés en ligne

- [Voip Users Conference](#) et sur [Google+](#)

4.4. Eviter des informations obsolètes ou incorrectes

- Quand on accède à la documentation Web sur Asterisk, on peut trouver des éléments d'information anciens ou incorrects.
- Vérifier la version d'Asterisk qui est mentionnée. Il y a des changements significatifs avec chaque version du logiciel.
- Vérifier la date de publication de la source d'information, surtout si la version n'est pas mentionnée.
- Prenez ces sources avec prudence tant que vous ne l'avez pas croisée avec d'autres et avec une validation pratique.
- Faire référence au [Wiki Asterisk](#) et à la [Chaîne Officielle Asterisk Youtube](#) pour des détails précis et à jour sur chaque version.
- La documentation officielle pourrait ne pas correspondre à ce qui se trouve dans le code source lui-même. Les erreurs peuvent être rapportée sur le [Wiki Asterisk](#) ou sur [Asterisk Issues Tracker](#) .

Asterisk Concepts de bases

- 1. Sources à consulter
- 2. Introduction à Asterisk
 - 2.1. Versions d'Asterisk
 - 2.2. Composants d'Asterisk
 - 2.3. Channels
 - Channels qui s'interfacent sur le PSTN
 - Channels qui s'interfacent en Voice-over IP
 - Autres channels
 - 2.4. Codec
 - Codecs et traduction de codecs
 - Codecs disponibles
 - 2.5. Modules
- 3. Installation
 - 3.1. Préparation du système
 - 3.2. Acquérir les sources
 - 3.3. Configurer et compiler
- 4. Adaptation du pare-feu
 - 4.1. FirewallD
 - 4.2. Désactivation
 - 4.3. Retour à iptables
- 5. Configuration minimale
 - 5.1. Fichier indications.conf
 - 5.2. fichier asterisk.conf
 - 5.3. Fichier modules.conf
 - 5.4. Fichier musiconhold.conf
- 6. Un Simple PBX
 - 6.1. Fichier sip.conf
 - 6.2. Fichier pjsip.conf
 - 6.3. Fichier extensions.conf
 - 6.4. Redémarrage du service
 - 6.5. Contextes, extensions et priorités
 - 6.6. Applications
- 7. Commandes de base
 - 7.1. Démarrage du service
 - 7.2. Lancement de la console
 - 7.3. Dialplan
 - 7.4. Vérification chan_pjsip
 - 7.5. Vérification chan_sip
- 8. Collecter des logs dans la console

1. Sources à consulter

Ces chapitres de la documentation officielle ont été examinés en salle de formation :

- <https://wiki.asterisk.org/wiki/display/AST/About+the+Project>
- <https://wiki.asterisk.org/wiki/display/AST/Getting+Started>

- <https://wiki.asterisk.org/wiki/display/AST/Fundamentals>
- <https://wiki.asterisk.org/wiki/display/AST/Basic+PBX+Functionality>
- <https://wiki.asterisk.org/wiki/display/AST/Hello+World>

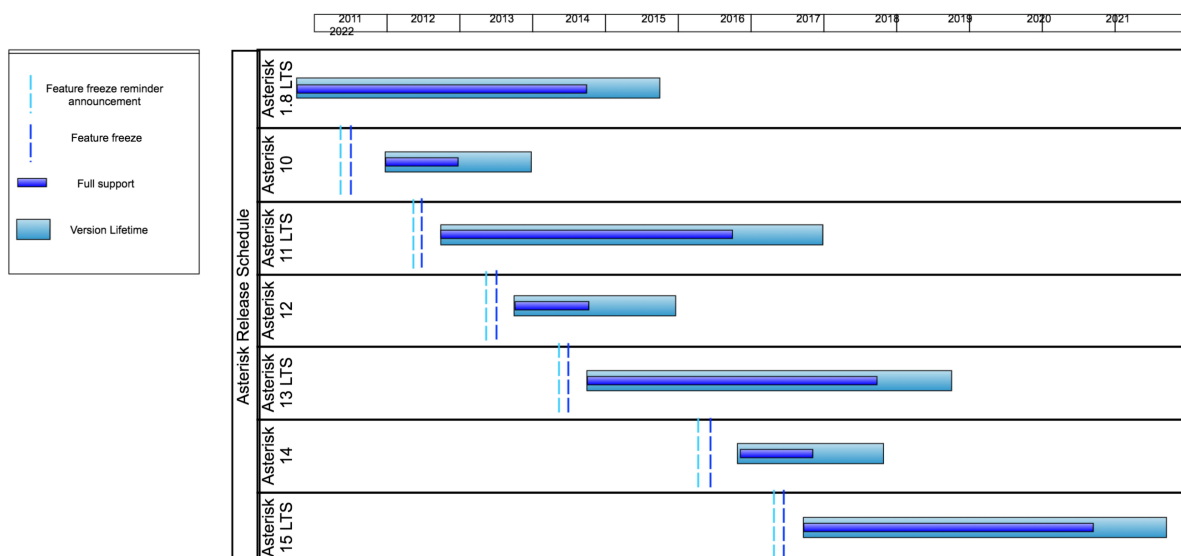
Ils correspondent aux chapitres suivants du livre officiel Asterisk *The Definitive Guide, 4th Edition* :

1. A Telephony Revolution
2. Asterisk Architecture
3. Installing Asterisk
4. Initial Configuration Tasks
5. User Device Configuration
6. Dialplan Basics
 - Dialplan Syntax
 - A Simple Dialplan

