M3105 : Services Réseaux Avancés / Supervision de Réseau

Guillaume Urvoy-Keller

September 12, 2019

Objectif

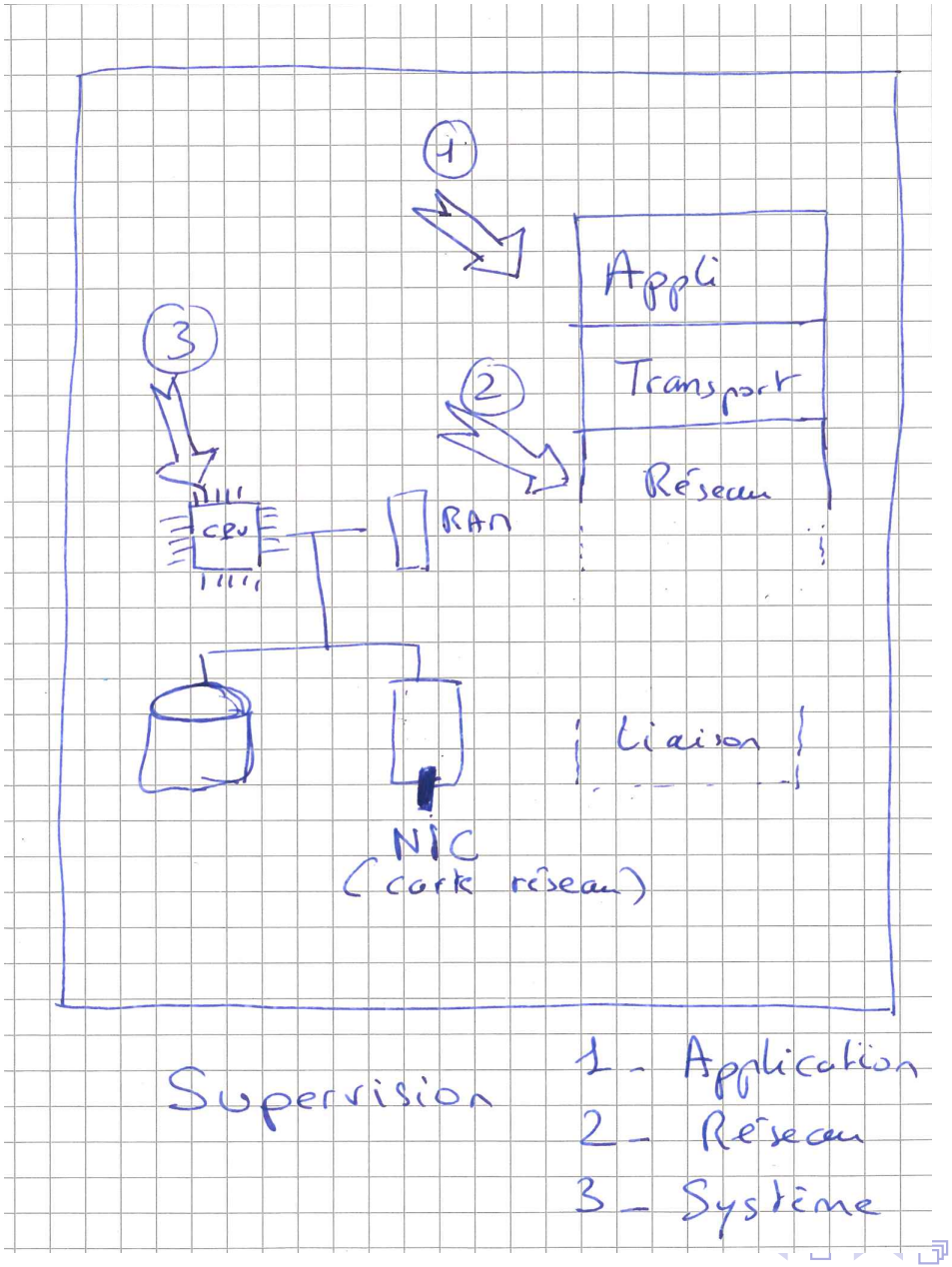
Supervision

- Réseau: Wireshark, ping, traceroute, smokeping
- Système/Application : SNMP, collectd, netdata, syslog, ELK

Installation Services

- Services spécifiques : DNS, LDAP
- Automatisation : Ansible

Supervision : Niveau Application / Réseau ou Système



Organisation du cours

CM/Cours

- Intro
- Supervision réseau: 1.passif, 2.actif
- Supervision système
- Configurations Serveurs (Ansible)
- Services :
  - DNS
  - LDAP

TD/TP

- TD analyse de trafic (perf, sécu) item TD Smokeping : analyse de mesures actives
- TP Supervision Réseau/Système
- TP Ansible
- TD DNS et LDAP
- TP DNS
- TP LDAP

Organisation du cours

- Notation : 30 % TP et 70 % examen final
- Examen final comporte :
  - des questions de cours
  - **des questions sur les TPs** $\Rightarrow$ vous devez faire un compte-rendu électronique pour vous (+ pour l'encadrant suivant ses demandes)

M3105 - Supervision Réseau

Guillaume Urvoy-Keller

September 11, 2018

Cas 1 : un serveur physique hébergeant un serveur Web

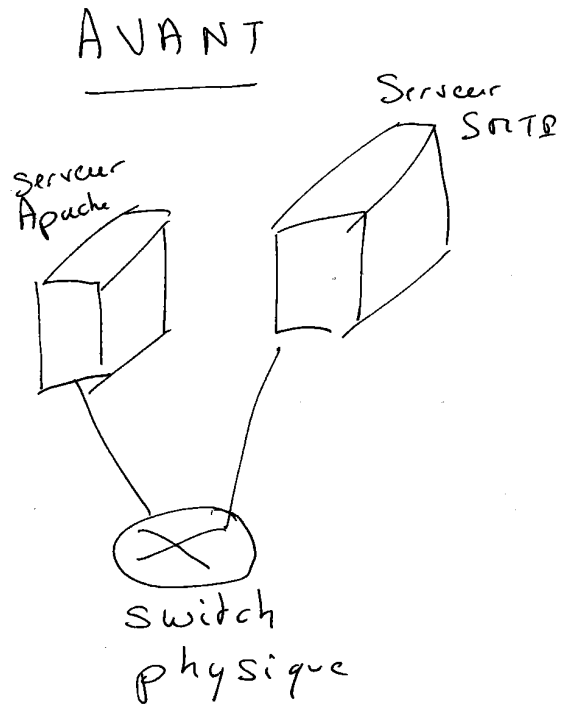
- Que faut-il monitorer?
- Le système :
  - RAM, CPU, I/O (entrées/sorties disques), réseau
  - Logs système (/var/log/syslog)
- L'application : Nombre de requêtes par seconde? Nombre d'objets non trouvés?
- Logs applicatifs (/var/log/apache2/access.log ou error.log)

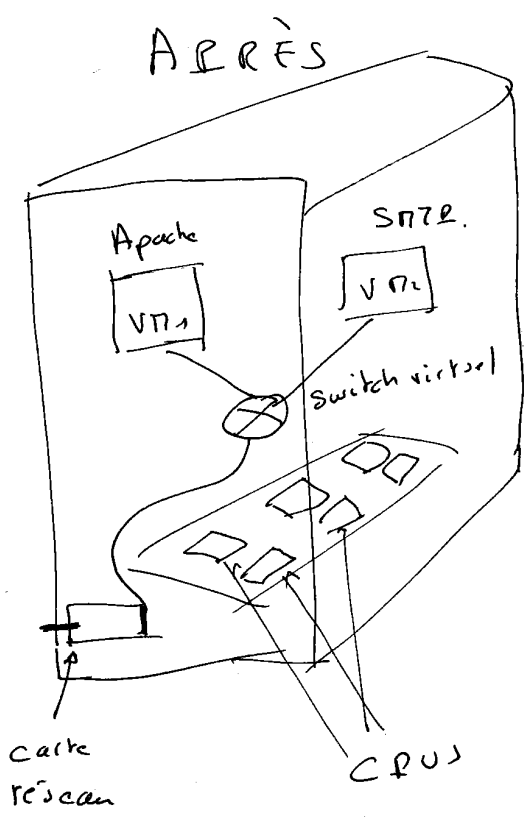
Cas 2 : un serveur virtuel (VM) hébergeant un serveur Web

- Que faut-il monitorer?
- La machine hôte?
- La machine virtuelle?
- Le réseau
  - Frontière réseau/système floue avec la virtualisation (voir transparent suivant)

Frontière Réseau/Système

- Le réseau débute dans les serveurs avec des switchs, des règles de routage/NAT
- Ces équipements réseaux virtuels partagent la puissance de calculs (coeurs/CPUs) avec les services hébergés dans VMs ou containers





Supervision Réseau

Monitoring Réseau

Passif

- Sur un serveur : Wireshark, tshark, tcpdump
- Sur un équipement réseau : Netflow, sflow, IPFIX

Actif

- Ping, traceroute
- Smokeping

Monitoring passif : cas des serveurs

Monitoring réseau sur un serveur

On peut vouloir :

- Connaitre le trafic entrant et sortant sur les différentes interfaces
- Connaitre les connexions réseaux actives
- Capturer le trafic
- Naviguer dans le trafic capturé

Différence importante entre un serveur et un switch/routeur

Le switch routeur n'a pas de disque!!

Trafic entrant et sortant

Beaucoup d'outils vont donner des statistiques sur les interfaces : netstat, ip, sar...

```
root@stretch:~# netstat -i
Kernel Interface table
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500    95556      0      0 0         23194      0      0      0 BMRU
eth1      1500   197909      0      0 0         20706      0      0      0 BMRU
lo       65536    38203      0      0 0         38203      0      0      0 LRU
```

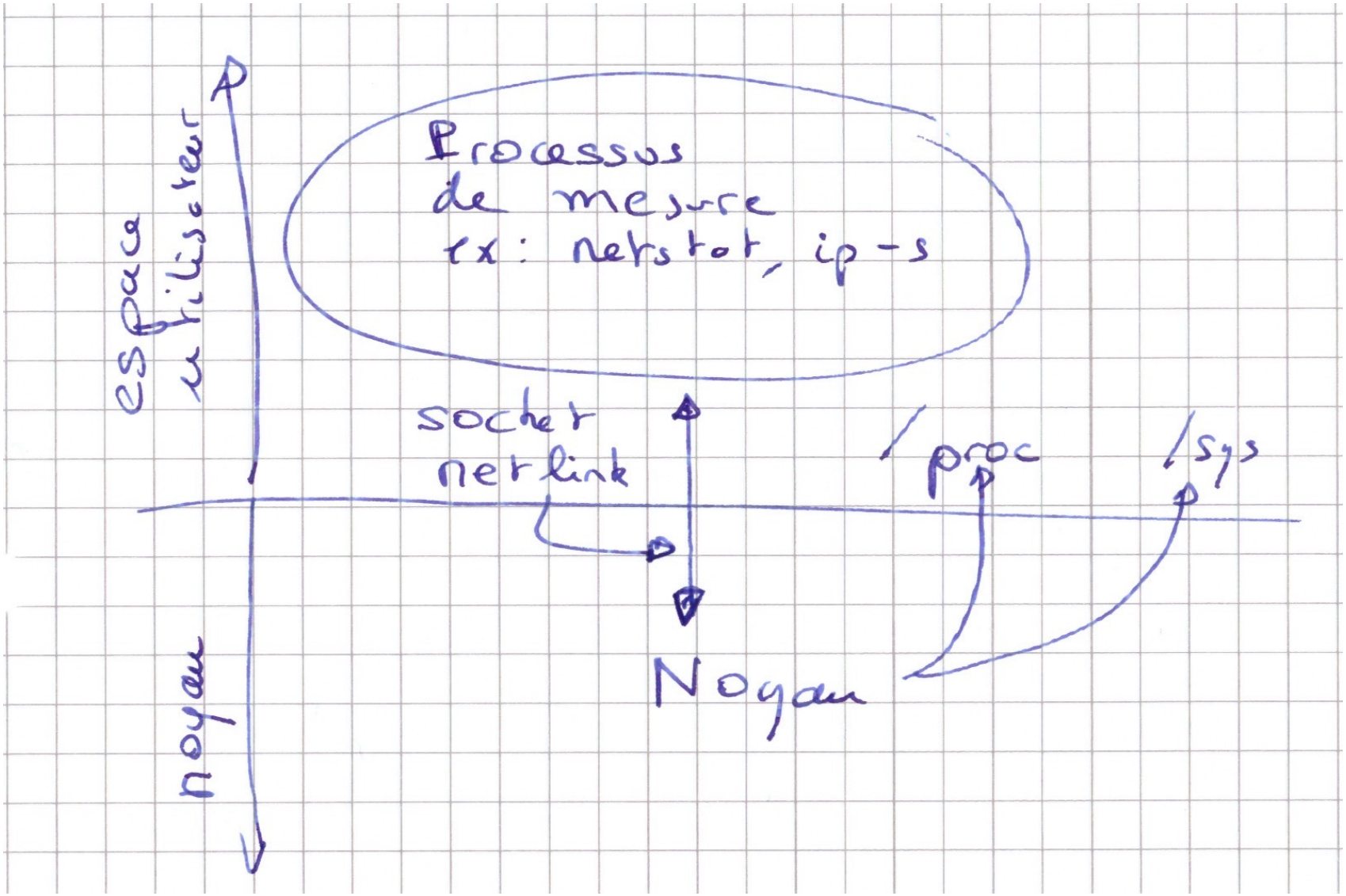
```
root@stretch:~# ip -s link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    RX: bytes  packets  errors  dropped overrun mcast
    36797553   38229    0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    36797553   38229    0       0       0       0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:8d:c0:4d brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    95784310   95593    0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    1973806    23217    0       0       0       0
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:18:6d:37 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    160715547  197979   0       0       0       11413
    TX: bytes  packets  errors  dropped carrier collsns
    3061833    20757    0       0       0       0
```

Trafic entrant et sortant

Ces outils vont chercher leurs informations dans ce que le noyau expose:

1. Soit au travers de système de fichiers virtuels (en mémoire)
   - /proc et /sys en Unix/Linux
   - Philosophie unix : tout est fichier
2. Soit au travers d'un moyen de communication avec le noyau
   - Exemple : socket NETLINK en Linux

Récupération information depuis le noyau



Trafic entrant et sortant

Option 1 : lecture des données exposées par le noyau dans un système de fichier

On peut le voir en utilisant l'outil *strace* qui suit les appels système d'un processus et en regardant les fichiers ouverts.

