Travaux Pratiques de Réseaux

Ethereal

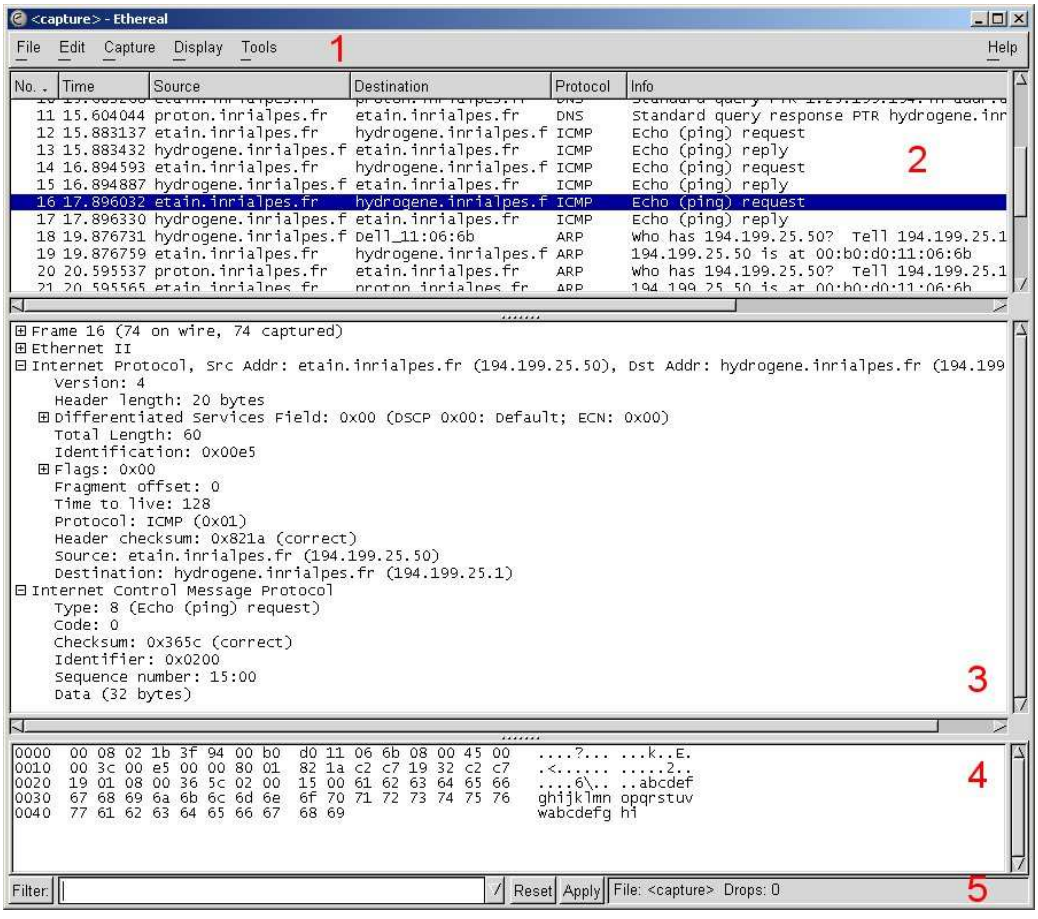
Avant propos

Cette documentation n'est absolument pas exhaustive. Elle a pour simple but de favoriser la prise en main de l'outil Ethereal dans le cadre des séances de travaux pratiques. La version que décrit cette notice est la 0.9.6. Les captures d'écrans proviennent d'une version Windows, cependant l'interface est relativement similaire dans la version Linux. Pour plus d'informations, il est idéal préférable de consulter la page officielle d'Ethereal (www.ethereal.com).

Qu'est ce qu'Ethereal ?

Ethereal est un outil, que l'on surnomme habituellement *sniffer*, dont le but est de tracer des communications réseaux. Ethereal a la particularité de permettre de tracer en ligne (i.e. en interrogeant les interfaces réseaux) mais également hors ligne (i.e. en rejouant des fichiers de capture de divers formats couramment utilisés).