2. Introduction à Asterisk

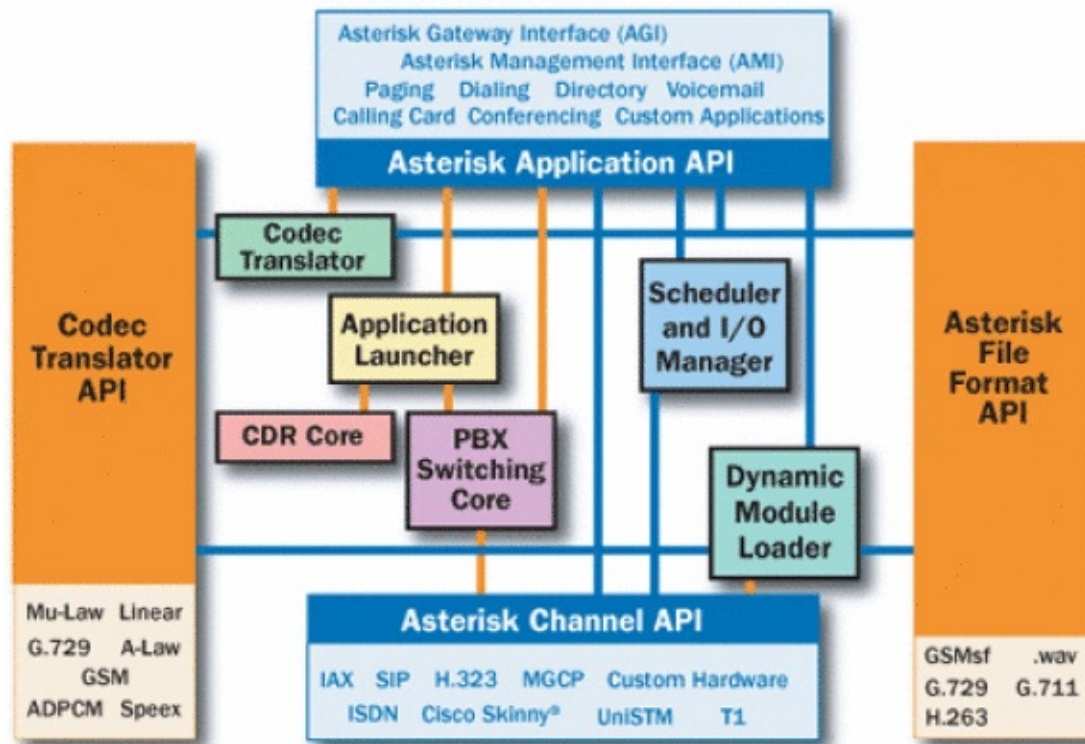
2.1. Versions d'Asterisk

Release Series	Release Type	Release Date	Security Fix Only	EOL
11.x	LTS	2012-10-25	2016-10-25	2017-10-25
12.x	Standard	2013-12-20	2014-12-20	2015-12-20
13.x	LTS	2014-10-24	2018-10-24	2019-10-24
14.x	Standard	2016-09-26	2017-09-26	2018-09-26
15.x	LTS	2017-10-03	2018-10-03	2019-10-03

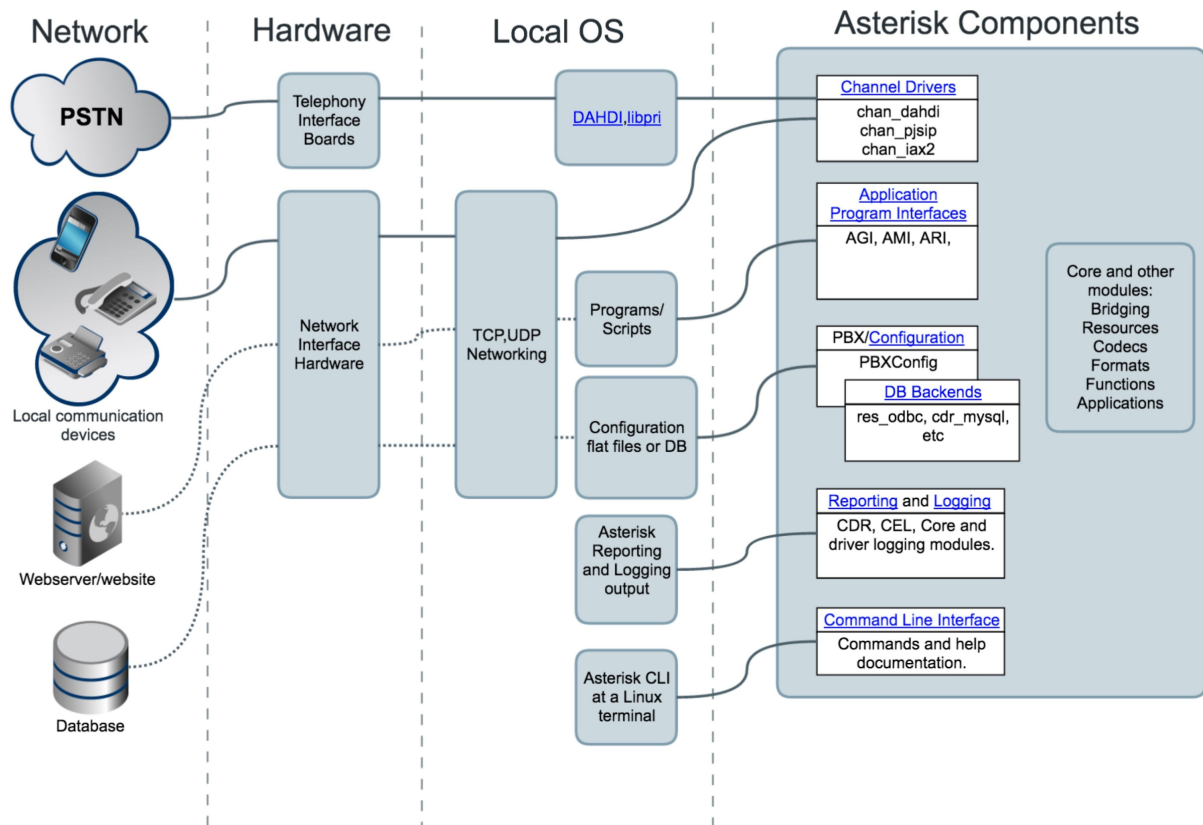


<https://wiki.asterisk.org/wiki/display/AST/Asterisk+Versions>

2.2. Composants d'Asterisk



<https://wiki.asterisk.org/wiki/display/AST/Asterisk+Architecture%2C+The+Big+Picture>



2.3. Channels

Un "Channel" est l'équivalent d'une ligne téléphonique mais en format digital. Un "canal" consiste en général en un système de signalisation analogique digitale (TDM) ou une combinaison de codecs et de protocoles de signalisation comme par exemple SIP-GSM ou IAX-uLaw. Au début de la téléphonie, toutes les connexions téléphoniques étaient analogiques et étaient susceptibles de subir de l'écho ou du bruit. Plus tard la plupart des systèmes ont été convertis en systèmes digitaux, avec un son analogique converti dans un format digital en utilisant le «pulse code modulation» (PCM) dans la plupart des cas. Ce format permet une transmission vocale avec 64 Kbps sans compression (PCMU ou PCMA).

Channels qui s'interfacent sur le PSTN

- chan_dahdi : Supporte les cartes venant de Sangoma, Digium, Xorcom et autres.
- chan_mISDN : Supporte les cartes ISDN* (RNIS) cards se basant sur les pilotes Linux ISDN.

Channels qui s'interfacent en Voice-over IP

- chan_sip : Supporte la voice-over IP en utilisant le protocole SIP. **Le pilote est remplacé par défaut par channel_pjsip à partir d'Asterisk 12 !**
- chan_pjsip : fournit les nouveaux services SIP
- chan_iax : Supporte la voice-over IP en utilisant le protocole IAX2.

Autres channels

- chan_agent : utilisé dans le cadre d'un ACD. Ce channel n'a pas de rapport avec un matériel ou un protocole. Il peut être utilisé pour de la mobilité, permettant à une personne d'utiliser n'importe quel téléphone seulement en connectant à l'agent.
- chan_local : C'est un « pseudo » channel qui peut créer un bouclage dans un dialplan (dans un « Context » Asterisk). C'est utile pour du routage récursif.
- ...

2.4. Codec

Un codec est un procédé qui permet de coder/décoder la voix en données digitales. Leur performance dépend de leur rapport qualité / bande passante.

Codecs et traduction de codecs

Nous avons l'habitude de placer autant de connexions voix que possible dans un réseau de données. Les codecs activent de nouvelles fonctionnalités en voix digitale telle que la compression permettant des gains en bande passante avec un ratio de 8 à 1. Mais il y a aussi des fonctionnalités de détection d'activité vocale, d'annulation de perte de paquets et de génération de bruit de confort. Plusieurs codecs sont disponibles sous Asterisk et peuvent traduits de manière transparente l'un vers l'autre.