Example avec netstat:

```
root@stretch:~# strace -o fichier_trace netstat -i
Kernel Interface table
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0      1500    95691      0      0 0         23269      0      0      0 BMRU
eth1      1500   198360      0      0 0         20928      0      0      0 BMRU
lo       65536    38297      0      0 0         38297      0      0      0 LRU
root@stretch:~# grep ^open fichier_trace
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
open("/lib/x86_64-linux-gnu/libselinux.so.1", O_RDONLY|O_CLOEXEC) = 3
open("/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
open("/lib/x86_64-linux-gnu/libpcre.so.3", O_RDONLY|O_CLOEXEC) = 3
open("/lib/x86_64-linux-gnu/libdl.so.2", O_RDONLY|O_CLOEXEC) = 3
open("/lib/x86_64-linux-gnu/libpthread.so.0", O_RDONLY|O_CLOEXEC) = 3
open("/proc/filesystems", O_RDONLY)     = 3
open("/usr/lib/locale/locale-archive", O_RDONLY|O_CLOEXEC) = 3
open("/usr/share/locale/locale.alias", O_RDONLY|O_CLOEXEC) = 6
open("/usr/share/locale/en_US.UTF-8/LC_MESSAGES/net-tools.mo", O_RDONLY) = -1 ENOENT (No such file or directory)
open("/usr/share/locale/en_US.utf8/LC_MESSAGES/net-tools.mo", O_RDONLY) = -1 ENOENT (No such file or directory)
open("/usr/share/locale/en_US/LC_MESSAGES/net-tools.mo", O_RDONLY) = -1 ENOENT (No such file or directory)
open("/usr/share/locale/en.UTF-8/LC_MESSAGES/net-tools.mo", O_RDONLY) = -1 ENOENT (No such file or directory)
open("/usr/share/locale/en.utf8/LC_MESSAGES/net-tools.mo", O_RDONLY) = -1 ENOENT (No such file or directory)
open("/usr/share/locale/en/LC_MESSAGES/net-tools.mo", O_RDONLY) = -1 ENOENT (No such file or directory)
open("/proc/net/dev", O_RDONLY)         = 6
```

Trafic entrant et sortant

Option 2 : récupération directement au travers de socket spéciales appelées socket NETFILTER

Example avec ip:

```
root@stretch:/home/vagrant# strace -o fichier_trace ip -s -i link ^C
root@stretch:/home/vagrant# grep NETLINK fichier_trace
socket(AF_NETLINK, SOCK_RAW|SOCK_CLOEXEC, NETLINK_ROUTE) = 3
bind(3, {sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000}, 12) = 0
getsockname(3, {sa_family=AF_NETLINK, nl_pid=4628, nl_groups=00000000}, [12]) = 0
recvmsg(3, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000}, msg_namelen=12, msg_iov=[{iov_base=[{{len=1232, type=0x10 /* NLMSG_
=NLM_F_MULTI, seq=1531235664, pid=4628}, "\0\0\4\3\1\0\0\0I\0\1\0\0\0\0\0\7\0\3\0lo\0\0\10\0\r\0\1\0\0\0"...}, {{len=1248, type=0x10 /* NLMSG_
=NLM_F_MULTI, seq=1531235664, pid=4628}, "\0\0\1\0\2\0\0\0C\20\1\0\0\0\0\0\t\0\3\0eth0\0\0\0\0\10\0\r\0"...}, {{len=1248, type=0x10 /* NLMSG_?
NLM_F_MULTI, seq=1531235664, pid=4628}, "\0\0\1\0\3\0\0\0C\20\1\0\0\0\0\0\t\0\3\0eth1\0\0\0\0\10\0\r\0"...}, {{len=0, type=0 /* NLMSG_??? */,
0, pid=0}}], iov_len=32768}], msg_iovlen=1, msg_controllen=0, msg_flags=0}, 0) = 3728
recvmsg(3, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000}, msg_namelen=12, msg_iov=[{iov_base=[{{len=20, type=NLMSG_DONE, flag
, seq=1531235664, pid=4628}, "\0\0\0\0"}, {{len=1, type=0x49 /* NLMSG_??? */, flags=NLM_F_REQUEST, seq=0, pid=196615}}], iov_len=32768}], msg_
_controllen=0, msg_flags=0}, 0) = 20
```

Connexions actives

- Les infos sont récupérées du noyau à nouveau.
- On distingue entre les types de sockets :
  1. TCP
  2. UDP
  3. Unix : entre processus dans la machine
  4. NETLINK : entre un processus et le noyau

  mais également :
  1. Socket client : connexion initiée vers la machine
  2. Socket serveur : en mode listen

Connexions actives

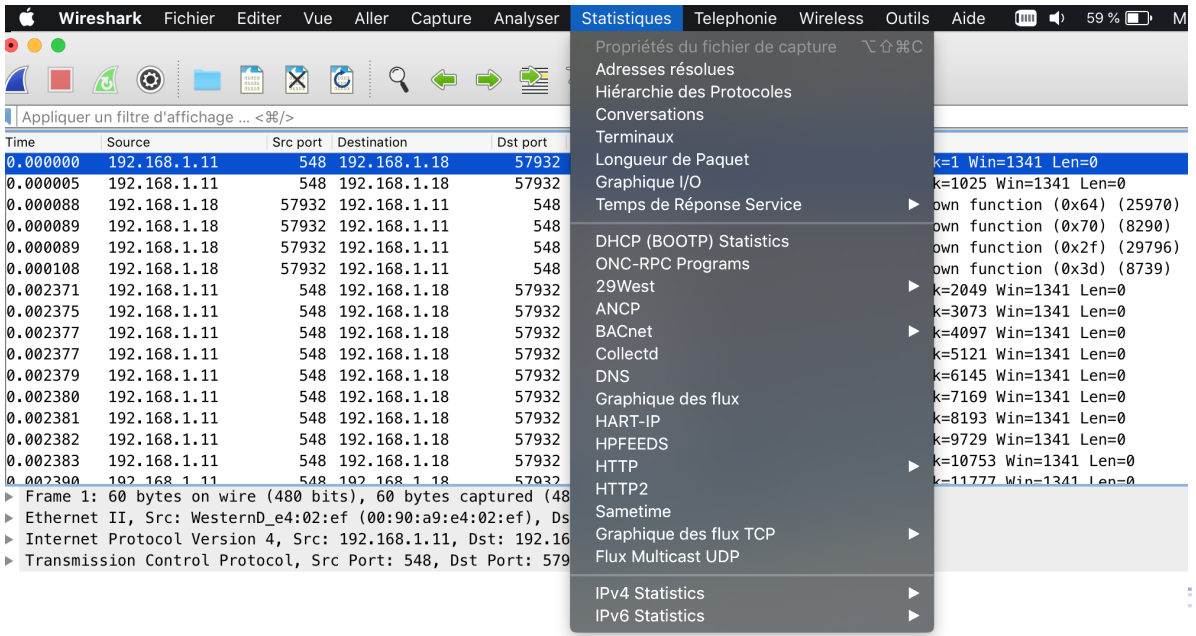
```
root@stretch:/home/vagrant# netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN      => serveur
tcp        0      0 10.0.2.15:ssh           10.0.2.2:54207          ESTABLISHED => cx ssh
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 [::]:http-alt           [::]:*                  LISTEN
tcp6       0      0 [::]:tproxy             [::]:*                  LISTEN
tcp6       0      0 [::]:8082               [::]:*                  LISTEN
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    11810    /run/user/1000/systemd/notify
unix  2      [ ACC ]     STREAM     LISTENING     11814    /run/user/1000/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     11819    /run/user/1000/gnupg/S.gpg-agent.ss
unix  2      [ ACC ]     STREAM     LISTENING     12634    /run/user/1000/gnupg/S.gpg-agent.br
unix  2      [ ACC ]     STREAM     LISTENING     12636    /run/user/1000/gnupg/S.gpg-agent
unix  2      [ ACC ]     STREAM     LISTENING     12638    /run/user/1000/gnupg/S.gpg-agent.ex
unix  2      [ ACC ]     STREAM     LISTENING     11430    /var/run/dbus/system_bus_socket
```

Capturer le trafic

- Wireshark
- tcpdump, tshark: capture en ligne de commande
- tcptrace: analyse basique d'une trace

Wireshark

- Permet de capturer le trafic
- Permet aussi de l'analyser (voir TD sur wireshark).
- Différents scénarios d'analyse :
  1. Vous cherchez une connexion précise et un trafic précis : les filtres wireshark vous aident.
  2. Vous cherchez à comprendre le trafic qui circule sur le lien : par où commencer? Regarder les paquets l'un après l'autre? Non, il faut commencer par avoir des **résumés** de la trace : protocoles présents, connexions présentes...



Wireshark

Wireshark permet :

- **Niveau protocole :** Quel est le nombre de paquets/octets de chaque protocole, par exemple Ethernet ou IPv4,IPv6, ou TCP/UDP..ou DNS, HTTP, etc..?
- **Niveau connexion :** Combien de connexions TCP et de conversations UDP?
  Rq: on dit conversation pour UDP pour le pas dire connexion car il n'y a pas d'établissement de connexion comme en TCP.

Old school : tcptrace

Un simple résumé de la trace en terme de connexion : 1 grande, 24 petites

```
guillaumesmbp2:tmp urvoy$ tcptrace -b trace.pcap
1 arg remaining, starting with 'trace.pcap'
Ostermann's tcptrace -- version 6.6.7 -- Thu Nov  4, 2004

56410 packets seen, 56406 TCP packets traced
elapsed wallclock time: 0:00:00.095494, 590717 pkts/sec analyzed
trace file elapsed time: 0:00:06.710829
TCP connection info:
  1: wdmycloud-10.home:548 - guillaumesmbp2.home:57932 (a2b)                                  18420> 37881<
  2: wr-in-f189.1e100.net:443 - guillaumesmbp2.home:59361 (c2d)                                 10>   10<
  3: guillaumesmbp2.home:60407 - server-52-85-23-190.mrs50.r.cloudfront.net:443 (e2f)            1>    1<
  4: guillaumesmbp2.home:60405 - server-52-85-23-190.mrs50.r.cloudfront.net:443 (g2h)            1>    1<
  5: guillaumesmbp2.home:60397 - server-52-85-23-190.mrs50.r.cloudfront.net:443 (i2j)            1>    1<
  6: guillaumesmbp2.home:60416 - a92-123-100-32.deploy.static.akamaitechnologies.com:443 (k2l)   1>    1<
  7: guillaumesmbp2.home:59362 - wr-in-f189.1e100.net:443 (m2n)                                  6>    6<
  8: guillaumesmbp2.home:60425 - a92-123-100-32.deploy.static.akamaitechnologies.com:443 (o2p)   1>    1<
  9: guillaumesmbp2.home:60424 - a92-123-100-32.deploy.static.akamaitechnologies.com:443 (q2r)   1>    1<
 10: guillaumesmbp2.home:60423 - a92-123-100-32.deploy.static.akamaitechnologies.com:443 (s2t)   1>    1<
 11: guillaumesmbp2.home:60427 - 151.101.121.108:443 (u2v)                                       1>    1<
 12: guillaumesmbp2.home:60408 - server-52-85-23-190.mrs50.r.cloudfront.net:443 (w2x)            1>    1<
 13: guillaumesmbp2.home:60460 - ip214.ip-151-80-95.eu:443 (y2z)                                 9>    6<
 14: guillaumesmbp2.home:60389 - par10s27-in-f202.1e100.net:443 (aa2ab)                          2>    2<
 15: guillaumesmbp2.home:60388 - 151.101.122.217:443 (ac2ad)                                     2>    1<
 16: guillaumesmbp2.home:60398 - 151.101.122.109:443 (ae2af)                                     2>    1<
 17: guillaumesmbp2.home:60392 - 104.25.107.104:443 (ag2ah)                                      2>    1<
 18: guillaumesmbp2.home:60390 - par10s34-in-f8.1e100.net:443 (ai2aj)                            2>    2<
 19: guillaumesmbp2.home:60399 - a2-18-139-190.deploy.static.akamaitechnologies.com:443 (ak2al) 2>    1<
 20: guillaumesmbp2.home:60401 - 151.101.120.188:443 (am2an)                                     2>    1<
 21: guillaumesmbp2.home:60406 - 104.19.195.151:443 (ao2ap)                                      2>    2<
 22: guillaumesmbp2.home:60366 - 184.220.186.35.bc.googleusercontent.com:443 (aq2ar)             4>    1<
 23: guillaumesmbp2.home:60433 - server-52-85-22-22.mrs50.r.cloudfront.net:443 (as2at)           1>    1<
 24: 13.94.211.113:443 - guillaumesmbp2.home:57885 (au2av)                                       2>    2<
 25: guillaumesmbp2.home:60442 - ec2-52-210-246-27.eu-west-1.compute.amazonaws.com:443 (aw2ax)   1>    1<
```

Old school : tcptrace

Analyse détaillée grande connexion TCP : back-up time machine. Pas de retransmission, semble ok...

```
guillaumesmbp2:tmp urvoy$ tcptrace -l trace.pcap
1 arg remaining, starting with 'trace.pcap'
Ostermann's tcptrace -- version 6.6.7 -- Thu Nov  4, 2004

56410 packets seen, 56406 TCP packets traced
elapsed wallclock time: 0:00:02.325609, 24256 pkts/sec analyzed
trace file elapsed time: 0:00:06.710829
TCP connection info:
25 TCP connections traced:
TCP connection 1:
        host a:        wdmycloud-10.home:548
        host b:        guillaumesmbp2.home:57932
        complete conn: no       (SYNs: 0)  (FINs: 0)
        first packet:  Wed Jul 11 08:38:32.374999 2018
        last packet:   Wed Jul 11 08:38:39.085829 2018
        elapsed time:  0:00:06.710829
        total packets: 56301
        filename:      trace.pcap
   a->b:                                b->a:
     total packets:         18420              total packets:         37881
     ack pkts sent:         18420              ack pkts sent:         37881
     pure acks sent:        18339              pure acks sent:           27
     sack pkts sent:            0              sack pkts sent:            0
     dsack pkts sent:           0              dsack pkts sent:           0
     max sack blks/ack:         0              max sack blks/ack:         0
     unique bytes sent:      1920              unique bytes sent:  19361660
     actual data pkts:         81              actual data pkts:      37854
     actual data bytes:      1920              actual data bytes:  19361660
     rexmt data pkts:           0              rexmt data pkts:           0
     rexmt data bytes:          0              rexmt data bytes:          0
     zwnd probe pkts:           0              zwnd probe pkts:           0
     zwnd probe bytes:          0              zwnd probe bytes:          0
     outoforder pkts:           0              outoforder pkts:           0
     pushed data pkts:         81              pushed data pkts:         49
```

Filtrer le trafic

Comment s'effectue la capture de paquets en wireshark/tcpdump?

sachant que :

1. Votre programme tourne en mode utilisateur
2. Les paquets sont gérés par le noyau

$\Rightarrow$ il faut traverser la frontière espace utilisateur / espace noyau

Capturer et filtrer en Unix : BPF (Berkerley Packet Filtering)

- Quand BPF écoute une interface réseau, le driver donne une copie du paquet à BPF avant de la donner au système.
- Le filtrage se fait dans le noyau
  - $\rightarrow$ évite de transférer tous les paquets (frontière espace utilisateur / espace noyau) puis de filtrer
  - Important dans un environnement de production où la mesure ne doit pas perturber le trafic de production (Ex : une carte à 10 Gb/s sur laquelle vous ne voulez filtrer qu'un seul de trafic...)