L'interface d'Ethereal



L'interface d'Ethereal est composée de 5 principaux éléments :

- une barre de menus (**1**)
- une zone de définition de filtres et d'affichage d'état de l'application (**5**)
- trois zones d'affichage de paquets
  - une zone principale (**2**) permettant de voir l'ensemble des paquets capturés (et ayant passé les filtres). Chacune des lignes représente un paquet auquel sont associés :
    - § un numéro
    - § une date de capture
    - § l'adresse de la source et de la destination (les formats d'adresses sont relatifs aux protocoles utilisés)
    - § le protocole utilisé et la nature du paquet (Ethereal détecte le protocole et « décode », i.e. interprète, le paquet si l'option correspondante a été validée)
  - une zone de « zoom » (**3**) permettant d'avoir (sous forme arborescente) plus d'informations sur le ou les paquets selectionnés dans la zone supérieure.
  - Une zone de « données » (**4**) permettant de visualiser, en ASCII et hexadécimal, le contenu du paquet selectionné considéré.

Les menus d'Ethereal

Fichier (File) : Permet notamment le chargement de fichiers de capture (mode hors ligne) et l'impression de contenu de paquets.

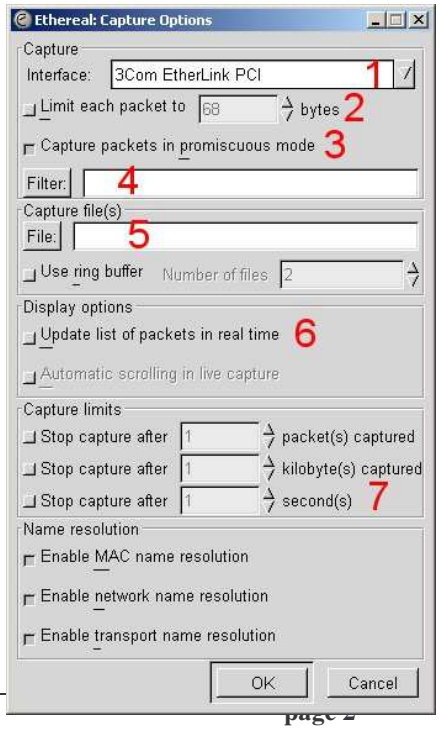
Edition (Edit) : Facilite la navigation dans la zone de paquets (aller au paquet n° , …) et permet de contrôler le décodage de protocoles

Capture : Permet le démarrage de capture (cf. section suivante)

Affichage (Display) : Permet le contrôle fin de l'affichage d'informations sur les paquets

Outils (Tools) : Propose quelques outils très utiles tels que le suivi de flux TCP (cf sections suivantes)

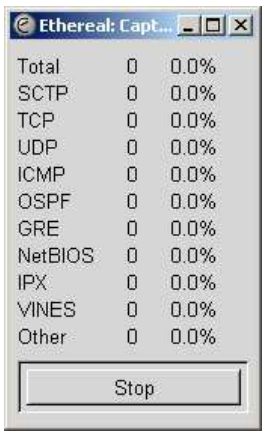
Démarrage d'une capture en ligne



Le démarrage d'une capture s'effectue au travers du menu **Capture > Start**. L'utilisateur se trouve alors confronté à une boite de dialogue permettant de configurer la capture (voir figure ci-contre).

Le minimum est au moins de sélectionner l'interface réseau sur laquelle on souhaite travailler (**1**). Il est également possible de limiter la taille des paquets (**2**) et de positionner l'interface dans le mode *promiscuous* (**3**). Cette dernière option est intéressante dans la mesure où

ce mode permet d'avoir accès à tous les paquets reçus par l'interface et non uniquement à ceux que celle-ci juge bon de faire remonter vers le système (i.e. les paquets en loopback, ceux associés à l'adresse MAC de l'interface, les paquets en broadcast et en multicast). Il est alors possible de sniffer les communications entre deux autres machines du réseau local[1]. Les paquets capturés peuvent être archivés dans un fichier (**4**). Les paquets peuvent également être filtrés (**5**) par rapport à des critères s'exprimant dans une syntaxe commune avec l'outil TCPDump (voir sections suivantes). Il est aussi possible d'opter ou non pour un affichage « temps réel » des paquets capturés (**6**). Enfin, il est possible de donner une limite (nombre de paquets, durée, …) permettant de stopper la capture (**7**).