Codecs disponibles

Pour vérifier quels codecs sont disponibles sur votre système veuillez frapper dans la console :

```
CLI>core show translation
```

Les codecs suivants sont supportés par Asterisk :

- G.711 ulaw (USA) - (64 Kbps). - MOS 4.3
- G.711 alaw (Europe) - (64 Kbps). - MOS 4.3

- G.722 (Haute Définition) – (64 Kbps).
- G.723.1 – seulement en pass-through mode (non traduisible) - MOS 3.9
- G.726 - (16/24/32/40kbps) - MOS 3.8
- G.729a - Licence nécessaire (8Kbps) - MOS 3.7
- GSM - (12-13 Kbps) - MOS 3.5
- iLBC - (15 Kbps) - MOS 4.14

Ils sont caractérisés par une valeur MOS qui spécifie leur qualité audible.

2.5. Modules

Pour établir les appels d'un téléphone à un autre, c'est l'application `dial()` qui est sollicitée. La plupart des fonctionnalités d'Asterisk sont implémentées en tant qu'applications.

On peut constater les applications disponibles avec la commande CLI `core show applications` :

```
CLI>core show applications
```

Vous pouvez ajouter des applications venant d'add-ons Asterisk, des fournisseurs tiers ou en en développant vous-même.

3. Installation

Distribution : Centos7

```
cat /etc/redhat-release
```

```
CentOS Linux release 7.2.1511 (Core)
```

```
CentOS Linux release 7.4.1708 (Core)
```

Minimum : 1Go RAM

3.1. Préparation du système

En tant qu'utilisateur `root`.

Si on utilise un serveur public (chez [Scaleway](#) par exemple), il est conseillé de privatiser l'accès avec [OpenVPN](#) à l'aide de la procédure suivante.

```
wget https://git.io/vpn -O openvpn-install.sh && bash openvpn-install.sh
```

Elle a pour effet de :

- Installer et de configurer un serveur OpenVPN
- Créer des fichiers de configuration client
- Installer et configurer le routage et le pare-feu n'autorisant que le service SSH et OpenVPN
- Avec cette configuration tout trafic passera par le tunnel

Pour supprimer ce routage par défaut et pousser d'autres routes, on pourra s'inspirer de cette procédure :

```
sed -i s/^push/#push/g /etc/openvpn/server.conf
```

```
ip=$(ip -4 addr show eth0 | grep inet | awk '{ print $2; }' | sed 's/\./.*$/')
echo push "route ${ip} 255.255.255.255" >> /etc/openvpn/server.conf
systemctl restart openvpn@server.service
```

Mise à jour du système et redémarrage

```
yum -y install epel-release && yum update -y && shutdown -r now
```

Synchronisation temporelle et installation de NTP (Network Time Protocol)

```
yum -y install chrony && systemctl enable chronyd && systemctl start chronyd
```

Localisation.

```
timedatectl list-timezones
timedatectl set-timezone Europe/Brussels
```

Nom d'hôte, par exemple "pbx01"

```
hostnamectl set-hostname pbx01
```

Locale.

```
localectl set-locale LANG=fr_BE.UTF-8
localectl set-keymap be
localectl
```

```
System Locale: LANG=fr_BE.UTF-8
               VC Keymap: be
               X11 Layout: be
               X11 Variant: oss
```

Ajouter un utilisateur `asteriskpbx`.

```
adduser asteriskpbx && passwd asteriskpbx && yum -y install sudo && gpasswd -a asteriskpbx wheel && exit
```

3.2. Acquérir les sources

Se reconnecter sous l'utilisateur `asteriskpbx` :

```
su - asteriskpbx
```

Créer la structure des dossiers.

```
mkdir -p ~/src/asterisk-complete/asterisk
cd ~/src/asterisk-complete/asterisk
```

Obtenir les sources d'Asterisk (ici version 13) et installer les dépendances.

```
wget http://downloads.asterisk.org/pub/telephony/asterisk/asterisk-13-current.tar.gz
tar xvfz asterisk-13-current.tar.gz
mv asterisk-13.12.2 current
cd current
```



```
sudo ./contrib/scripts/install_prereq install
./contrib/scripts/get_mp3_source.sh
```

3.3. Configurer et compiler

Sur n'importe quelle plateforme, on peut exécuter la compilation selon des principes similaires.

Un script vérifie la configuration de la compilation.

```
./configure --libdir=/usr/lib64 --with-pjproject-bundled
```

Support des MP3 (format_mp3), sons français (CORE-SOUNDS-FR-GSM et EXTRA-SOUNDS-FR-GSM) à choisir dans *menuselect*. Le terminal doit être à une taille minimale de 80 x 27.

```
make menuselect
```

Compilation.

make

Installation des binaires.

```
sudo make install
```

A la fin de la compilation, on nous propose de générer des copier fichiers d'exemple et de documentation. Pour l'exercice, nous construirons nous-mêmes les fichiers de configuration.

```
+----- Asterisk Installation Complete -----+
+
+   YOU MUST READ THE SECURITY DOCUMENT
+
+ Asterisk has successfully been installed.
+ If you would like to install the sample
+ configuration files (overwriting any
+ existing config files), run:
+
+ For generic reference documentation:
+   make samples
+
+ For a sample basic PBX:
+   make basic-pbx
+
+
+----- or -----+
+
+ You can go ahead and install the asterisk
+ program documentation now or later run:
+
+   make progdocs
+
+
+ **Note** This requires that you have
+ doxygen installed on your local system
+-----+
```

Pour installer le service.

```
sudo make config
```

Pour le démarrer.

```
sudo systemctl start asterisk
```

Pour l'activer au démarrage.

```
sudo systemctl enable asterisk
```

```
asterisk.service is not a native service, redirecting to /sbin/chkconfig.  
Executing /sbin/chkconfig asterisk on
```

Pour vérifier le statut du service.

```
sudo systemctl status asterisk
```

```
• asterisk.service - LSB: Asterisk PBX  
  Loaded: loaded (/etc/rc.d/init.d/asterisk)  
  Active: active (running) since ven 2016-11-11 16:52:06 CET; 24s ago  
    Docs: man:systemd-sysv-generator(8)  
  Main PID: 27274 (asterisk)  
    CGroup: /system.slice/asterisk.service  
            └─27272 /bin/sh /usr/sbin/safe_asterisk  
              └─27274 /usr/sbin/asterisk -f -vvvg -c  
  
nov 11 16:52:05 localhost.localdomain systemd[1]: Starting LSB: Asterisk PBX...  
nov 11 16:52:06 localhost.localdomain asterisk[26129]: Starting asterisk:  
nov 11 16:52:06 localhost.localdomain systemd[1]: PID file /var/run/asterisk/asterisk.pid not readable (yet?)..  
.art.  
nov 11 16:52:06 localhost.localdomain systemd[1]: asterisk.service: Supervising process 27274 which is not ou..  
.its.  
nov 11 16:52:06 localhost.localdomain systemd[1]: Started LSB: Asterisk PBX.  
nov 11 16:52:12 localhost.localdomain systemd[1]: asterisk.service: Supervising process 27274 which is not ou..  
.its.  
Hint: Some lines were ellipsized, use -l to show in full.
```

Note : Adaptation des droits sur les dossiers installés

```
sudo chown -R asteriskpbx:asteriskpbx /var/lib/asterisk/  
sudo chown -R asteriskpbx:asteriskpbx /var/spool/asterisk/  
sudo chown -R asteriskpbx:asteriskpbx /var/log/asterisk/  
sudo chown -R asteriskpbx:asteriskpbx /var/run/asterisk/
```

ou encore en une seule ligne :

```
sudo chown -R asteriskpbx:asteriskpbx {/var/lib,/var/spool,/var/log,/var/run}/asterisk
```

Une option d'installation configure la rotation des logs.

```
$ sudo make install-logrotate
```

4. Adaptation du pare-feu

Malgré qu'il soit conseillé d'utiliser firewalld, il est possible de le désactiver et de repasser aux commandes iptables.