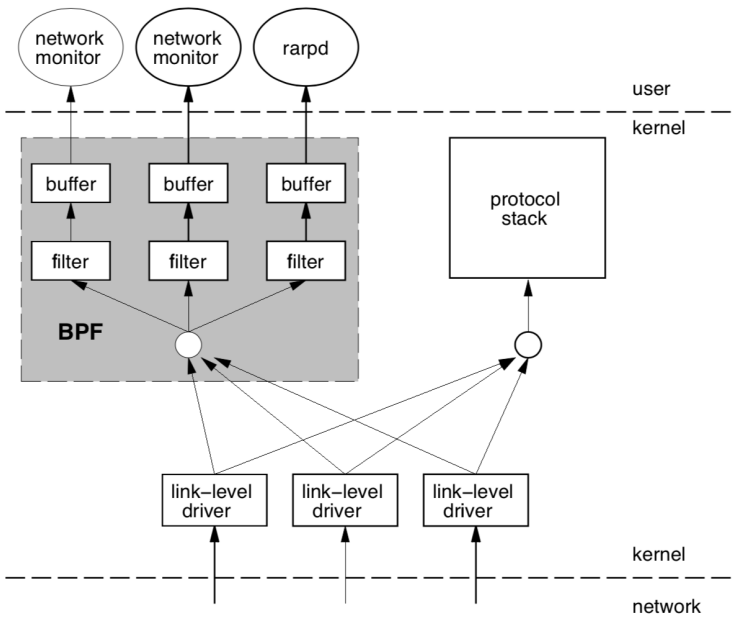


Figure 1: BPF Overview

Capturer et filtrer en Unix : BPF (Berkerley Packet Filtering)

BPF prend le filtre donné en entrée de tcpdump et l'exécute **dans** le noyau.

```
guillaumesmbp2:tmp urvoy$ tcptrace -l trace.pcap
1 arg remaining, starting with 'trace.pcap'
Ostermann's tcptrace -- version 6.6.7 -- Thu Nov  4, 2004

56410 packets seen, 56406 TCP packets traced
elapsed wallclock time: 0:00:02.325609, 24256 pkts/sec analyzed
trace file elapsed time: 0:00:06.710829
TCP connection info:
25 TCP connections traced:
TCP connection 1:
        host a:        wdmycloud-10.home:548
        host b:        guillaumesmbp2.home:57932
        complete conn: no       (SYNs: 0)  (FINs: 0)
        first packet:  Wed Jul 11 08:38:32.374999 2018
        last packet:   Wed Jul 11 08:38:39.085829 2018
        elapsed time:  0:00:06.710829
        total packets: 56301
        filename:      trace.pcap
   a->b:                              b->a:
     total packets:         18420           total packets:         37881
     ack pkts sent:         18420           ack pkts sent:         37881
     pure acks sent:        18339           pure acks sent:           27
     sack pkts sent:            0           sack pkts sent:            0
     dsack pkts sent:           0           dsack pkts sent:           0
     max sack blks/ack:         0           max sack blks/ack:         0
     unique bytes sent:      1920           unique bytes sent:  19361660
     actual data pkts:         81           actual data pkts:      37854
     actual data bytes:      1920           actual data bytes:  19361660
     rexmt data pkts:           0           rexmt data pkts:           0
     rexmt data bytes:          0           rexmt data bytes:          0
     zwnd probe pkts:           0           zwnd probe pkts:           0
     zwnd probe bytes:          0           zwnd probe bytes:          0
     outoforder pkts:           0           outoforder pkts:           0
     pushed data pkts:         81           pushed data pkts:         49
```

Capturer et filtrer en Unix : BPF (Berkerley Packet Filtering)

On peut voir le code avec l'option -d e tcpdump :

```
guillaumesmbp2:Figs urvoy$ tcpdump -d 'host www.unice.fr and port 80' -i en0
(000) ldh      [12]
(001) jeq      #0x800           jt 2    jf 18
(002) ld       [26]
(003) jeq      #0x863bcc09      jt 6    jf 4
(004) ld       [30]
(005) jeq      #0x863bcc09      jt 6    jf 18
(006) ldb      [23]
(007) jeq      #0x84            jt 10   jf 8
(008) jeq      #0x6             jt 10   jf 9
(009) jeq      #0x11            jt 10   jf 18
(010) ldh      [20]
(011) jset     #0x1fff          jt 18   jf 12
(012) ldxb     4*([14]&0xf)
(013) ldh      [x + 14]
(014) jeq      #0x50            jt 17   jf 15
(015) ldh      [x + 16]
(016) jeq      #0x50            jt 17   jf 18
(017) ret      #262144
(018) ret      #0
```

Code semble complexe, mais ce sont des références à des numéros d'octets dans le paquet, des comparaisons :
par exemple ligne 000 et 001 on charge le 12ème octets (numéro de port dst) et on vérifie si il vaut 80, on continue, sinon on va en ligne 18 et on retourne 0 (pas de match pour le filtre)

de BPF à eBPF

- BPF permet ... d'exécuter du code (un filtre) dans le noyau sur des paquets.
- eBPF est une généralisation : exécution (sécurisée)de code sur des objets traités par le noyau, par exemple des paquets, des sockets TCP/UDP...ou toute fonction du noyau
  - Intérêt : pouvoir observer le noyau avec un minimum de perturbation $\rightarrow$ utile en production!!

de BPF à eBPF

Illustration des capacités d'eBPF : les outils développés par Brendan Gregg (Netflix) pour monitorer le réseau, le système, les applications....

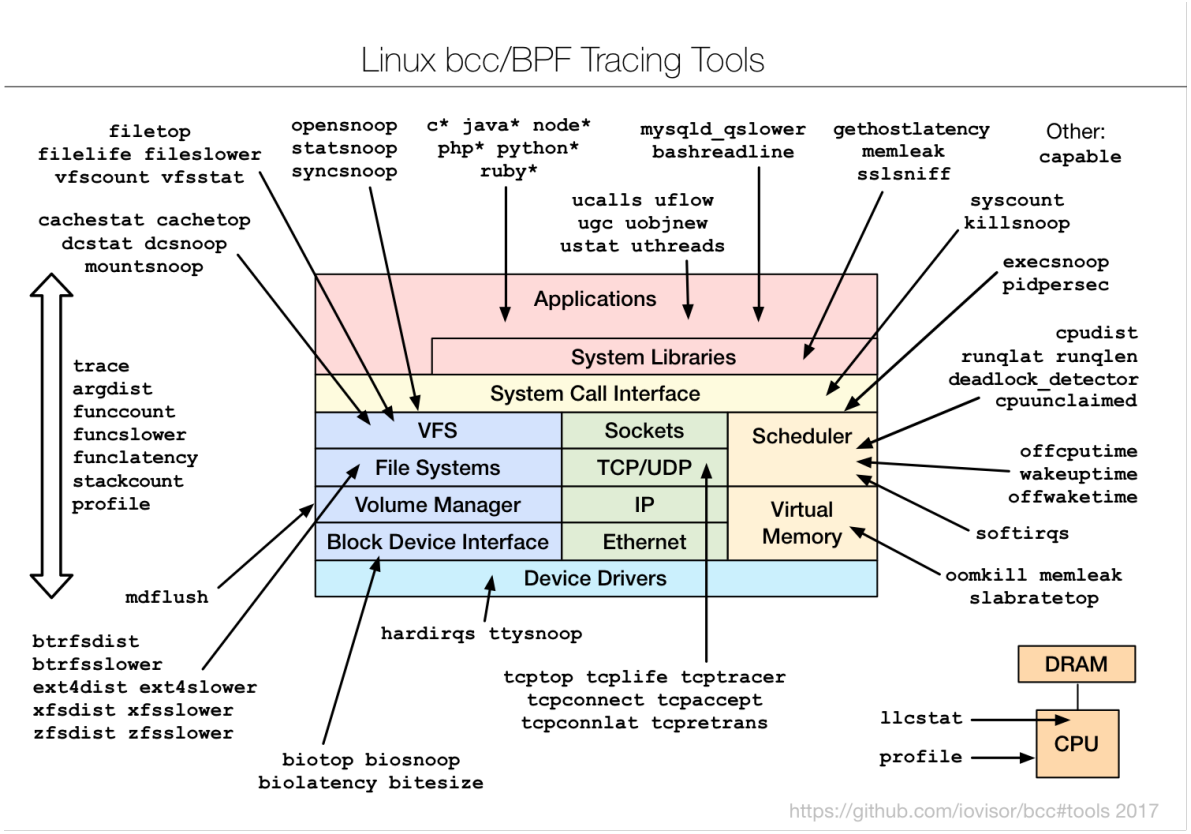


Figure: Source: `http://www.brendangregg.com/ebpf.html`

Monitoring passif: cas switch/routeur physique/virtuel

Netflow, sFlow, port mirrorring

Choses à savoir pour faire des mesures en réseau :

- Un switch/routeur physique n'a pas de disque
- Un switch/routeur a des éléments programmables (circuits ASICs) rapides et une CPU centrale lente : *fast path* et *slow path* respectivement
- Un switch/routeur virtuel a aussi un *fast path* (implémenté dans le noyau) et un *slow path* quand on transite vers l'espace utilisateur

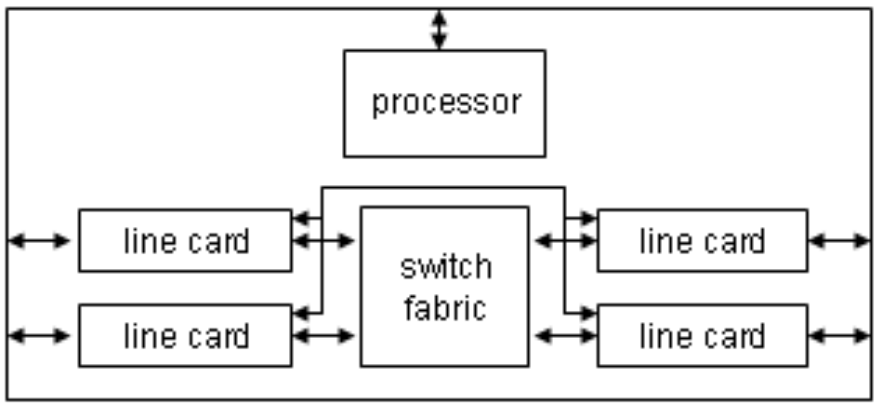


Figure: Source: `http://www.soi.wide.ad.jp/class/20020032/slides/06/img/13.png`

Netflow, sFlow, port mirrorring

pour faire des mesures :

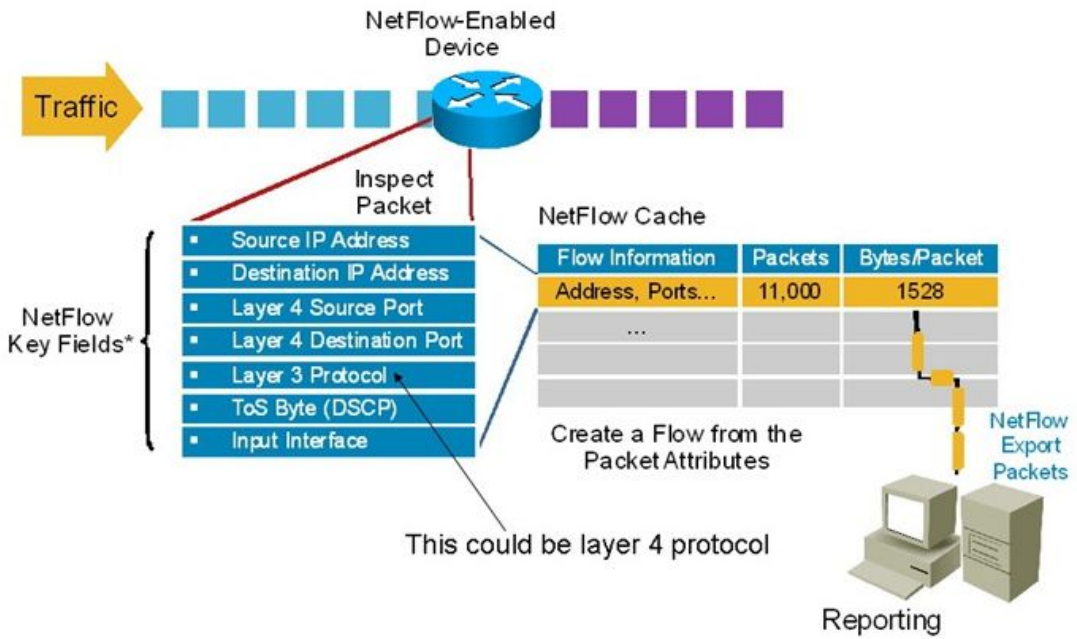
1. On recopie les paquets sur une interface spéciale : *p*ort mirroring
2. On garde des infos en mémoire RAM (de la CPU centrale) sur les paquets $\rightarrow$ goulet d'étranglement possible (*slow path*):
   - Neflow: on garde en mémoire une ligne par connexion/conversation et on exporte toutes les 5 min vers une machine appelée collecteur
   - sFlow: on filtre les paquets et on envoie vers l'en-tête vers le collecteur : pas de calcul côté routeur pour trouver les connexions. Une sorte de port mirroring plus intelligent.

Echantillonnage

En Netflow/sFlow, on un paramètre d'échantillonnage (ex : on traite un paquet sur 100 ou 1000...) pour ne pas surcharger le routeur/switch.
Ex : Facebook utilise sflow sur tous ces équipements et échantillonne un paquet sur 30000!

NetFlow Overview – Cont.

A flow is a unidirectional stream of packets, between a given source and a destination, that have several components in common (seven key fields).



© 2007 – 2010, Cisco Systems, Inc. All rights reserved. Cisco Public

Figure: Source: Cisco

M3105 - Mesures actives : ping, traceroute, smokeping

Guillaume Urvoy-Keller

July 16, 2018

Ping

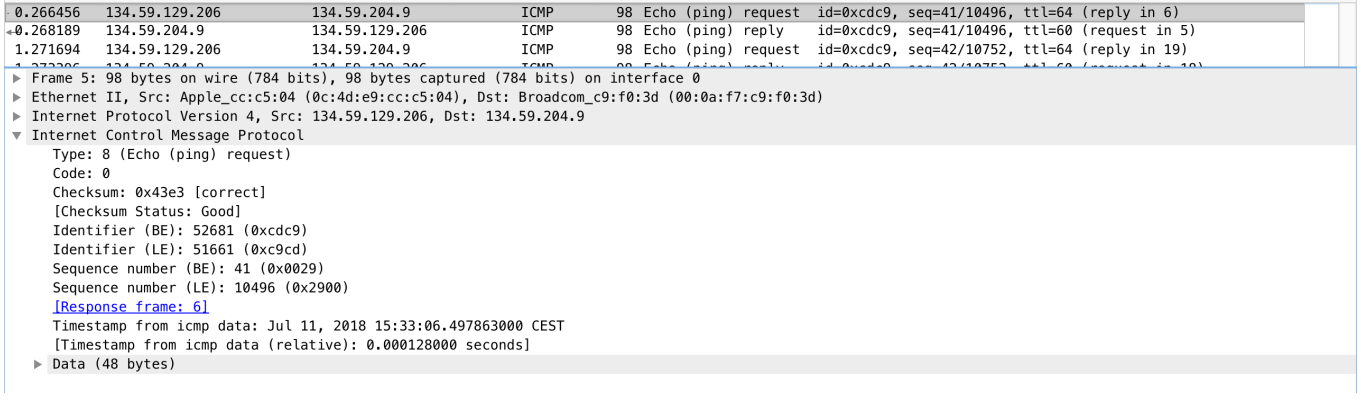
Ping

- Envoie réception de paquet ICMP (Internet Control Message Protocol)
- Code (dans l'en-tête ICMP): echo request et echo reply
- Mesure du temps d'aller-retour (RTT)

Ping

Example

```
guillaumesmbp2:2018 urvoy$ ping www.unice.fr
PING sites.unice.fr (134.59.204.9): 56 data bytes
64 bytes from 134.59.204.9: icmp_seq=0 ttl=60 time=1.810 ms
64 bytes from 134.59.204.9: icmp_seq=1 ttl=60 time=1.714 ms
64 bytes from 134.59.204.9: icmp_seq=2 ttl=60 time=1.864 ms
64 bytes from 134.59.204.9: icmp_seq=3 ttl=60 time=1.932 ms
64 bytes from 134.59.204.9: icmp_seq=4 ttl=60 time=1.936 ms
```