Lors de la capture, il est possible de visualiser, en plus des informations situées dans les zones d'affichages décrites précédemment, des statistiques sur le nombre et la nature des paquets capturés (voir figure à droite).

Les filtres

Filtres de capture

Au démarrage d'une capture, il est tout d'abord possible de préciser quels paquets doivent être capturés (voir figure page 6, en haut à droite, zone 4). On parle alors de filtre de capture. Leur syntaxe est identique à celle utilisé par l'outil TCPDump (pour plus de renseignements il faut donc se reporter à la page de manuel correspondante).

Un filtre est une combinaison logique (**or**, **and**, **not**) de primitives. Une primitive peut être un protocole (**tcp**, **ip**, **arp** …) ce qui signifie que seuls les paquets pour lesquels le protocole est utilisé seront capturés. Une primitive peut être un qualificateur parmi **host**, **net** et **port** permettant de specifier respectivement un identifiant d'hôte, de réseau ou de port. Ces qualificateurs peuvent être affinés par l'utilisation d'identifiants de direction **src** et **dst** (il est possible de combiner les deux, *src and dst host 172.26.7.62*).

Exemples :

host 172.26.7.62, ne capture que les paquets de ou à destination de la machine d'IP 172.26.7.62
host 172.26.7.62 and tcp and port 21, idem mais restreint aux connexions ftp

Filtres d'affichage

Il est également possible en cours de capture de restreindre l'affichage à un sous-ensemble de paquets (voir figure en bas de la page 1, zone 5, « apply ») sachant qu'il est tout à fait possible de revenir à un affichage complet en supprimant le filtre ainsi défini (voir même figure, même zone, « reset »). On parle alors de filtres d'affichage. Leur syntaxe n'est pas la même que les précédents.

[1] on notera au passage les limites d'Ethereal lorsque ces machines sont reliées non pas par un répéteur mais par un switch.

Un filtre est un ensemble de conditions (parenthèsées) reliées par des operateurs logiques (**and**, **or**, **not, xor**). Une condition est exprimée à l'aide d'un opérateur de comparaison (voir le tableau ci-contre) et d'un attribut typé lié à un protocole décodable par ethereal. L'opérateur de comparaison peut être exprimé avec l'une ou l'autre des deux syntaxes disponibles.

Exemple de filtre :
(ip.addr eq 194.199.25.50) and (tcp.port eq 110)

Syntaxe1	Syntaxe 2	Description and example
eq	==	**Egalité**
ne	!=	**Non-egalité**
gt	>	**Supériorité**
lt	<	**Infériorité**
ge	>=	`frame.pkt_len ge 0x100`
le	<=	`frame.pkt_len <= 0x20`

Les tableaux des pages suivantes présentent les attibuts disponibles pour les protocoles les plus courants.

Attributs IP	Description	Type
ip.addr	Source or Destination Address	IPv4 address
ip.checksum	Header checksum	Unsigned 16-bit integer
ip.checksum_bad	Bad Header checksum	Boolean
ip.dst	Destination	IPv4 address
ip.flags	Flags	Unsigned 8-bit integer
ip.flags.df	Don't fragment	Boolean
ip.flags.mf	More fragments	Boolean
ip.frag_offset	Fragment offset	Unsigned 16-bit integer
ip.fragment	IP Fragment	No value
ip.fragment.error	Defragmentation error	No value
ip.fragment.multipletails	Multiple tail fragments found	Boolean
ip.fragment.overlap	Fragment overlap	Boolean
ip.fragment.overlap.conflict	Conflicting data in fragment overlap	Boolean
ip.fragment.toolongfragment	Fragment too long	Boolean
ip.fragments	IP Fragments	No value
ip.hdr_len	Header Length	Unsigned 8-bit integer
ip.id	Identification	Unsigned 16-bit integer
ip.len	Total Length	Unsigned 16-bit integer
ip.proto	Protocol	Unsigned 8-bit integer
ip.src	Source	IPv4 address
ip.tos	Type of Service	Unsigned 8-bit integer
ip.tos.cost	Cost	Boolean
ip.tos.delay	Delay	Boolean
ip.tos.precedence	Precedence	Unsigned 8-bit integer
ip.tos.reliability	Reliability	Boolean
ip.tos.throughput	Throughput	Boolean
ip.ttl	Time to live	Unsigned 8-bit integer
ip.version	Version	Unsigned 8-bit integer