Options disponibles

- Firewalld : le plus actualisé
- Désactivation : déconseillé
- Iptables

4.1. Firewalld

```
sudo systemctl start firewalld && systemctl enable firewalld
```

```
sudo firewall-cmd --zone=public --add-port=5060/udp --permanent
sudo firewall-cmd --zone=public --add-port=5060/tcp --permanent
sudo firewall-cmd --zone=public --add-port=5061/tcp --permanent
sudo firewall-cmd --zone=public --add-port=4569/udp --permanent
sudo firewall-cmd --zone=public --add-port=10000-20000/udp --permanent
```

```
sudo firewall-cmd --reload
sudo firewall-cmd --zone=public --list-all
```

```
public (default, active)
  interfaces: eno16777736
  sources:
  services: dhcpv6-client ssh
  ports: 5061/tcp 4569/udp 5060/tcp 10000-20000/udp 5060/udp
  masquerade: no
  forward-ports:
  icmp-blocks:
  rich rules:
```

4.2. Désactivation

```
sudo systemctl stop firewalld && systemctl disable firewalld
```

4.3. Retour à iptables

```
sudo yum install -y iptables-services
sudo systemctl enable iptables
sudo systemctl start iptables
```

```
sudo iptables -F
sudo iptables -X
sudo iptables -A INPUT -i lo -j ACCEPT
sudo iptables -A INPUT -d 127.0.0.0/8 -j REJECT
sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A OUTPUT -j ACCEPT
sudo iptables -A INPUT -p tcp -m state --state NEW --dport 22 -j ACCEPT
sudo iptables -A -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
sudo iptables -A -A INPUT -m limit --limit 5/min -j LOG --log-prefix "iptables denied: " --log-level 7
sudo iptables -A INPUT -p udp -m udp --dport 5060 -j ACCEPT
sudo iptables -A INPUT -p udp -m tcp --dport 5060 -j ACCEPT
sudo iptables -A INPUT -p udp -m tcp --dport 5061 -j ACCEPT
sudo iptables -A INPUT -p udp -m udp --dport 4569 -j ACCEPT
sudo iptables -A INPUT -p udp -m udp --dport 10000:20000 -j ACCEPT
sudo iptables -A -A INPUT -j DROP
sudo iptables -A -A FORWARD -j DROP
```

Note: DHCPc/DHCPv6c ACCEPT ?

```
sudo iptables-restore < /etc/iptables.firewall.rules
```

```
sudo iptables -L
```

```
sudo /usr/libexec/iptables/iptables.init save
```

5. Configuration minimale

On doit trouver au minimum 5 fichiers dans le dossier `/etc/asterisk` :

- `asterisk.conf`
- `indications.conf`
- `modules.conf`
- `extensions.conf`
- `sip.conf` OU `pjsip.conf`

Accessoirement, pour commencer, on peut ajouter le fichier `musiconhold.conf` .

5.1. Fichier indications.conf

Créer le dossier `/etc/asterisk` et copie du fichier `indications.conf`

```
sudo mkdir -p /etc/asterisk
sudo chown asteriskpbx:asteriskpbx /etc/asterisk
cd /etc/asterisk/
cp ~/src/asterisk-complete/asterisk/current/configs/samples/indications.conf.sample ./indications.conf
```

5.2. fichier asterisk.conf

Copie du fichier `asterisk.conf` dans le dossier `/etc/asterisk` et changement du `runuser` et `rungroup` avec la valeur `asteriskpbx` :

```
cp ~/src/asterisk-complete/asterisk/current/configs/samples/asterisk.conf.sample ./asterisk.conf
```

Éléments à ajouter dans le fichier `asterisk.conf`

```
runuser = asteriskpbx
rungroup = asteriskpbx
```

5.3. Fichier modules.conf

Créer le fichier `modules.conf` . C'est le fichier d'activation/désactivation des modules.

```
cat >> /etc/asterisk/modules.conf
```

```
; The modules.conf file, used to define which modules Asterisk should load (or
; not load).
;
[modules]
```

```

autoload=yes

; Resource modules currently not needed
noload => res_speech.so
noload => res_phoneprov.so
noload => res_ael_share.so
noload => res_clialaliases.so
noload => res_adsi.so

; PBX modules currently not needed
noload => pbx_ael.so
noload => pbx_dundi.so

; Channel modules currently not needed
noload => chan_oss.so
noload => chan_mgcp.so
noload => chan_skinny.so
noload => chan_phone.so
noload => chan_agent.so
noload => chan_unistim.so
noload => chan_alsa.so

; Application modules currently not needed
noload => app_nbcat.so
noload => app_amd.so
noload => app_minivm.so
noload => app_zapateller.so
noload => app_ices.so
noload => app_sendtext.so
noload => app_speech_utils.so
noload => app_mp3.so
noload => app_flash.so
noload => app_getcpeid.so
noload => app_setcallerid.so
noload => app_adsiprog.so
noload => app_forkcdr.so
noload => app_sms.so
noload => app_morsecode.so
noload => app_followme.so
noload => app_url.so
noload => app_alarmreceiver.so
noload => app_disa.so
noload => app_dahdiras.so
noload => app_senddtmf.so
noload => app_sayunixtime.so
noload => app_test.so
noload => app_externalivr.so
noload => app_image.so
noload => app_dictate.so
noload => app_festival.so

```

Ctrl+D

5.4. Fichier musiconhold.conf

Configurer `musiconhold.conf` :

```
cat >> musiconhold.conf
```

```
; musiconhold.conf
```

```
[general]

[default]
mode=files
directory=moh
```

Ctrl+D

6. Un Simple PBX

6.1. Fichier sip.conf

Pour comparaison avec le fichier `pjsip.conf` et pour mémoire, un exemple de fichier `sip.conf`.

```
;sip.conf
[general]
transport=udp

[telephones](!)
type=friend
host=dynamic
context=appels-internes
disallow=all
allow=ulaw

[telephone1](telephones)
secret=1111
[telephone2](telephones)
secret=1111
[telephone3](telephones)
secret=1111
```