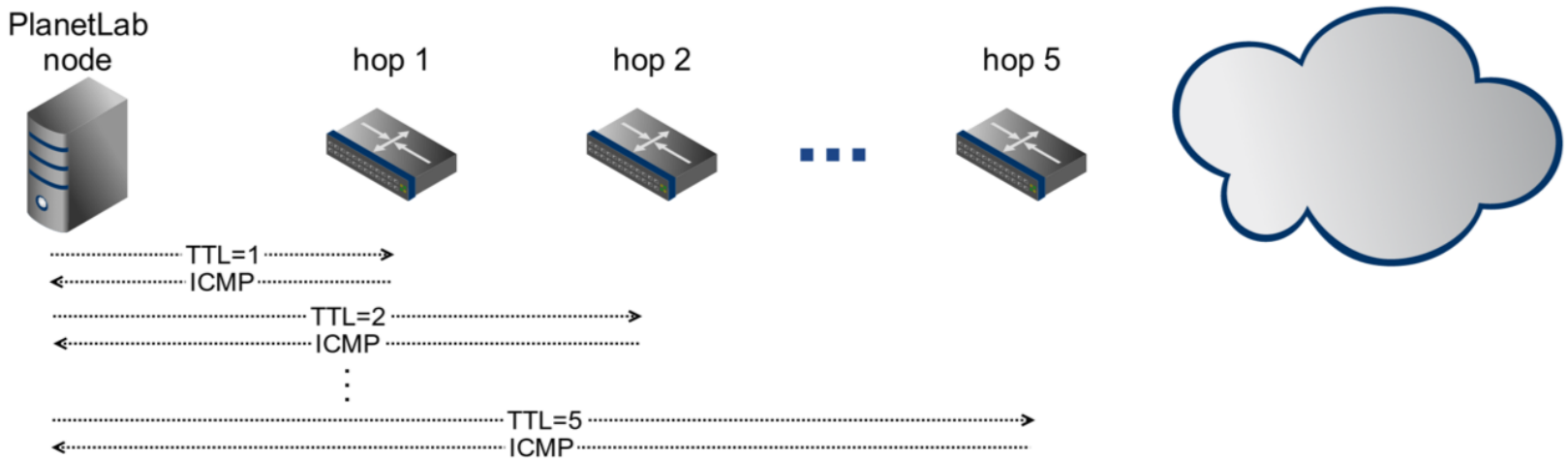


Traceroute

Traceroute

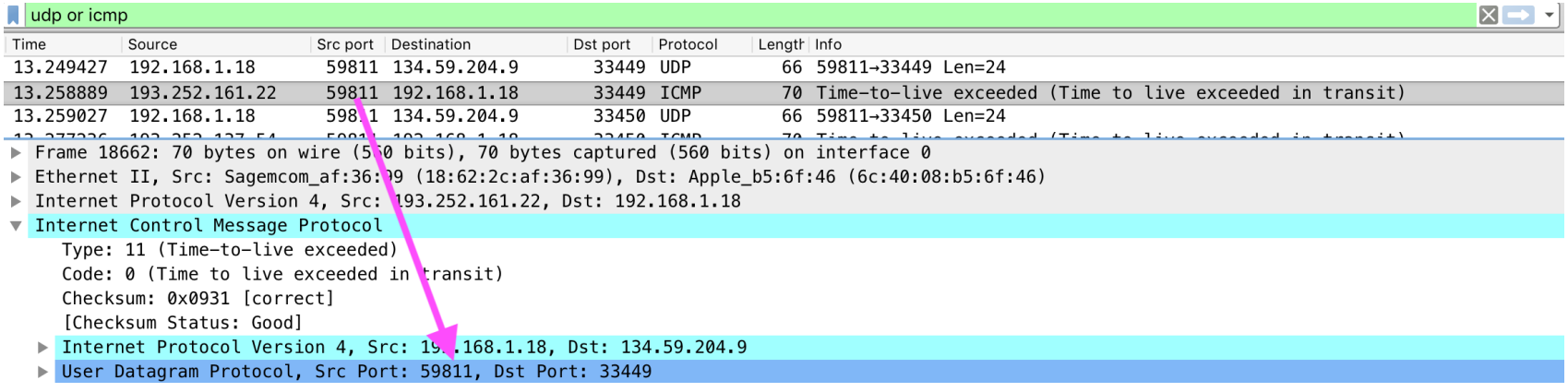
- Traceroute peut envoyer des paquets
  - ICMP
  - UDP ← valeur par défaut
  - TCP

  avec TTL croissant de 33434 à 33534 ....
  et reçoit des routeurs des paquets ICMP code 11 (TTL expired) + une partie du paquet détruit



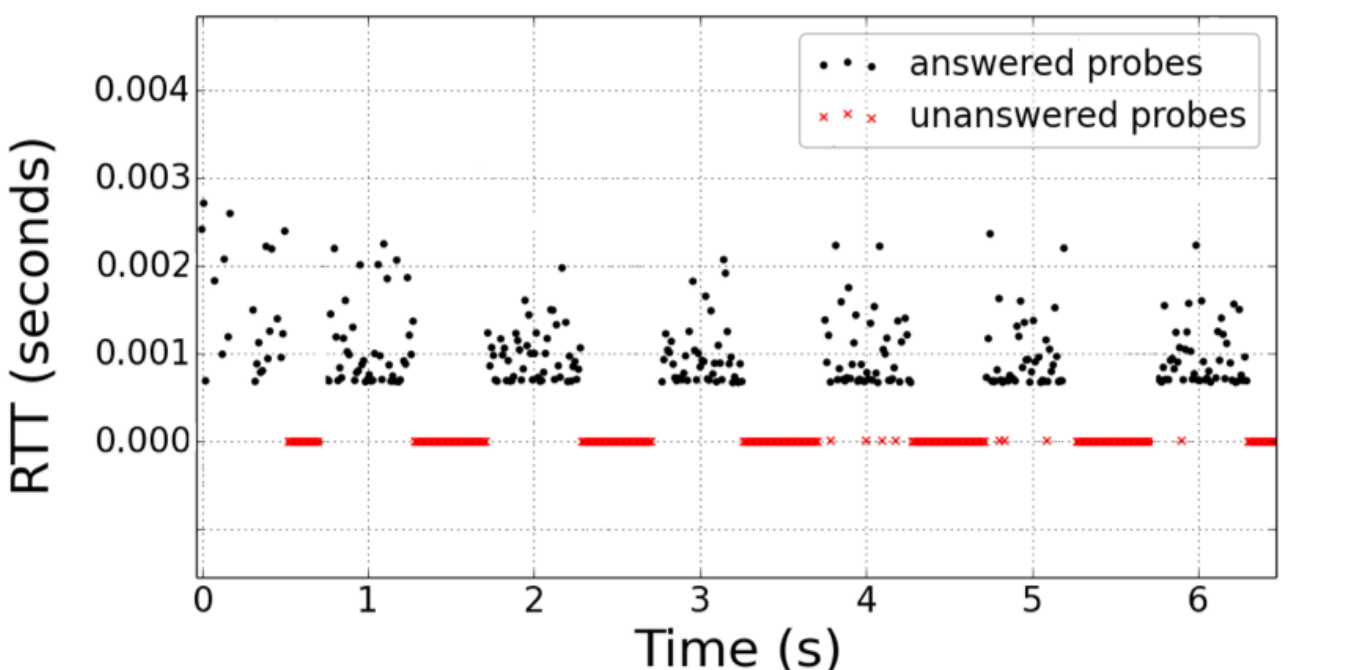
Traceroute

Comportement typique : un paquet UDP envoyé et son en-tête encapsulée dans le paquet ICMP renvoyé par le routeur



Traceroute

- Bonne pratique : Les ISPs doivent configurer leurs routeurs pour répondre à traceroute (répondre en ICMP)
- **mais**, les ISPs veulent prévenir des attaques $\Rightarrow$ limitation du débit de réponse du type x paquets par secondes
- 46 % des routeurs (campagne de mesures 2014-2015, + de 1000 routeurs de l'Internet) :
  - envoie continue de paquets au routeur qui expirent
  - En rouge les paquets sans réponse, en noir avec réponse



Limitation débit de réponse des routeurs aux messages ICMP

Débit fonction de la marque du routeur.

Valeurs typiques : 1 paquet-par-seconde (pps), 10 pps, 50 pps, 100 pps, 500 pps

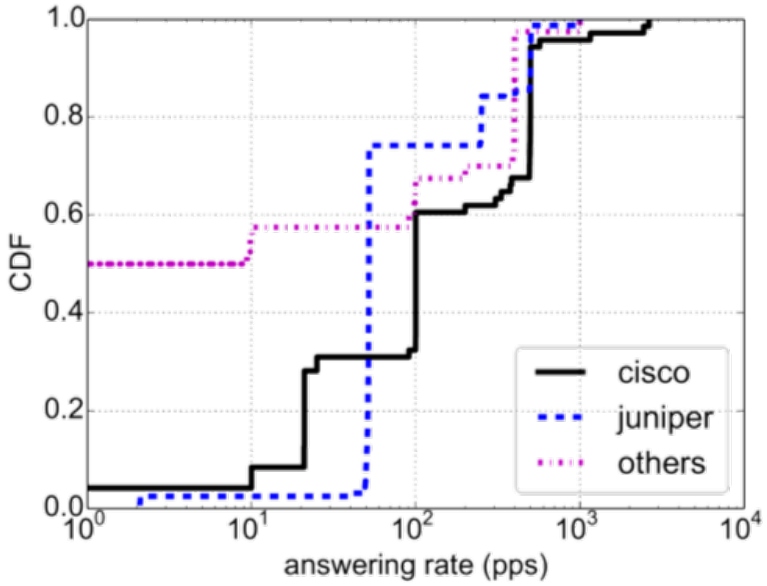


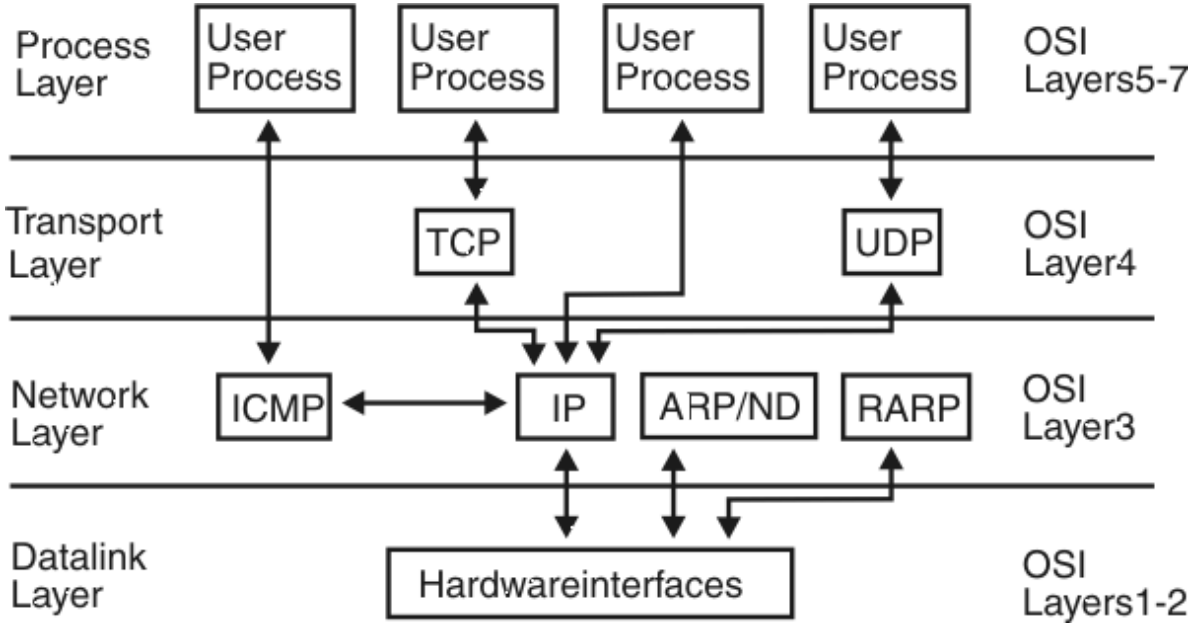
Figure: Riccardo Ravaioli, Guillaume Urvoy-Keller, Chadi Barakat: Characterizing ICMP rate limitation on routers. ICC 2015

Smokeping

Généralisation ping

Pourquoi ne pas faire l'équivalent d'un ping sur différents protocoles ou différentes applications?
Exemples :

- en TCP, on envoie un SYN, on attend un SYN-ACK et on mesure le temps
- en HTTP, on envoie un GET et on attend un 200 OK



Smokeping

- Ping généralisé :
  - « Ping » niveau HTTP $\rightarrow$ un curl sur une page
  - « Ping » niveau SSL $\rightarrow$ openssl sur un serveur
  - « Ping » d'un serveur DNS $\rightarrow$ requête DNS
  - Niveau application : LDAP, NFS
- Représentation graphique avec MRTG
- Base de données RRD (Round Robin Database) :
  - Base de données spécialisées pour stoker des séries temporelles
  - Occupe une taille fixe sur le disque. Les données plus anciennes sont stockées avec une granularité moins fine