Attributs TCP	Description	Type
tcp.ack	Acknowledgement number	Unsigned 32-bit integer
tcp.checksum	Checksum	Unsigned 16-bit integer
tcp.checksum_bad	Bad Checksum	Boolean
tcp.dstport	Destination Port	Unsigned 16-bit integer
tcp.flags	Flags	Unsigned 8-bit integer
tcp.flags.ack	Acknowledgment	Boolean
tcp.flags.cwr	Congestion Window Reduced (CWR)	Boolean
tcp.flags.ecn	ECN-Echo	Boolean
tcp.flags.fin	Fin	Boolean
tcp.flags.push	Push	Boolean
tcp.flags.reset	Reset	Boolean
tcp.flags.syn	Syn	Boolean
tcp.flags.urg	Urgent	Boolean
tcp.hdr_len	Header Length	Unsigned 8-bit integer
tcp.nxtseq	Next sequence number	Unsigned 32-bit integer
tcp.port	Source or Destination Port	Unsigned 16-bit integer
tcp.seq	Sequence number	Unsigned 32-bit integer
tcp.srcport	Source Port	Unsigned 16-bit integer
tcp.urgent_pointer	Urgent pointer	Unsigned 16-bit integer
tcp.window_size	Window size	Unsigned 16-bit integer

Attributs ARP	Description	Type
arp.dst.atm_num_e164	Target ATM number (E.164)	String
arp.dst.atm_num_nsap	Target ATM number (NSAP)	Byte array
arp.dst.atm_subaddr	Target ATM subaddress	Byte array
arp.dst.hlen	Target ATM number length	Unsigned 8-bit integer
arp.dst.htype	Target ATM number type	Boolean
arp.dst.hw	Target hardware address	Byte array
arp.dst.pln	Target protocol size	Unsigned 8-bit integer
arp.dst.proto	Target protocol address	Byte array
arp.dst.slen	Target ATM subaddress length	Unsigned 8-bit integer
arp.dst.stype	Target ATM subaddress type	Boolean
arp.hw.size	Hardware size	Unsigned 8-bit integer
arp.hw.type	Hardware type	Unsigned 16-bit integer
arp.opcode	Opcode	Unsigned 16-bit integer
arp.proto.size	Protocol size	Unsigned 8-bit integer
arp.proto.type	Protocol type	Unsigned 16-bit integer
arp.src.atm_num_e164	Sender ATM number (E.164)	String
arp.src.atm_num_nsap	Sender ATM number (NSAP)	Byte array
arp.src.atm_subaddr	Sender ATM subaddress	Byte array
arp.src.hlen	Sender ATM number length	Unsigned 8-bit integer
arp.src.htype	Sender ATM number type	Boolean
arp.src.hw	Sender hardware address	Byte array
arp.src.pln	Sender protocol size	Unsigned 8-bit integer
arp.src.proto	Sender protocol address	Byte array
arp.src.slen	Sender ATM subaddress length	Unsigned 8-bit integer
arp.src.stype	Sender ATM subaddress type	Boolean

Suivi de flux TCP

Le suivi de flux TCP se révèle être un outil particulièrement puissant et pratique dans la mesure où, à partir d'une trame TCP sélectionnée, il permet de visualiser sous forme ASCII ou hexadécimale l'ensemble des échanges ultérieurs entre le client et le serveur.

L'exemple ci-dessous présente l'affichage résultant de l'utilisation de cet outil dans le cadre d'une connexion entre un client et un serveur POP (les messages émis par le client sont en rouge, ceux émis par le serveur sont en bleu).