6.2. Fichier pjsip.conf

```
;pjsip.conf
;=====TRANSPORT

[simpletrans]
type=transport
protocol=udp
bind=0.0.0.0

;=====ENDPOINT TEMPLATES

[endpoint-basic](!)
type=endpoint
context=appels-internes
disallow=all
allow=ulaw

[auth-userpass](!)
type=auth
auth_type=userpass

[aor-single-reg](!)
type=aor
max_contacts=1

;=====EXTENSION telephone1
```

```
[telephone1](endpoint-basic)
auth=telephone1
aors=telephone1

[telephone1](auth-userpass)
password=1111
username=telephone1

[telephone1](aor-single-reg)

;=====EXTENSION telephone2

[telephone2](endpoint-basic)
auth=authtel2
aors=telephone2

[authtel2](auth-userpass)
password=1111
username=telephone2

[telephone2](aor-single-reg)

;=====EXTENSION telephone3

[telephone3](endpoint-basic)
auth=authtel3
aors=telephone3

[authtel3](auth-userpass)
password=1111
username=telephone3

[telephone3](aor-single-reg)
```

6.3. Fichier extensions.conf

```
;extensions.conf
[general]

[globals]

[appels-interne]
exten=>2301,1,Dial(PJSIP/telephone1,20)
exten=>2302,1,Dial(PJSIP/telephone2,20)
exten=>2303,1,Dial(PJSIP/telephone3,20)
exten=>9999,1,Dial(PJSIP/telephone1,20)

exten=>1234,1,Answer()
same=>n,SayNumber(${EXTEN})
same=>n,Hangup()
```

6.4. Redémarrage du service

```
$ sudo systemctl restart asterisk
```

6.5. Contextes, extensions et priorités

<https://wiki.asterisk.org/wiki/display/AST/Contexts%2C+Extensions%2C+and+Priorities>

6.6. Applications

Applications :

- Dial
- Answer
- SayNumber
- Hangup
- Playback
- Background

7. Commandes de base

7.1. Démarrage du service

```
$ sudo systemctl restart asterisk
```

7.2. Lancement de la console

Lancer le CLI Asterisk à partir d'une console

```
asterisk -rvvv
```

Lancer le CLI Asterisk avec verbosité

```
asterisk -rvvvvvv
```

7.3. Dialplan

Relancer la configuration du Dialplan (extensions.conf)

```
> dialplan reload
```

Visualiser le dialplan

```
> dialplan show
[ Context '__func_periodic_hook_context__' created by 'func_periodic_hook' ]
'beep' (CID match '') => 1. Answer() [func_periodic_hook]
                        2. Playback(beep) [func_periodic_hook]
'hook' (CID match '') => 1. Set(EncodedChannel=${CUT(HOOK_CHANNEL,-,1-2)}) [func_periodic_hook]
                        2. Set(GROUP_NAME=${EncodedChannel}${HOOK_ID}) [func_periodic_hook]
                        3. Set(GROUP(periodic-hook)=${GROUP_NAME}) [func_periodic_hook]
                        4. ExecIf(${GROUP_COUNT(${GROUP_NAME}@periodic-hook)} > 1)?Hangup() [func_periodic_hook]
                        5. Set(ChannelToSpy=${URIDECODE(${EncodedChannel})}) [func_periodic_hook]
                        6. ChanSpy(${ChannelToSpy},qEB) [func_periodic_hook]

[ Context 'appels-internes' created by 'pbx_config' ]
'1234' => 1. Answer() [pbx_config]
          2. SayNumber(${EXTEN}) [pbx_config]
          3. Hangup() [pbx_config]
'2301' => 1. Dial(SIP/telephone1,20) [pbx_config]
'2302' => 1. Dial(SIP/telephone2,20) [pbx_config]
'2303' => 1. Dial(SIP/telephone3,20) [pbx_config]
'9999' => 1. Dial(SIP/telephone1,20) [pbx_config]
```


7.4. Vérification chan_pjsip

```
> pjsip show aors
```

```
> pjsip show auths
```

```
> pjsip show contacts
```

```
> pjsip show endpoints
```

7.5. Vérification chan_sip

Relancer la configuration SIP.

```
> sip reload
```

Vérifier les téléphones enregistrés (REGISTER).

```
> sip show peers
```

Vérifier les paramètres des téléphones.

```
> sip show users
```

8. Collecter des logs dans la console

- <https://wiki.asterisk.org/wiki/display/AST/Collecting+Debug+Information> :

```
> logger add channel myDebugLog {notice,warning,error,debug,verbose,dtmf}  
> core set verbose 5  
> core set debug 5  
> module reload logger
```

Asterisk intermédiaire

- Sources
- 1. Interactivité
 - 1.1. Rappels et notions sur le Dialplan
 - Application Dial()
 - Variables
 - Correspondance de motifs d'appel
 - Directive Include
 - Applications Goto, Background, WaitExten
 - 1.2. Service d'enregistrement audio et serveur interactif
- 2. Boîtes vocales
 - 2.1. NTP/SMTP
 - 2.2. Configuration
 - 2.3. Extensions.conf / Plan d'appel
 - 2.4. Commandes
 - 2.5. Tests à partir des téléphones
- 3. Connectivité externe
 - 3.1. Configuration du Trunk
 - 3.2. Route sortante
 - 3.3. Route entrante
 - 3.4 Simplification wizard de pjsip.conf
 - 3.5. Internationalisation
- 4. Trafic sécurisé
 - 4.1. Simple trunk SIP
 - Étape 1 Trunk SIP
 - Étape 2 Route sortante
 - 4.2. Client/serveur SIPS/SRTP
 - 4.3. Serveur/serveur SIPS
 - 4.4. Trunking IAX2
- 5. Fonctionnalités diverses
 - 5.1. Dire l'adresse IP avec la fonction curl
 - 5.2. Fichiers d'exemples
 - 5.3. Exploration de fonctionnalités plus avancées

Sources

Ce document fait référence aux chapitres suivants du livre officiel Asterisk: The Definitive Guide, 4th Edition :

-6. Dialplan Basics, 7. Outside Connectivity, 8. Voicemail, 9. Internationalization, 10. Deeper into the Dialplan, 11. Parking, Paging, and Conferencing, 13. Automatic Call Distribution (ACD) Queues, 14. Device States, 15. The Automated Attendant, 17. Interactive Voice Response.