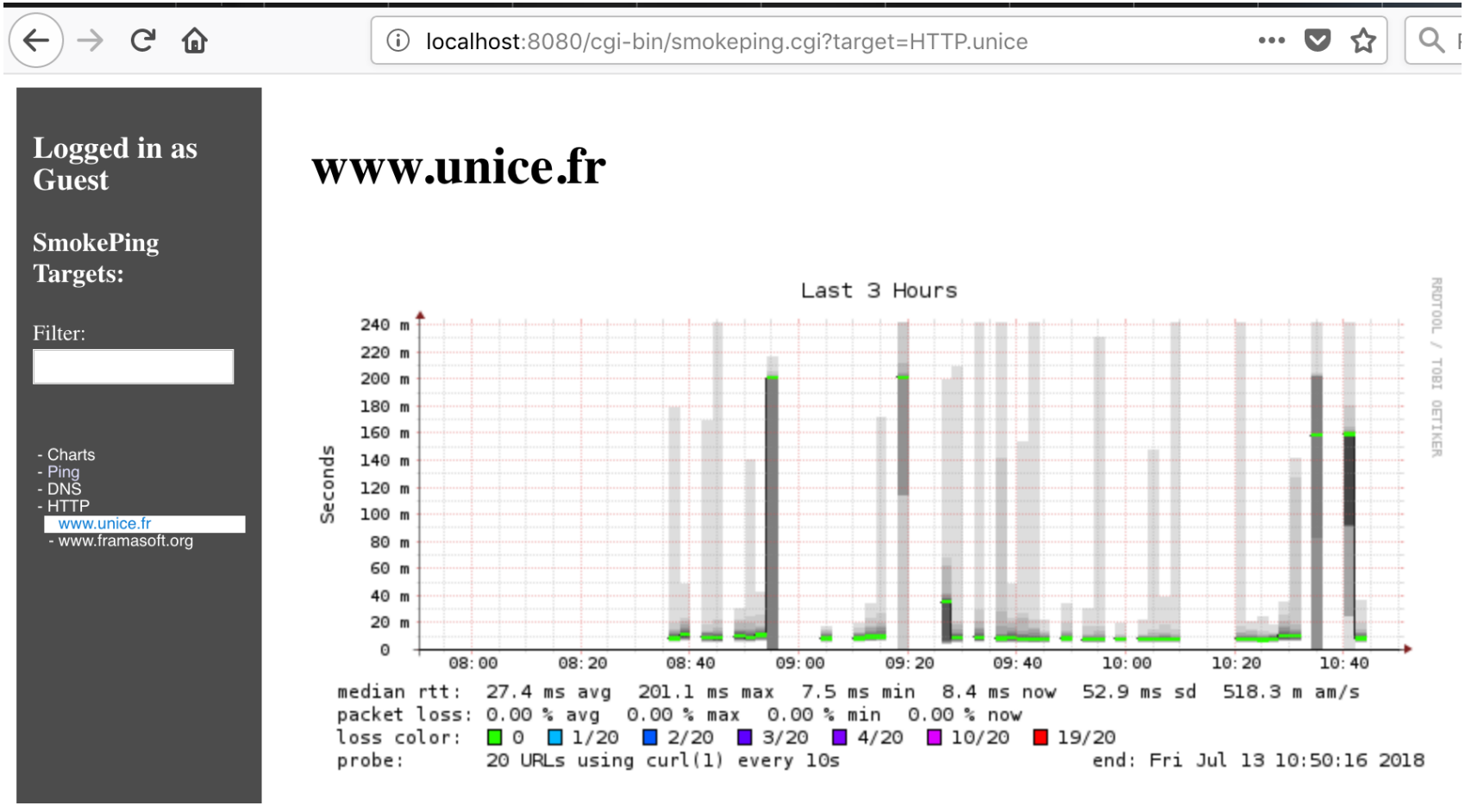
Smokeping

On configure les méthodes de mesure utilisées (Fichier Probes) et les listes de machines sur lesquelles on les applique (Fichier Targets)

```
*** Probes ***

+ FPing
binary = /usr/bin/fping
step = 10

+ DNS
binary = /usr/bin/dig
step = 10

+ Curl
binary = /usr/bin/curl
step = 10
urlformat = http://%host%/

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
Probes                                    1,1           All
```

```
*** Targets ***

probe = FPing

menu = Top
title = Network Latency Grapher
remark = Welcome to the SmokePing website of M3105 course.

+ Ping
menu = Ping
title = Ping of local GW of I3S Lab
host = 172.19.130.254

+ DNS
probe= DNS
menu = DNS
title = DNS on 8.8.8.8
host = 8.8.8.8

+ HTTP
menu = HTTP
title = HTTP on unice website and framasoft
probe = Curl

++ unice
host = www.unice.fr

++ framasoft
host= www.framasoft.org

#alerts = someloss
~
~
~
~
~
~
~
~
Targets                                   1,1           All
```

Smokeping


localhost:8080/cgi-bin/smokeping.cgi?target=HTTP.unice
Logged in as Guest
SmokePing Targets:
Filter:
- Charts
- Ping
- DNS
- HTTP
www.unice.fr
- www.framasoft.org
www.unice.fr
Last 3 Hours
Seconds
240 m
220 m
200 m
180 m
160 m
140 m
120 m
100 m
80 m
60 m
40 m
20 m
0
08:00
08:20
08:40
09:00
09:20
09:40
10:00
10:20
10:40
RRDTOOL / TOBI OETIKER
median rtt: 27.4 ms avg 201.1 ms max 7.5 ms min 8.4 ms now 52.9 ms sd 518.3 m am/s
packet loss: 0.00 % avg 0.00 % max 0.00 % min 0.00 % now
loss color: 0 1/20 2/20 3/20 4/20 10/20 19/20
probe: 20 URLs using curl(1) every 10s
end: Fri Jul 13 10:50:16 2018

M3105 - Supervision Système/Application : SNMP

Guillaume Urvoy-Keller

July 13, 2018

Simple Network Management Protocol

- Très ancien protocole
- Pas seulement protocole mais :
  1. Stockage et nomage des données
  2. Un protocole pour y accéder
- Implémenté dans tous les équipements : serveurs, routeurs, imprimantes, points d'accès wifi, caméra IP
- SNMP v1 (1988) n'est plus utilisé et non compatible avec SNMP V2
- SNMP v2 : RFC 1441 (1993)
- SNMP v3 = SNMP v2 + crypto

SNMP

Vocabulaire :

- Agent : service qui tourne dans serveurs, routeurs, imprimantes, points d'accès wifi, caméra IP et maintien base de données
- Manager : service qui interroge l'agent avec des messages *GET* (récupération valeur) et des messages *SET* (écriture d'une valeur, par exemple le nom de l'équipement)
- L'agent peut aussi être configuré pour envoyer des alertes - *trap*
- Protocole SNMP au dessus d'UDP (questions et réponses simples et courtes)

Communication Agent/Manager

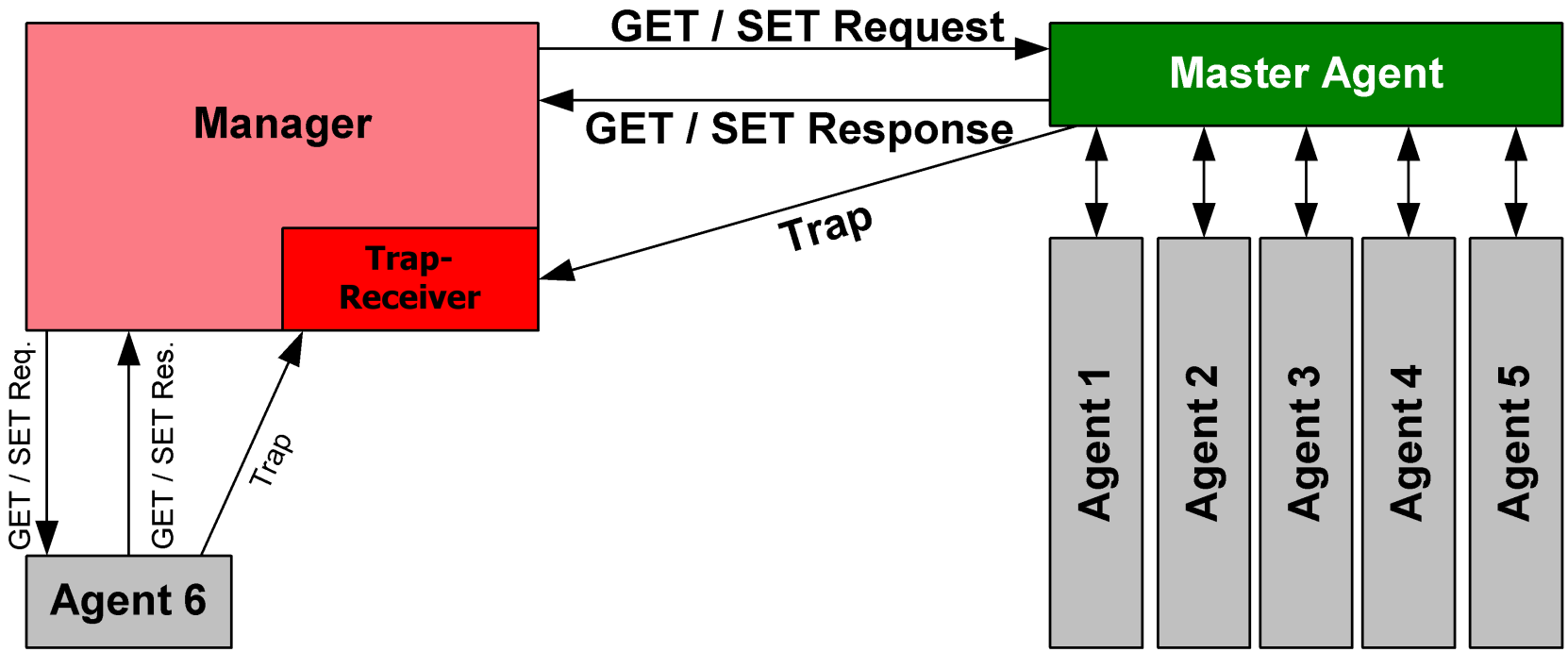


Figure: Source : https://en.wikipedia.org/wiki/Simple_Network_Management_Protocol#/media/File:SNMP_communication_principles_diagram.PNG

MIBs SNMP

- Une base de données d'un agent s'appelle une MIB : Management Interface Base
- En fait, un agent implémente une ou plusieurs MIB :
  - Une MIB de base que l'on trouve partout, la MIB-2
  - des MIBs particulières suivant que l'équipement est un serveur Web, un serveur Radius.

SNMP

Pourquoi SNMP reste intéressant?

1. Parce qu'on le trouve sur tous les équipements réseau.
2. Parce que les bases de données SNMP sont très riches.

Pourquoi est-ce simple d'utilisation (un manager sait lire toutes les données d'un agent si il a les mots de passe)?

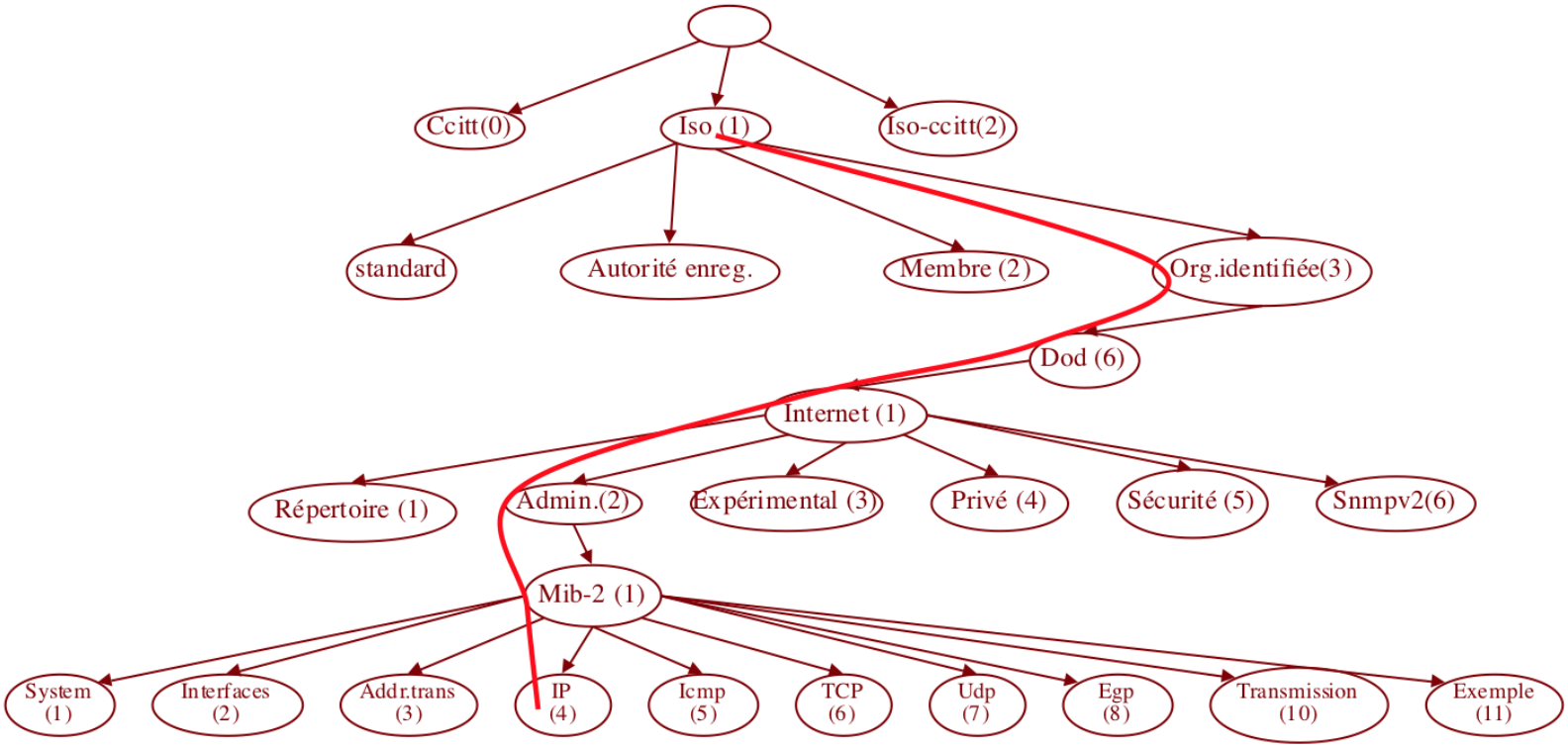
1. Parce que les MIBs sont normalisées et on a une adresse unique dans un grand arbre que tout le monde connait
   - Les objets dans chaque MIB ont un adresse unique $\Rightarrow$ le paramètre sys.Desc qui est une chaine de caractère s'appelle toujours avec un **GET iso.3.6.1.2.1.1.1.0**

Richesse SNMP

Des infos sur le système, les interfaces réseaux, les connexions TCP en cours, le hardware, les processus qui tournent

```
root@ubuntu-xenial:/etc/snmp# snmpwalk -Os -c public -v 2c 127.0.0.1
sysDescr.0 = STRING: Linux ubuntu-xenial 4.4.0-130-generic #156-Ubuntu SMP Thu Jun 14 08:53:28 UTC 2018 x86_64
sysObjectID.0 = OID: netSnmpAgentOIDs.10
sysUpTimeInstance = Timeticks: (36710) 0:06:07.10
[...]
ifDescr.2 = STRING: Intel Corporation 82540EM Gigabit Ethernet Controller
[...]
tcpConnState.10.0.2.15.22.10.0.2.2.52002 = INTEGER: established(5)
[...]
hrDeviceDescr.196608 = STRING: GenuineIntel: Intel(R) Core(TM) i5-4278U CPU @ 2.60GHz
[...]
hrSWRunName.11581 = STRING: "curl"
hrSWRunName.11630 = STRING: "kworker/1:1"
hrSWRunName.14240 = STRING: "apache2"
hrSWRunName.14243 = STRING: "apache2"
```

Arbre des MIBs



IP = 1.3.6.1.2.1.4

http://www.alvestrand.no/objectid/

http://wwwsnmp.cs.utwente.nl/ietf/mibs/

MIBs d'un agent

- Tout équipement doit implanter la MIB-2, i.e. gérer les objets de cette MIB (RFC 1213) :
  - Statistiques sur les interfaces (vitesse, MTU, octets envoyés/reçus, etc.)
  - Informations sur le système : localisation, contact
- En plus de la MIB-2, un agent implantera :
  - Des MIBs standardisées supplémentaires
    Ex : BGP version 4 MIB (RFC 1657), RADIUS Authentication Server MIB (RFC 2619)
  - Des MIBS propriétaires sous la partie **privé** de l'arbre

Contrôle d'accès SNMP

- Communauté SNMP (v1 et v2) = **mot de passe**
- Une communauté est associée à des droits : read-only, read-write
  - public : read-only
  - private : read-write
- Chaque communauté snmp peut limiter les accès à des objets $\rightarrow$ notion de vues
- Communauté envoyé en clair en SNMP v2
  - Il existe un « authentication trap » pour savoir si quelqu'un a essayé de se connecter avec un mauvais mot de passe
  - On indique en plus l'adresse IP du manager