1. Interactivité

Chapitre 6 "Dialplan Basics"

1.1. Rappels et notions sur le Dialplan

- <https://wiki.asterisk.org/wiki/display/AST/Contexts%2C+Extensions%2C+and+Priorities>
- <https://wiki.asterisk.org/wiki/display/AST/Special+Dialplan+Extensions>

Application Dial()

Chapitre 6, Section "Using the Dial() Application"

Définition de paramètres

Variables

Chapitre 6, Section "Using Variables"

- <https://wiki.asterisk.org/wiki/display/AST/Channel+Variables>
- <https://wiki.asterisk.org/wiki/display/AST/Manipulating+Variables+Basics>
- <https://wiki.asterisk.org/wiki/display/AST/Setting+and+Substituting+Channel+Variables>
- <https://wiki.asterisk.org/wiki/display/AST/Selecting+Characters+from+Variables>

Correspondance de motifs d'appel

Chapitre 6, Section "Pattern Matching"

- <https://wiki.asterisk.org/wiki/display/AST/Pattern+Matching>

Motifs :

- `_` : définit un pattern
- `x` : 0 à 9
- `N` : 2 à 9
- `z` : 1 à 9
- `.` : wilcard

Directive Include

Chapitre 6, Section "Include"

- <https://wiki.asterisk.org/wiki/display/AST/Include+Statements+Basics>
- <https://wiki.asterisk.org/wiki/display/AST/Using+Include+Statements+to+Create+Classes+of+Service>

Notion de "Classes de service"

```
appels-internationaux
  appels-nationaux
    appels-interne
      services-interne
      ...
```

Applications Goto, Background, WaitExten

Chapitre 6, section "The Goto(), Background(), and WaitExten() Applications"

- <https://wiki.asterisk.org/wiki/display/AST/Goto+Application+and+Priority+Labels>
- <https://wiki.asterisk.org/wiki/display/AST/Background+and+WaitExten+Applications>
- <https://wiki.asterisk.org/wiki/display/AST/Record+Application>
- <https://wiki.asterisk.org/wiki/display/AST/SayDigits%2C+SayNumber%2C+SayAlpha%2C+and+SayPhonetic+Applications>

- <https://wiki.asterisk.org/wiki/display/AST/Creating+a+Simple+IVR+Menu>
- <https://wiki.asterisk.org/wiki/display/AST/Handling+Special+Extensions>

1.2. Service d'enregistrement audio et serveur interactif

Chapitre 15 "The Automated Attendant"

Scénario proposé : service d'enregistrement de fichiers audio notés `xx.wav` (en deux chiffres). On propose ici d'encoder le message `01` qui sera utilisé sur le serveur interactif.

- Création d'un service d'enregistrement audio (numéro `_71XX`)
- Création d'un service de rejeu de fichier (numéro `_72XX`)
- Mise en œuvre d'un prototype de serveur interactif

```
;extensions.conf adapté
;service d'enregistrement audio (71XX) et rejeu du fichier audio (72XX)

[appels-internes]
exten=>_230X,1,Dial(PJSIP/telephone${EXTEN:3})
exten=>201,1,GoTo(testmenu,start,1)
include=>recplay

[recplay]

exten => _71XX,1,answer()
same => n,Record(en/${EXTEN:2}.ulaw,,10,y)
same => n,Playback(${EXTEN:2})
same => n,Hangup()

exten => _72XX,1,answer()
same => n,Playback(${EXTEN:2})
same => n,Hangup()

[testmenu]

exten => start,1,Answer()
same => n,Background(01)
      same => n,WaitExten(10)

exten => 1,1,Dial(PJSIP/telephone1,10)
      same => n,Playback(vm-nobodyavail)
      same => n,Hangup()

exten => i,1,Playback(pbx-invalid)
      same => n,Goto(testmenu,start,1)

exten => t,1,Playback(vm-goodbye)
      same => n,Hangup()
```

2. Boîtes vocales

Chapitre 8 "Voicemail"

2.1. NTP/SMTP

Sous Debian/Ubuntu :

```
apt-get install ntpdate
ntpdate be.pool.ntp.org
dpkg-reconfigure tzdata
```

Configuration adéquate du serveur SMTP.

2.2. Configuration

Commandes :

```
mv /etc/asterisk/voicemail.conf /etc/asterisk/voicemail.conf.old
nano /etc/asterisk/voicemail.conf
```

```
; voicemail.conf
[general]
format=wav
serveremail=no-reply@raspbx
attach=yes
skipms=3000
maxsilence=10
silencethreshold=128
maxlogins=3
emaildateformat=%A, %B %d, %Y at %r
pagerdateformat=%A, %B %d, %Y at %r
sendvoicemail=yes ; Allow the user to compose and send a voicemail while inside

[zonemessages]
european=Europe/Brussels|'vm-received' a d b 'digits/at' HM

[default]
2301 => 1111,user1,user1@user1.be
2302 => 1111,user2,user2@user2.be
```

2.3. Extensions.conf / Plan d'appel

```
;extensions.conf adapté
;*98 pour atteindre le menu voicemail

[appels-interne]
include=>vm-menu

[vm-menu]

exten => *98,1,VoiceMailMain()

exten=>2301,1,Dial(PJSIP/telephone1,10)
same => n,VoiceMail(2301@default,u)

exten=>2302,1,Dial(PJSIP/telephone2,10)
same => n,VoiceMail(2302@default,u)
```

2.4. Commandes

```
*CLI>voicemail reload
*CLI>dialplan reload
```

2.5. Tests à partir des téléphones

3. Connectivité externe

Chapitre 7 "Outside Connectivity"

3.1. Configuration du Trunk

```
;sip.conf adapté
[general]
register => ACCOUNT_NUMBER:SIP_PASSWORD@sip.de.anveo.com:5010
context=unauthenticated ; default context for incoming calls
allowguest=no ; disable unauthenticated calls
srvlookup=no ; disabled DNS SRV record lookup on outbound calls

[anveo]
type=friend
host=sip.de.anveo.com
port=5010
username= ACCOUNT_NUMBER
secret= SIP_PASSWORD
insecure=port,invite
disallow=all
allow=ulaw
context=from-anveo
```

Avec PJSIP, on fera référence au wiki https://wiki.asterisk.org/wiki/display/AST/res_pjsip+Configuration+Examples mais surtout à <https://wiki.asterisk.org/wiki/display/AST/PJSIP+Configuration+Wizard> et au fichier d'exemple des sources. Le type wizard facilite les configurations de scénarios communs.

```
; pjsip.conf
[anveo]
type = wizard
sends_auth = yes
sends_registrations = yes
transport = ipv4
remote_hosts = sip.de.anveo.com:5010
outbound_auth/username = ACCOUNT_NUMBER
outbound_auth/password = SIP_PASSWORD
endpoint/allow = ulaw
endpoint/context = from-anveo
```

3.2. Route sortante

```
;extensions.conf adapté

[appels-nationaux]
include => appels-internes
include => appels-fixes-be
include => appels-mobiles-be

[appels-fixes-be]
exten => _0XXXXXXX,1,Set(CALLERID(number)=32123456789)
same => n,Dial(PJSIP/anveo/${EXTEN})

[appels-mobiles-be]
```

```

exten => _0XXXXXXXX,1,Set(CALLERID(number)=32123456789)
same => n,Dial(PJSIP/anveo/${EXTEN})

```

```

;sip.conf adapté
[telephones](!)
context=appels-nationaux

```

3.3. Route entrante

```

[from-anveo]
exten => 3212345679,1,Dial(SIP/telephone1,5)
same => n,Goto(appels-interne,1234,1)

```

3.4 Simplification wizard de pjsip.conf

Voir <https://wiki.asterisk.org/wiki/display/AST/PJSIP+Configuration+Wizard> et le fichier d'exemple des sources :

```

[telephones](!)
type = wizard
accepts_auth = yes
accepts_registrations = yes
transport = ipv4
endpoint/allow = ulaw
endpoint/context = default

[telephone1](telephones)
inbound_auth/username = telephone1
inbound_auth/password = 1111

```

3.5. Internationalisation

Chapitre 9 "Internationalization"

<https://wiki.asterisk.org/wiki/display/AST/Internationalization+and+Localization>

- Fichiers et dossiers de langue (/var/lib/asterisk/sounds/en , /var/lib/asterisk/sounds/fr , ...)
- Variable de langue
 - sur le l'appel (Set(CHANNEL(language)=fr))
 - sur l'utilisateur (dans pjsip.conf : endpoint/language = en Asterisk 14)
 - globale (section [globals] dans extensions.conf .

4. Trafic sécurisé

4.1. Simple trunk SIP

On trouvera un exemple chan_sip à adapter en PJSIP.

Étape 1 Trunk SIP

1.a. Le plus simple est de créer deux périphériques SIP aux noms identiques et d'identifier leur adresse IP.

Sur le serveur du **Site A** dans le fichier sip.conf :

```

[Trunk_AB]
type=peer

```

```
host=192.168.1.201
context=trunk_ab
username=Trunk_AB ; nom du périphérique distant (ici identiques)
secret=test1234 ; secret partagé
```

Sur le serveur du **Site B** dans le fichier sip.conf :

```
[Trunk_AB]
type=peer
host=192.168.1.200
context=trunk_ab
username=Trunk_AB
secret=test1234
```

1.b. Vérifiez l'état du trunk dans la console Asterisk :

```
*CLI> sip show peers
```

Étape 2 Route sortante

2.a. Il s'agit de modifier le plan d'appel dans `extensions.conf` sur les deux serveurs, ajoutez la ligne dans le contexte des appels entrants `[appels-internes]` via un `include` sur le contexte `[trunk_ab]`. On utilise le préfixe d'appel `9` pour joindre la destination sur le Trunk.

```
exten => _9XXXX,1,Dial(SIP/Trunk_AB/${EXTEN:1})
```

2.b. Tentez de joindre un poste du Site B à partir d'un poste du Site A et inversement.

2.c. Comment devez-vous modifier votre configuration si vous nommez différemment les périphériques SIP du trunk ?

2.d. Comment sont traités les appels qui n'aboutissent pas ? Peut-on leur donner un autre traitement ?

4.2. Client/serveur SIPS/SRTP

- <https://wiki.asterisk.org/wiki/display/AST/Secure+Calling+Tutorial>

4.3. Serveur/serveur SIPS

- <https://wiki.asterisk.org/wiki/display/AST/SIP+TLS+Transport>

4.4. Trunking IAX2

- <https://www.voip-info.org/wiki/view/Asterisk+-+dual+servers>

5. Fonctionnalités diverses

5.1. Dire l'adresse IP avec la fonction curl

Chapitre 17 "Interactive Voice Response" p. 484

Au préalable installer Curl.

```
sudo apt-get install libcurl4-openssl-dev
```


ou

```
sudo yum -y install libcurl-devel
```

et tenter ceci dans `extensions.conf` :

```
; extensions.conf modifié

[appels-internes]
include=>adresse-ip

[adresse-ip]

exten => *764,1,Verbose(2, Utilise CURL pour obtenir son adresse IP de ipinfo.io)
    same => n,Set(CHANNEL(language)=fr)
    same => n,Answer()
    same => n,Set(MyIPAddressIs=${CURL(http://ipinfo.io/ip)})
    same => n,SayAlpha(${MyIPAddressIs})
    same => n,Hangup()
```

5.2. Fichiers d'exemples

On peut comprendre et envisager la mise en oeuvre de l'exemple de base situé dans les sources : `~/src/asterisk-complete/asterisk/current/configs/`

- basic-pbx
- Samples

5.3. Exploration de fonctionnalités plus avancées

- -10. **Deeper into the Dialplan**
- -11. Parking, Paging, and Conferencing
- -13. **Automatic Call Distribution (ACD) Queues**
- -14. **Device States**
- -17. Interactive Voice Response
- -19. Fax

Asterisk Avancé

- 1. Base de données
- 2. AGI / AMI
 - 2.1. Frameworks et librairies AGI
 - 2.2. Exemple AGI en bash
 - 2.3. Exemple AGI en node.js
 - 2.4. ARI
- 3. Backups d'Asterisk
- 4. Mise-à-jour d'Asterisk
- 5. Haute Disponibilité / Clustering
 - 5.1. The Definitive Guide
 - 5.2. Solutions HA
- 6. Text-to-speech / Speech recognition

The Definitive Guide, 4th Edition :

- -16. Relational Database Integration
- -18. External Services
- -20. Asterisk Manager Interface (AMI)
- -21. Asterisk Gateway Interface (AGI)
- -22. Clustering
- -24. System Monitoring and Logging
- -26. Security

1. Base de données

- The Definitive Guide, 4th Edition : 16. Relational Database Integration, pp. 417-478.
- <https://wiki.asterisk.org/wiki/display/AST/Getting+Asterisk+Connected+to+MySQL+via+ODBC>
- <https://wiki.asterisk.org/wiki/display/AST/Realtime+Database+Configuration>
- <https://wiki.asterisk.org/wiki/display/AST/SIP+Realtime%2C+MySQL+table+structure>

2. AGI / AMI

- <https://wiki.asterisk.org/wiki/plugins/gliffy/viewer.action?inline=false&pageId=29395573&attachmentId=29687931&name=AMI-ARI-AGI&lastPage=%2Fpages%2Fviewpage.action%3FpageId%3D29395573>
- <https://wiki.asterisk.org/wiki/display/AST/Interfaces>
- Bases de données et connectivité "Realtime"
- AGI : sorte de CGI Apache pour Asterisk : permet de s'interfacer entre le dialplan Asterisk et un programme externe.
- AMI : permet de gérer Asterisk dans un modèle client/serveur : contrôler le PBX, placer des appels, vérifier des mailbox, surveiller des channels et des queues en tant que commandes Asterisk.
- ARI : asynchronous API REST permettant de créer sa propre application voicemail.

2.1. Frameworks et librairies AGI

Name	Language	Website	Protocols
Adhearsion	Ruby	http://www.adhearsion.com/	AMI/FastAGI
Asterisk-Java	Java	https://blogs.reucon.com/asterisk-java/	AMI/FastAGI
PAGI	PHP	https://github.com/marcelog/PAGI	AGI
PHPAGI	PHP	http://phpagi.sourceforge.net/	AGI
Panoramisk	Python+AsyncIO	https://github.com/gawel/panoramisk	AMI/FastAGI
Pyst2	Python	https://github.com/rdegges/pyst2	AMI/AGI
StarPy	Python+Twisted	https://github.com/asterisk/starpy	AMI/FastAGI
Nanoagi	C++	http://sourceforge.net/projects/nanoagi/	AGI
AsterNET	.NET (C#/VB.net)	https://github.com/skrusty/AsterNET	AMI/FastAGI
Ding-dong	node.js	https://www.npmjs.com/package/ding-dong	AGI

Source : <https://wiki.asterisk.org/wiki/pages/viewpage.action?pageId=32375589>

2.2. Exemple AGI en bash

The Definitive Guide, 4th Edition, pages 583-584.