Contrôle d'accès

Fichier de configuration snmpd.conf d'une machine Debian

```
###############################################################################
#
#  AGENT BEHAVIOUR
#
#  Listen for connections from the local system only
agentAddress  udp:127.0.0.1:161      # Par défaut, agent écoute seulement la loopback
#  Listen for connections on all interfaces (both IPv4 *and* IPv6)
#agentAddress udp:161,udp6:[::1]:161
###############################################################################
#
#  ACCESS CONTROL
#
                                                #  system + hrSystem groups only
view   systemonly  included   .1.3.6.1.2.1.1    # one définit une vue = une partie de l'arbre MIB2
view   systemonly  included   .1.3.6.1.2.1.25.1
                                                #  Full access from the local host
#rocommunity public  localhost
                                                #  Default access to basic system info
 rocommunity public  default    -V systemonly    #on associe une communauté à une vue
                                                #  rocommunity6 is for IPv6
 rocommunity6 public  default   -V systemonly
###############################################################################
#
#  SYSTEM INFORMATION
#
sysLocation    Sitting on the Dock of the Bay   # on va faire un SET sur la base de données pour mettre les bonnes valeur
sysContact     Me <me@example.org>
                                                # Application + End-to-End layers
sysServices    72
###############################################################################
#
```

Managers

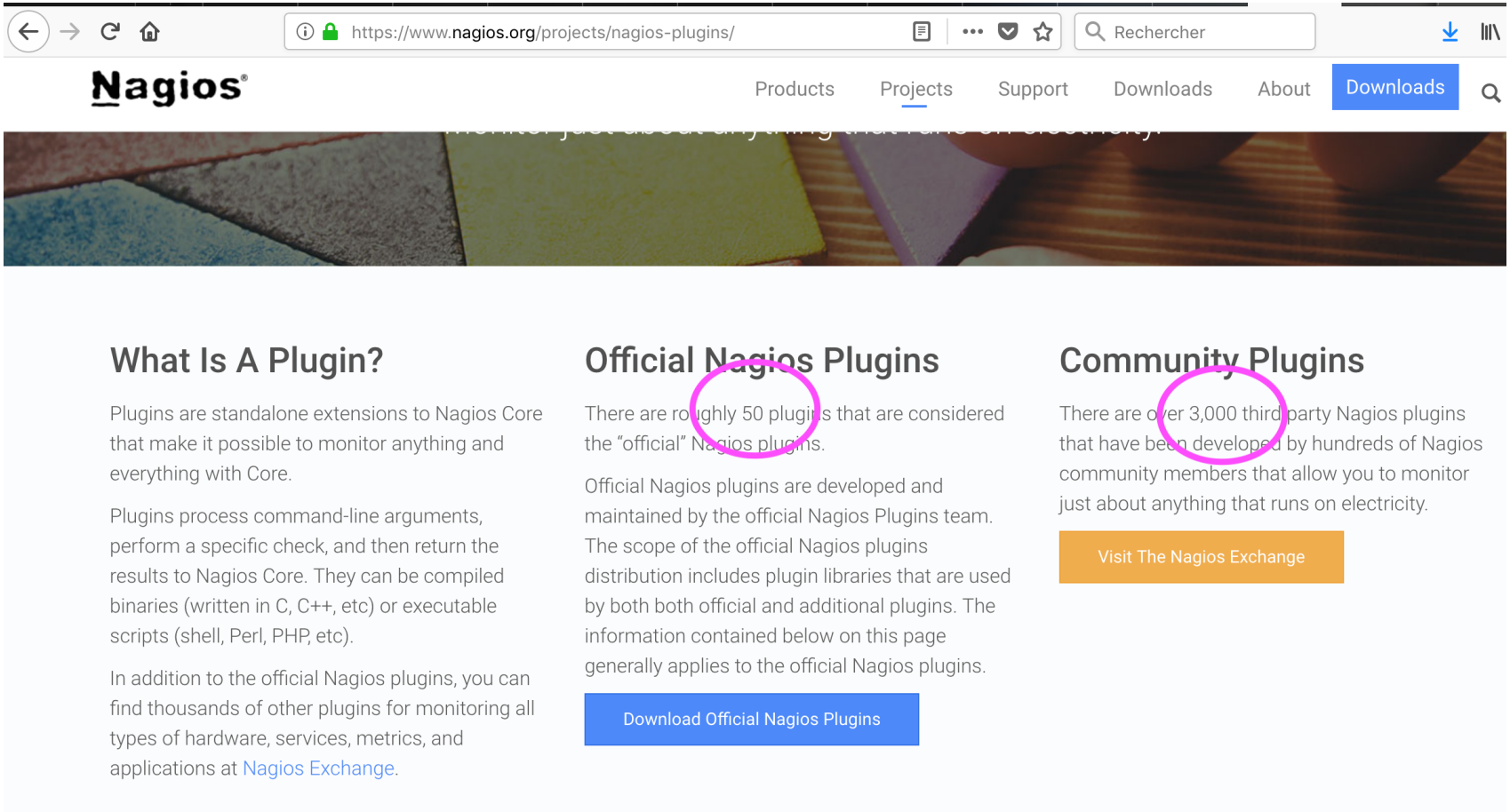
Différentes approches

1. On utilise un outil spécifique
   - Ces outils interrogent en SNMP et font les affichages
   - Ils supportent des pluggins qui s'installent sur les serveurs (en plus des agents SNMP) pour récupérer de données niveau applicatif

   Ex: Nagios, Observium (`http://demo.observium.org`), Cacti, Munim
2. Pour le cas des serveurs, on utilise des suites d'outils qui
   1. collectent les données bas-niveau (SNMP ou niveau applicatif)
   2. les écrivent dans une base de données
   3. Offre un moteur d'affichage pour créer ses propres pages Web

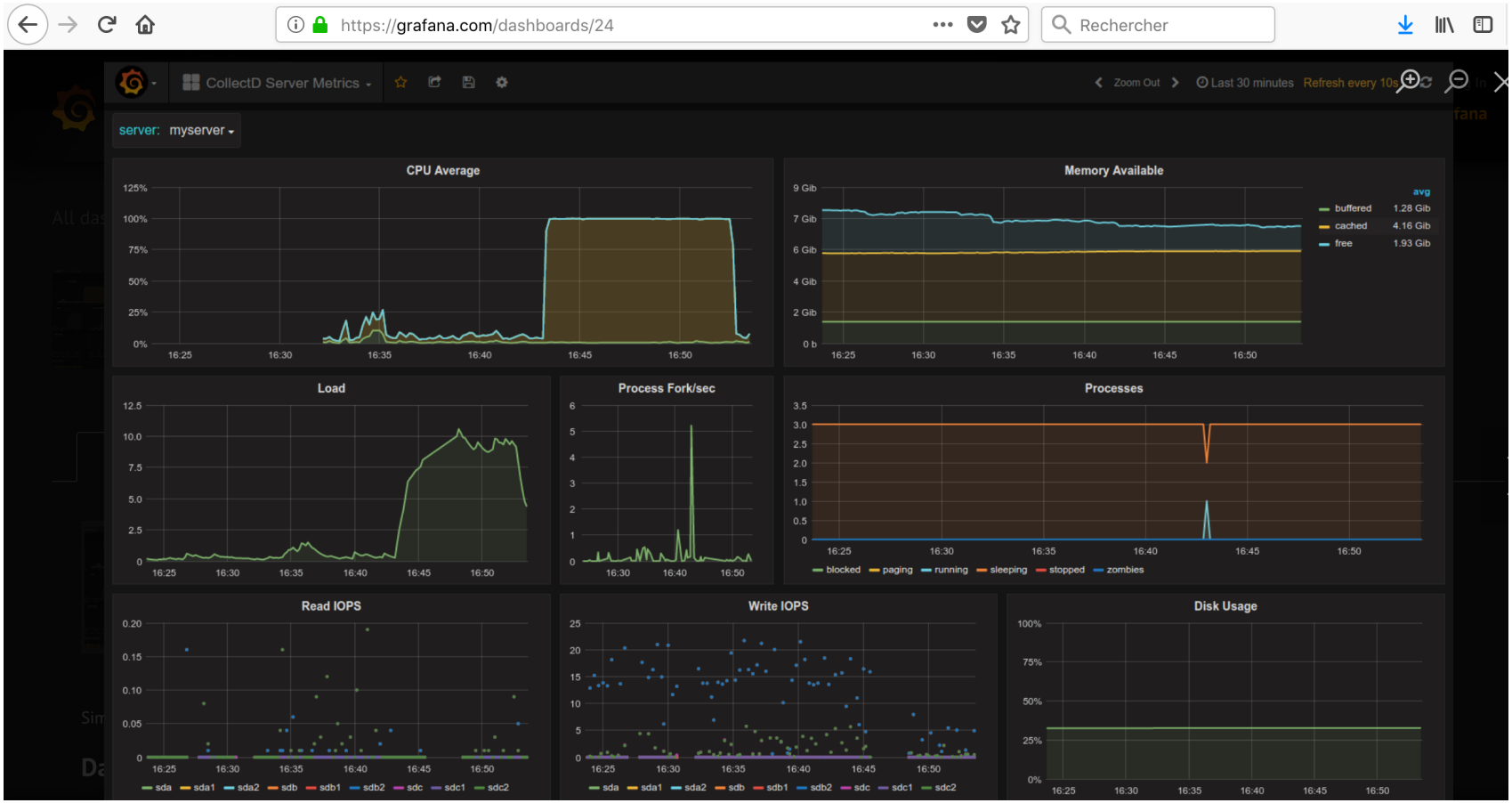
   Ex: Collectd + Graphite + Grafana ou Telegraph

Nagios



Avantage Nagios and co.: fonctionne pour tous les équipements (serveur...caméra IP)

Grafana



Avantage collectd+graphite+grafana : hautement customisables mais réservé aux serveurs.

M3105 : Ansible - Configuration automatique de serveurs/services

Guillaume Urvoy-Keller

July 16, 2018

Ansible

Automatisation Configuration

Cycle typique :

1. Installation serveur
2. Configuration $\leftarrow$ Ansible

Configuration

Ce que l'on veut automatiser :

- installation de package
- démarrage/re-démarrage de services
- copies de fichier, par exemple page d'accueil serveur Web.

De plus, on veut :

- pouvoir configurer plein de machines à la fois
  - Ex: 50 machines viennent d'être démarrées sur un cloud public ou privé et il faut les configurer
- appliquer une configuration fonction de la distribution de la machine
  - Ex : *yum install apache2* si Centos et *apt install apache2* si Debian/Ubuntu
- avoir un retour clair sur ce qui a été fait/ce qui a échoué

Ansible et ses concurrents

Les concurrents les plus connus :

- Chef et Puppet
- Puppet
- Ansible

Ansible et ses concurrents

Modèle *Pull* pour Chef/Pupplet

on installe un logiciel sur le serveur à configurer et ce client va interroger un serveur central

Modèle *Push* pour Ansible

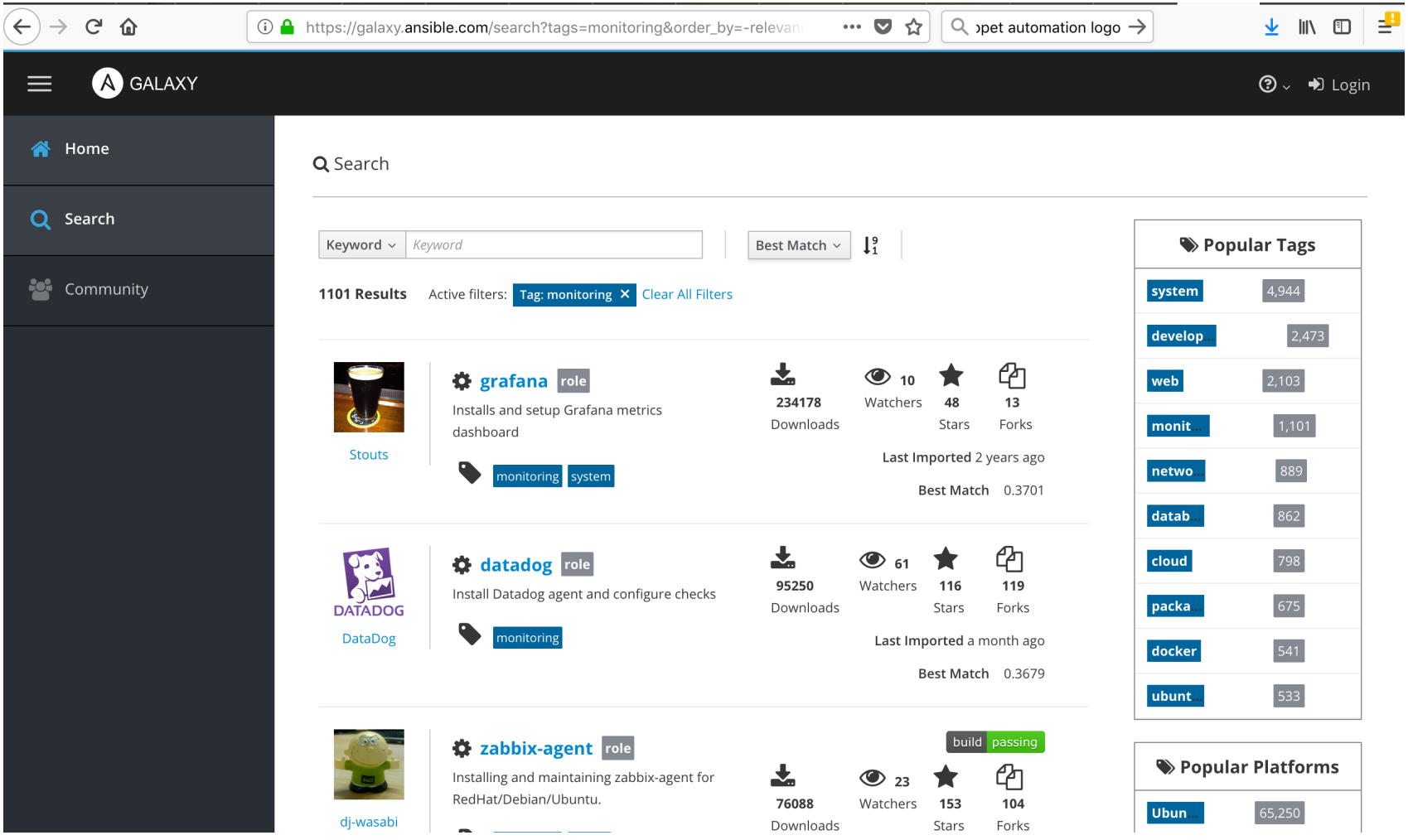
le serveur central pousse directement la configuration en ssh

Ansible - caractéristiques

- Fichiers de configuration simples à écrire et à lire : pas de programmation complexe!
- Notion de **Playbook** : ensemble de commandes à appliquer pour une certaine tâche
  Ex : playbook configuration base serveur centos ou playbook apache
- Notion de **Role** : niveau d'abstraction au dessus d'un playbook
  Ex: permet d'appliquer au serveur centos-apache les rôles *centos* et *apache*.
  Ex: si serveur Ubuntu, alors on appliquera les rôles *ubuntu* et *apache*

Ansible Galaxy

Ansible Galaxy : communauté utilisateurs Ansible s'échange des fichiers de configuration de rôles



Playbook

Ecrits en YAML

```
- name: Configure webserver with nginx
  hosts: testserver
  sudo: True
  tasks:
    - name: install nginx
      apt: name=nginx update_cache=yes
    - name: copy nginx config file
      copy: src=files/default dest=/etc/nginx/sites-available/default
    - name: enable configuration
      file: >
        dest=/etc/nginx/sites-enabled/default
        src=/etc/nginx/sites-available/default
        state=link
    - name: copy index.html
      template: src=templates/index.html.j2 dest=/usr/share/nginx/html/index.html mode=0644
    - name: restart nginx
      service: name=nginx state=restarted
```