Création d'un fichier `/var/lib/asterisk/agi-bin/hello-world.sh`

```
#!/bin/bash
# Consume all variables sent by Asterisk
while read VAR && [ -n ${VAR} ] ; do : ; done
# Answer the call.
echo "ANSWER"
read RESPONSE
# Say the letters of "Hello World"
echo 'SAY ALPHA "Hello World" ""'
read RESPONSE
exit 0
```

Attribuer des permissions d'exécution sur le script :

```
chmod +x /var/lib/asterisk/agi-bin/hello-world.sh
```

Ajout d'un extension qui appelle le script dans dialplan (`extensions.conf`) :

```
include=>hello-world

[hello-world]
exten => 500,1,AGI(hello-world.sh)
```

Dans la console d'Asterisk :

```
dialplan reload
```

2.3. Exemple AGI en node.js

- <https://www.npmjs.com/package/ding-dong>

2.4. ARI

ARI : asynchronous API REST permettant de créer sa propre application voicemail.

- <https://wiki.asterisk.org/wiki/pages/viewpage.action?pageId=29395573>
- <https://github.com/asterisk/ari-examples>

3. Backups d'Asterisk

- <https://wiki.asterisk.org/wiki/display/AST/Asterisk+Backups>
- Par exemple : http://www.klaverstyn.com.au/david/wiki/index.php?title=Asterisk_Backup_Shell_Script

Cibles :

- Fichiers compilés et installés
- Dump des bases de données
- Fichiers de logs

En format d'archive compressée avec une planification Cron

4. Mise-à-jour d'Asterisk

- <https://wiki.asterisk.org/wiki/display/AST/Updating+or+Upgrading+Asterisk>

Re-compilation des sources.

5. Haute Disponibilité / Clustering

- Système
- Logiciel Asterisk
- Bases de données
- Etat des périphériques
- Lignes

5.1. The Definitive Guide

The Definitive Guide, 4th Edition :

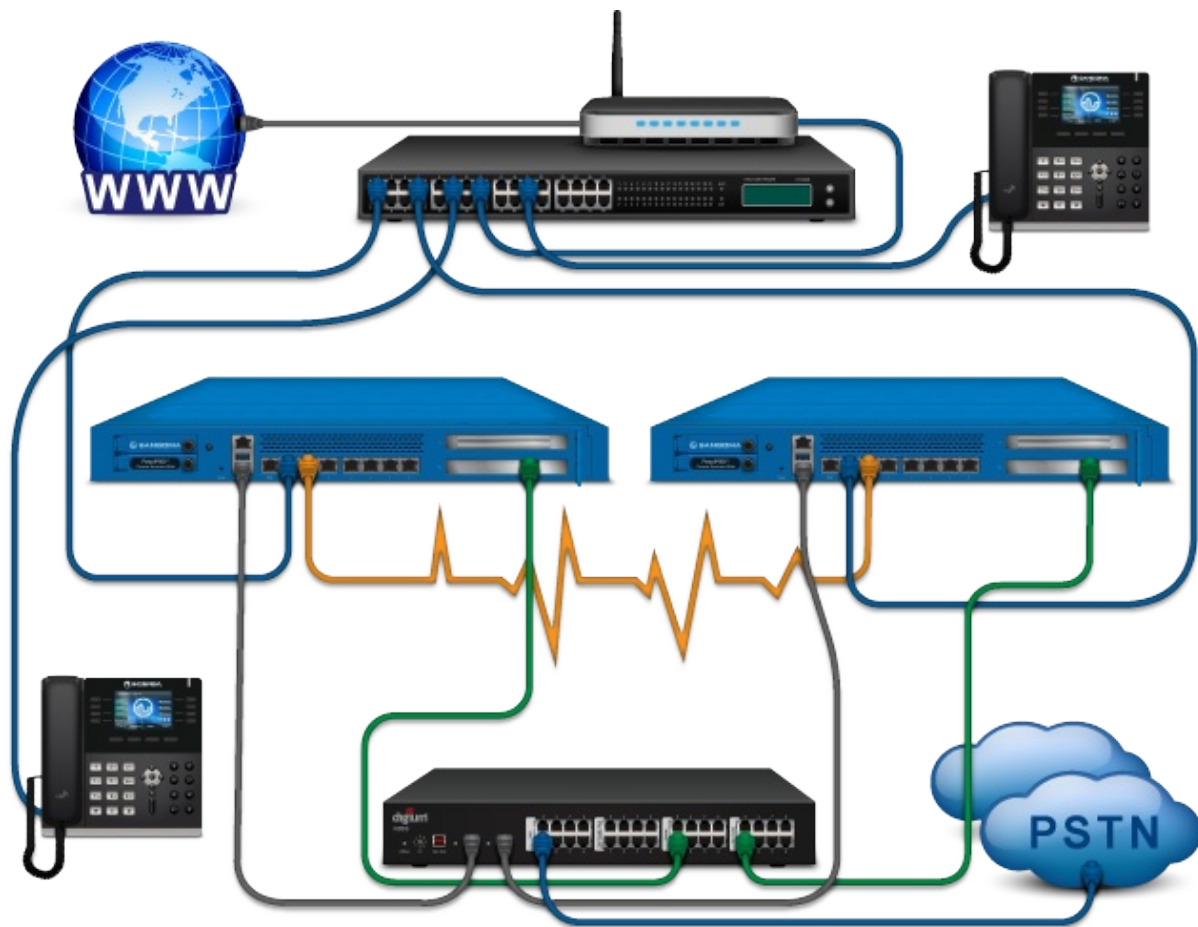
- Asterisk and Database Integration p. 606
- Single Database p. 606
- Replicated Databases p. 608
- Asterisk and Distributed Device States p. 609
- Distributing Device States over a LAN p. 610
- Distributing Device States over a WAN p. 611
- Multiple Queues, Multiple Sites p. 613

voir http://klaverstyn.com.au/david/wiki/index.php?title=Asterisk_Cluster_on_CentOS

5.2. Solutions HA

- <https://www.voip-info.org/wiki/view/Asterisk+High+Availability+Design>
- <https://www.voip-info.org/wiki/view/Asterisk+High+Availability+Solutions>

- <https://wiki.freepbx.org/display/FCM/FreePBX+High+Availability>



Source : <https://www.freepbx.org/high-availability/>

6. Text-to-speech / Speech recognition

texte-to-speech

- <https://www.blog-des-telecoms.com/text-to-speech-comment-utiliser-festival-espeak-et-googlelets-avec-asterisk-solution-de-telephonie-open-source/>
- <http://zaf.github.io/asterisk-googlelets/> : GoogleTTS text to speech script for asterisk
- <https://zaf.github.com/asterisk-googletranslate/> : Text translation using Google Translate API for Asterisk
- <https://zaf.github.io/asterisk-speech-recog/>: Speech recognition script for Asterisk that uses Cloud Speech API by Google.
- <https://zaf.github.com/asterisk-mstts/> : Speech synthesis using Microsoft Translator API for Asterisk
- <https://zaf.github.com/Asterisk-Flite/> : Asterisk Flite text to speech module
- <https://zaf.github.com/Asterisk-eSpeak/> : Asterisk e-Speak text to speech module