Playbook

En-tête : configuration générale
Tâches : liste d'action séquentielle à exécuter sur serveurs, via des modules
Modules : prise en charge d'une action. Ex: modules **apt** ou **service**

Playbook

Variables : au début d'un playbook, Ansible scan la machine cible et récupère des infos (type de distribution, adresse IP, ...).
Ex : module **template va le fichier index.html.j2 et l'instancier**

```
<html>
<head>
<title>
Welcome to ansible
</title> </head>
<body>
<h1>
nginx, configured by Ansible
</h1>
<p>
If you can see this, Ansible successfully installed nginx.
</p>
<p>
{{ ansible_managed }}
</p>
</body>
</html>
```

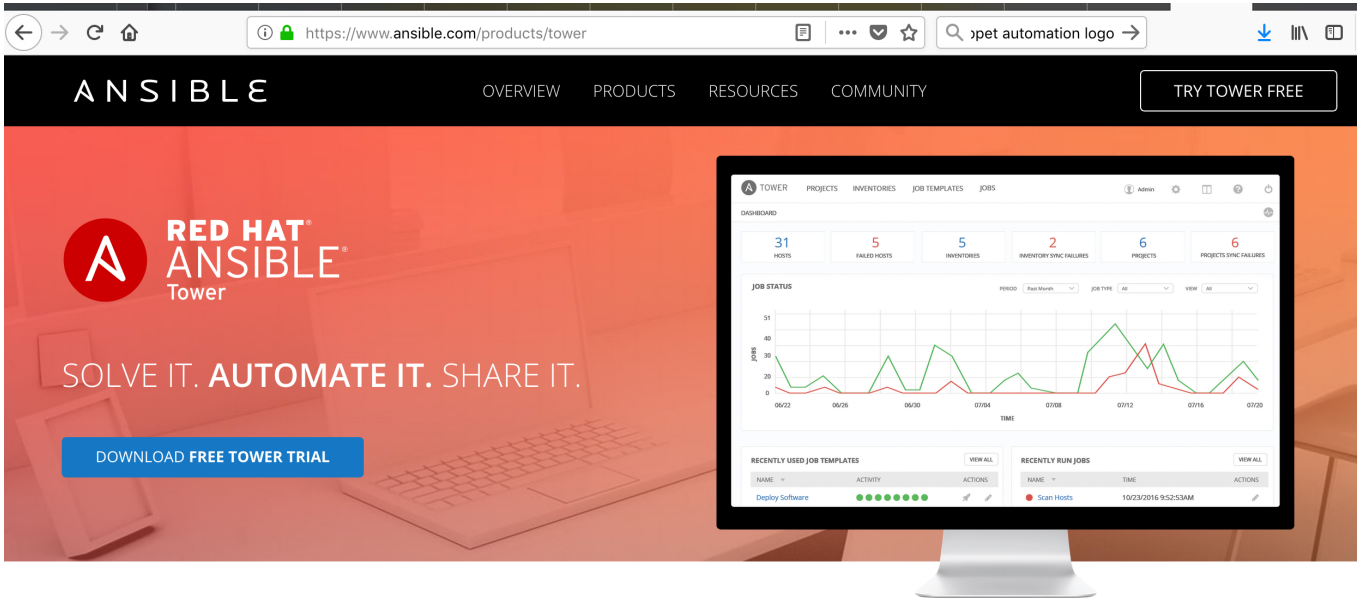
Ansible Playbook

Dans le playbook, on précise les serveurs cibles.
Le serveur testserveur dans le cas précédent
On peut aussi faire des groupes de serveurs qui seront traités en parallèles. Ex : [web] à la place de testserveur

```
---
[web]
web-1.example.com
web-2.example.com
[db]
db-a.example.com
db-b.example.com
```

Ansible Tower

- Ansible est open-source et supporté par Red-hat
- Red-hat vend un produit qui utilise Ansible : Ansible Tower.
  Ajoute une interface graphique, simplifie encore l'automatisation et le suivi.
  Vise le marché des **devops**



M3105 - DNS : Domain Name Service

Guillaume Urvoy-Keller

July 16, 2018

DNS

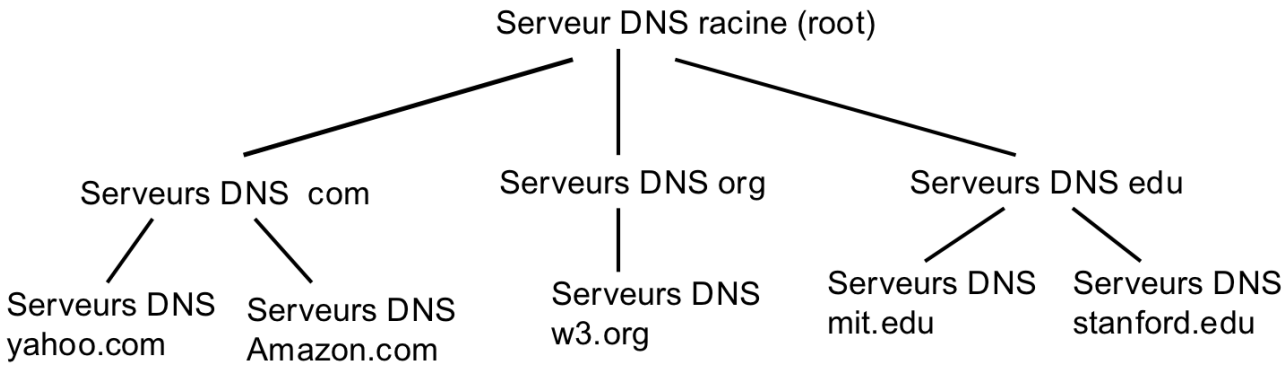
- Base de données distribuée implantée par une hiérarchie de serveurs de noms
- Protocole de niveau applicatif
  - Protocole client/serveur
  - Suite de requêtes/réponses
- Utilise UDP comme couche transport
  - Port : 53
  - UDP est suffisant car requêtes/réponses tiennent dans un datagramme
  - Avec TCP, il faudrait établir la cx $\rightarrow$ perte de temps!
  - Si requête (ou réponse perdue) le client réémet au bout d'un certain temps
- Pleins de clients : nslookup (windows/unix), host, dig

Exemple client : dig

dig montre bien la structure du paquet DNS avec ses différentes parties (question, réponse, etc.)

```
guillaumesmbp2:DNS urvoy$ dig www.unice.fr

; <<>> DiG 9.10.6 <<>> www.unice.fr
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 420
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.unice.fr.                  IN      A

;; ANSWER SECTION:
www.unice.fr.           38902   IN      CNAME   sites.unice.fr.
sites.unice.fr.         38902   IN      A       134.59.204.9

;; Query time: 36 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Mon Jul 16 10:04:03 CEST 2018
;; MSG SIZE  rcvd: 74
```

Une base de données distribuées



Les serveurs roots connaissent les serveurs TLD qui connaissent les serveurs ayant autorités.
Seuls les serveurs ayant autorité contiennent les bases de données des sites $\rightarrow$ c'est ce que l'on veut apprendre à configurer en M3105

Le *Resolver*

Ne fait pas partie de la hiérarchie

Proxy entre les clients locaux et les serveurs DNS

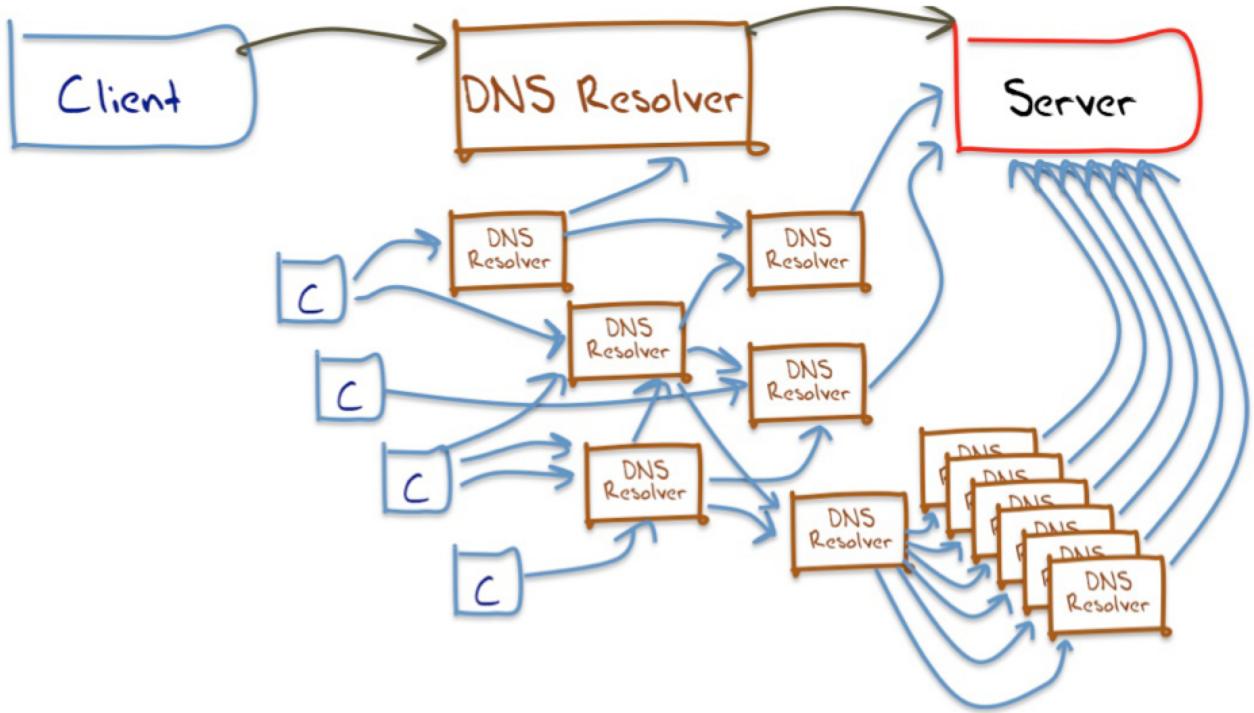


Figure: `https://labs.ripe.net/Members/gih/dnsfig2.png`

Base de donnée d'un serveur ayant autorité

Une compagnie possède

- Des adresses IPs
- Un nom de domaine

donc

Une compagnie possède

- Une zone directe nom $\rightarrow$ adresse IP
- Une zone inverse adresse IP $\rightarrow$ nom

Arbre des zones

Toutes les zones (directes **et** inverses) sont dans le même arbre de référence.

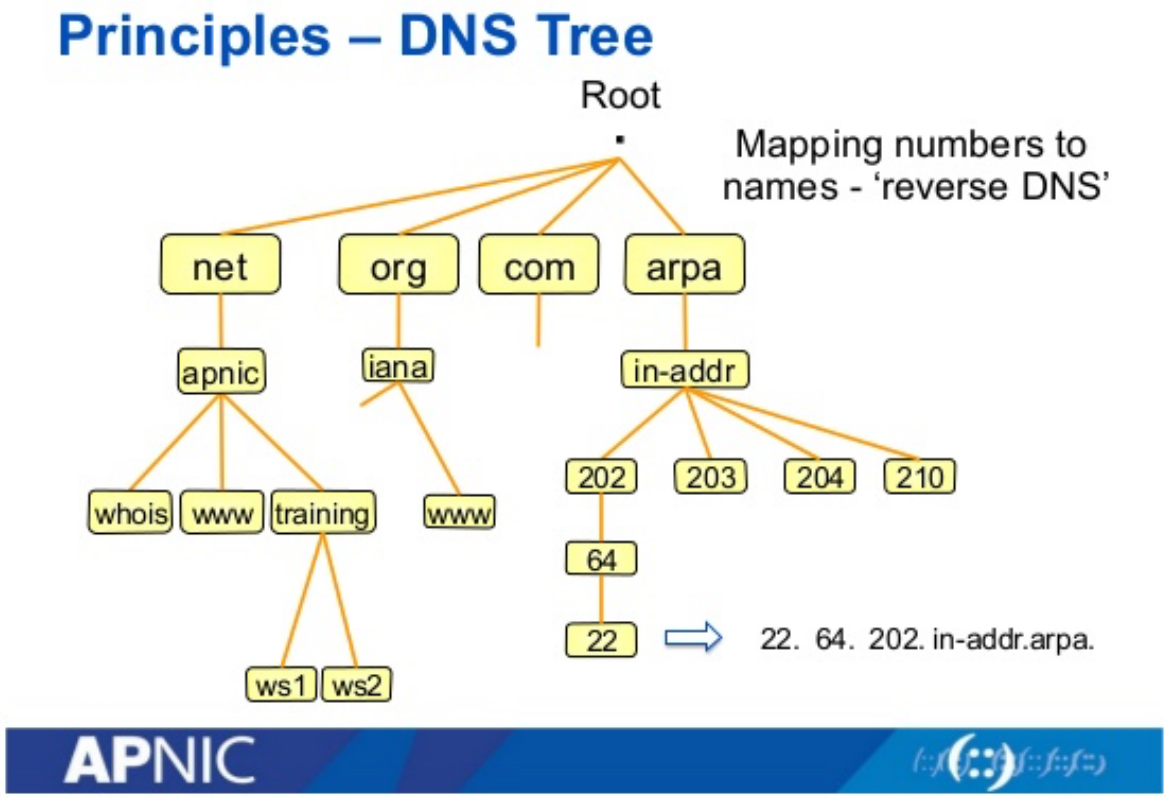


Figure: Source : `https://blog.apnic.net/author/roman/`

Fichier de zone directe

- @ est l'alias du nom de la zone
- Contient des enregistrements. Ex : *nom_machine A adresse_IP* ou *nom_domaine NS nom_serveur_DNS*
- Enregistrement important : le SOA en début de fichier

```
$TTL    86400 ; 24 hours could have been written as 24h or 1d
; $TTL used for all RRs without explicit TTL value
$ORIGIN example.com.
@  1D  IN  SOA ns1.example.com. hostmaster.example.com. (
                                2002022401 ; serial
                                3H ; refresh
                                15 ; retry
                                1w ; expire
                                3h ; minimum
                               )
@       IN  NS     ns1.example.com. ; in the domain
@       IN  NS     ns2.smokeyjoe.com. ; external to domain
@       IN  MX  10 mail.another.com. ; external mail provider
; server host definitions
ns1     IN  A      192.168.0.1  ;name server definition
www     IN  A      192.168.0.2  ;web server definition
```

Fichier de zone inverse. Ex: on suppose ici que c'est 192.168.0/24

Nom de zone inverse est écrit dans le même ordre de priorité que zone directe : example.com et 0.168.192.IN-ADDR.ARPA car ARPA ou .com sont les plus génériques.

Important

- Dans un enregistrement, par exemple : nom_serveur A IP_serveur, si nom_serveur ne finit par un point, le serveur ajoute le nom de la zone!
- Erreur typique : on écrit serveur.example.com qui sera traduit en serveur.example.com.example.com.
- Idem avec les adresses IPs à gauche des enregistrement. Ex: 1 se traduira en 192.168.0.1

Fichier de zone inverse. Ex: on suppose ici que c'est 192.168.0/24

```
$TTL 86400 ; 24 hours, could have been written as 24h or 1d
$ORIGIN 0.168.192.IN−ADDR.ARPA.
@ 1D IN SOA ns1.example.com. hostmaster.example.com. (
                                        2002022401 ; serial
                                        3H ; refresh
                                        15 ; retry
                                        1w ; expire
                                        3h ; minimum
                                       )
; Name servers for the zone − both out−of−zone − no A RRs required
          IN NS ns1.example.com.
          IN NS ns2.smokeyjoe.com.
; server host definitions
1 IN PTR ns1.example.com.
2 IN PTR www.example.com.
; non server domain hosts
3 IN PTR bill.example.com.
4 IN PTR fred.example.com.
```

Relation nom de zone et fichier de zone

Le nom de la zone est décrit dans un le fichier de config qui pointe vers le fichier de zone correspondant

```
// BIND configuration file

options {
        directory "/var/named";
        // Place additional options here.
};
zone "movie.edu" in {
        type master;
        file "db.movie.edu";
};

zone "249.249.192.in-addr.arpa" in {
        type master;
        file "db.192.249.249";
};

zone "253.253.192.in-addr.arpa" in {
        type master;
        file "db.192.253.253";
};

zone "0.0.127.in-addr.arpa" in {
        type master;
        file "db.127.0.0";
};

zone "." in {
        type hint;
        file "db.cache";

};
```

M3105 - LDAP : Lightweigth Directory Access Protocol

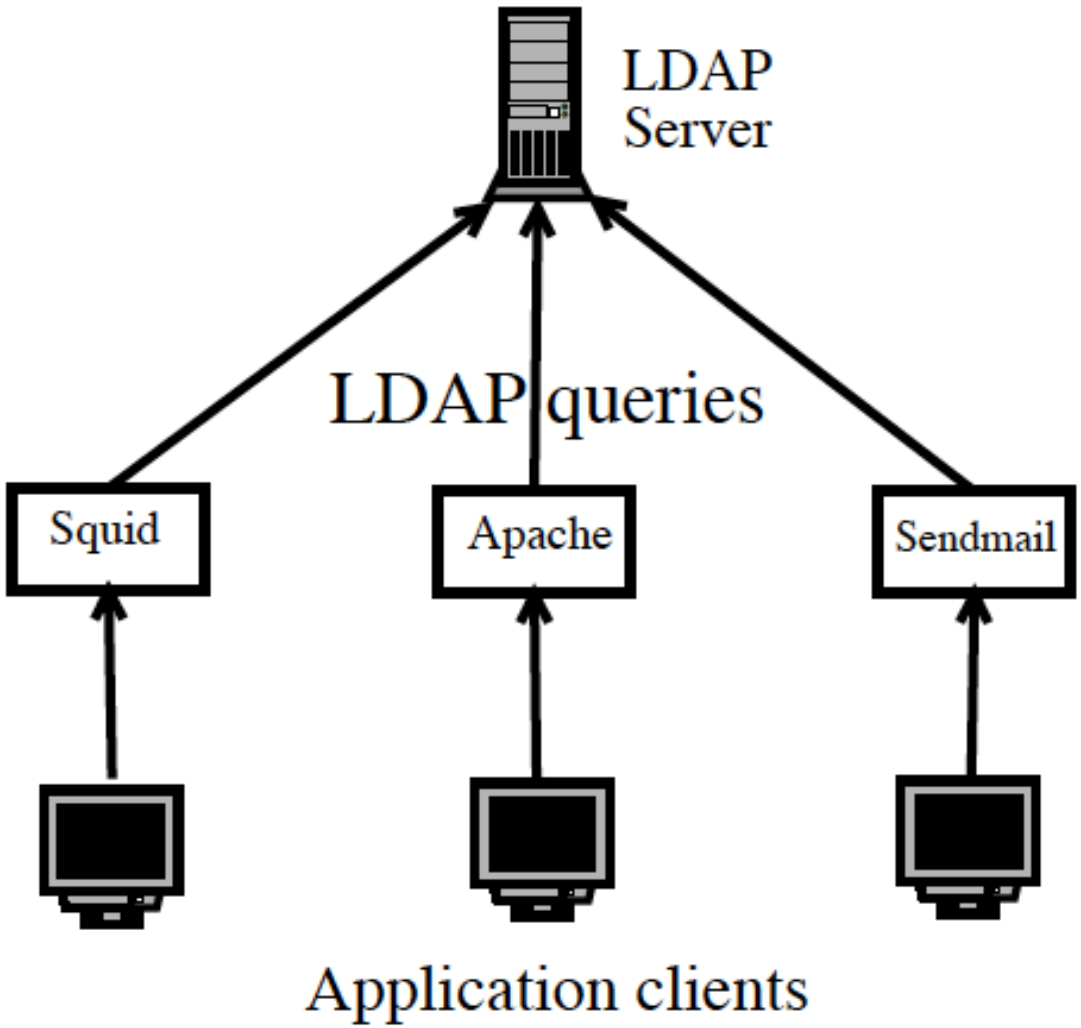
Guillaume Urvoy-Keller

July 16, 2018

Introduction

- Entreprises ont besoin de distribuer des données :
  - Email
  - Téléphone
  - Comptes informatique
- Solution incontournable : LDAP – Lightweigth Directory Access Protocol
- Standardisé par IETF
- Dans le monde Windows, Active Directory utilise LDAP pour gérer les machines et les comptes
- Les applications ont aussi besoin de vérifier les comptes... et donc d'accéder à LDAP

Besoin d'authentification



Exemple d'informations stockées

Un exemple typique d'une entrée dans /etc/passwd exprimée sous forme LDAP

```
uid: ghopper
cn: Grace Hopper
userPassword: {crypt}$1$pZaGA2RL$MPDJoc0afuhHY6yk8HQFp0
loginShell: /bin/bash
uidNumber: 1202
gidNumber: 1202
homeDirectory: /home/ghopper
```

Exemple d'informations stockées

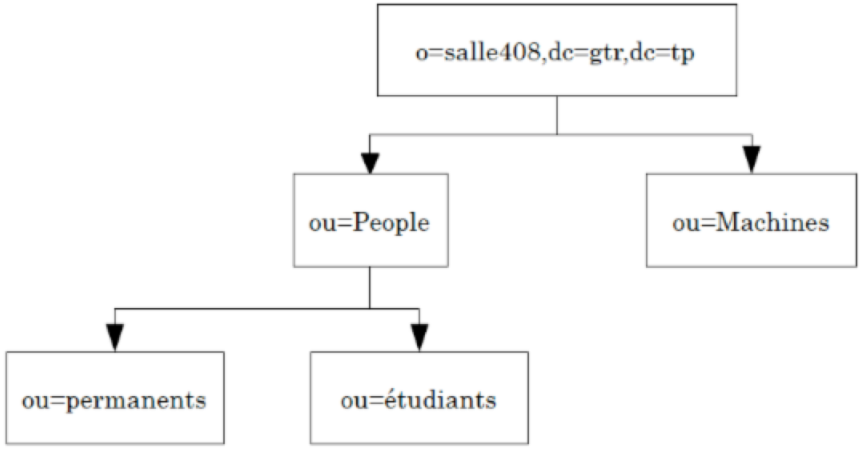
Dans l'exemple précédent :

- Les données sont au format LDIF
- Les données sont stockées dans la base LDAP sous une forme différente (binaire...)
- Mais la possibilités de convertir dans les deux sens "Base de données $\leftrightarrow$ LDIF" est simple et permet échange entre serveurs LDAP

Les entrées LDAP

- Les entrées LDAP sont organisées de manière hiérarchique avec un **identifiant unique** par entrée appelé **dn**
- Données organisées en arbre $\rightarrow$ dn donne chemin dans l'arbre
- Racine de l'arbre : dc=navy, dc=mil :
  - schéma typique où on utilise le domaine DNS pour définir la racine, ici navy.mil

Base LDAP "complète"




```
dn: o=salle408,dc=RT,dc=tp
objectClass: top
objectClass: organization
o: salle408

dn: cn=admin,o=salle408,dc=RT,dc=tp
cn: admin
objectClass: organizationalRole
objectClass: top
objectClass: simpleSecurityObject
description: administrateur LDAP
userPassword: test

dn: ou=Machines,o=salle408,dc=RT,dc=tp
ou: Machines
objectClass: organizationalUnit
objectClass: top

dn: ou=People,o=salle408,dc=RT,dc=tp
ou: People
objectClass: organizationalUnit
objectClass: top

dn: ou=permanents,ou=People,o=salle408,dc=RT,dc=tp
ou: permanents
objectClass: organizationalUnit
objectClass: top

dn:: b3U9w6l0dWRpYW50cyxvdT1QZW9wbGU
        sbzlzYWxsZTQwOCxkYzlSVCxkYz10cA==
ou:: w6l0dWRpYW50cw==
objectClass: organizationalUnit
objectClass: top
```

6 dn $\Rightarrow$ 6 objets (certains écrits en base64)

Construction d'une base

On fait le schéma de la base voulue ....comme on veut

- Schéma appelé DIT : Directory Information Tree

Par contre, un objet de la base hérite de classes d'objets pré-existantes (on ne peut pas en définir!)

- Attributs obligatoires
- Attributs optionnels
- Notion d'héritages entre classes

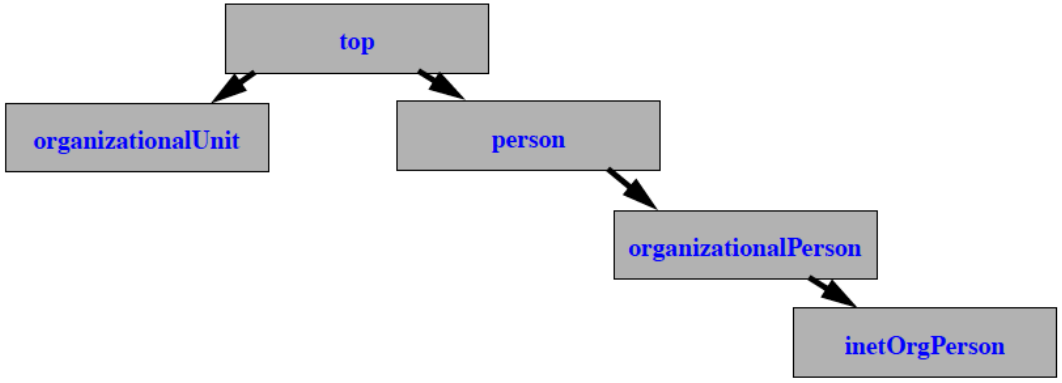
Classes d'objets

Exemples de classes d'objets

Entry Type	Required Attributes	Optional Attributes
inetOrgPerson (defines entries for a person)	•commonName (cn) •surname (sn) •objectClass	•businessCategory •carLicense •departmentNumber • description •employeeNumber •facsimileTelephone •Number •givenName •mail •manager •mobile •organizationalUnit (ou) •pager •postalAddress •roomNumber •secretary •seeAlso •telephoneNumber •title •labeledURI •uid
organizationalUnit (defines entries for organizational units)	•ou •objectClass	•businessCategory •description •facsimileTelephoneNumber •location (l) •postalAddress •seeAlso •telephoneNumber

Héritage entre classes

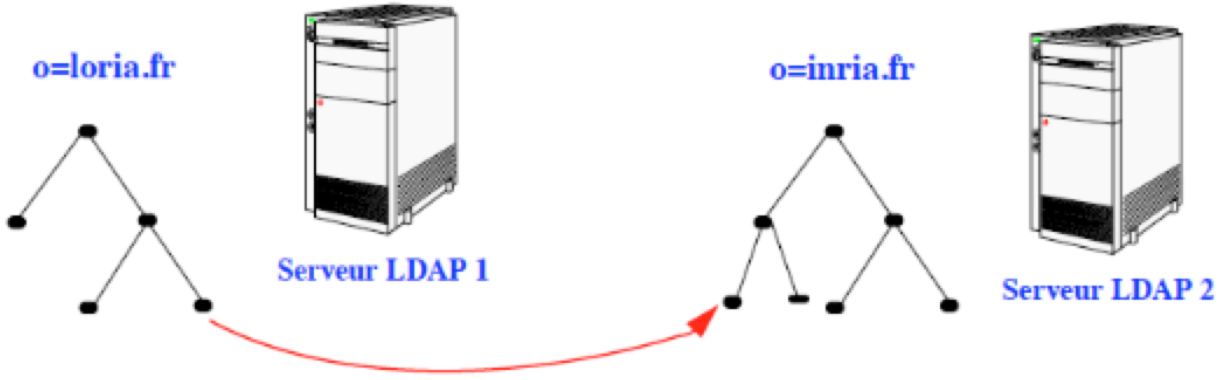
Les classes d'objets forment une hiérarchie, au sommet de laquelle se trouve l'objet `top`.



- **Chaque objet hérite des propriétés (attributs) de l'objet dont il est le fils.**
- **On précise la classe d'objet d'une entrée à l'aide de l'attribut `objectClass`.**
- **Il faut obligatoirement indiquer la parenté de la classe d'objet en partant de l'objet `top` et en passant par chaque ancêtre de l'objet.**

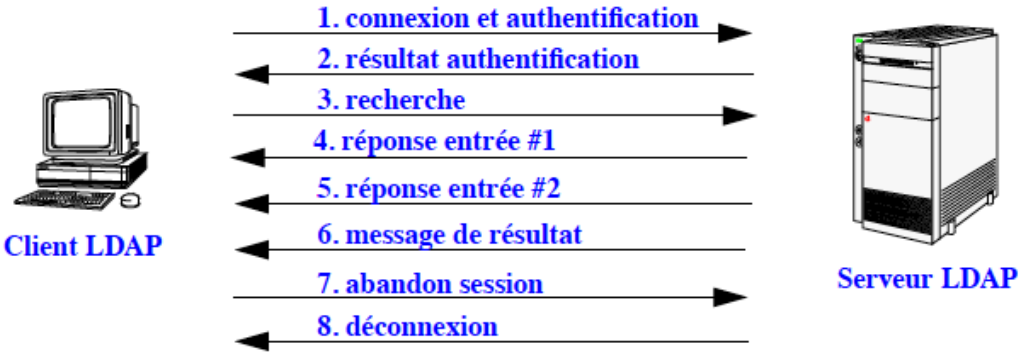
La base peut être distribuée - entreprise multi-sites

Le DIT est découpé en sous-arbre, chacun géré par un serveur distinct



Procole LDAP

Il faut se connecter avant d'accéder aux données



Authentification

L'authentification

LDAP est un protocole avec connexion : il faut s'authentifier pour ouvrir la connexion (`bind`) en fournissant une identité.

LDAPv3 propose plusieurs choix d'authentification :

- ***Anonymous authentification*** **- accès sans authentification permettant de consulter les données accessibles en lecture pour tous.**
- ***Root DN authentification*** **- accès administrateur (tous les droits).**
- ***Mot de passe en clair*** **- un DN plus un password qui transite en clair sur le réseau.**
- ***Mot de passe + SSL ou TLS*** **- la session est chiffrée et le mot de passe ne transite plus en clair.**
- ***Certificats sur SSL*** **- échange de certificats SSL (clefs publiques/privées).**
- ***Simple Authentification and Security Layer (SASL)*** **- mécanisme externe d'authentification.**

Contrôle d'accès (aux données)

Une fois authentifié, on peut controller l'acccès aux données, c.a.d. ce que l'utilisateur voit : tout ou partie du DIT, en lecture, écriture, etc.

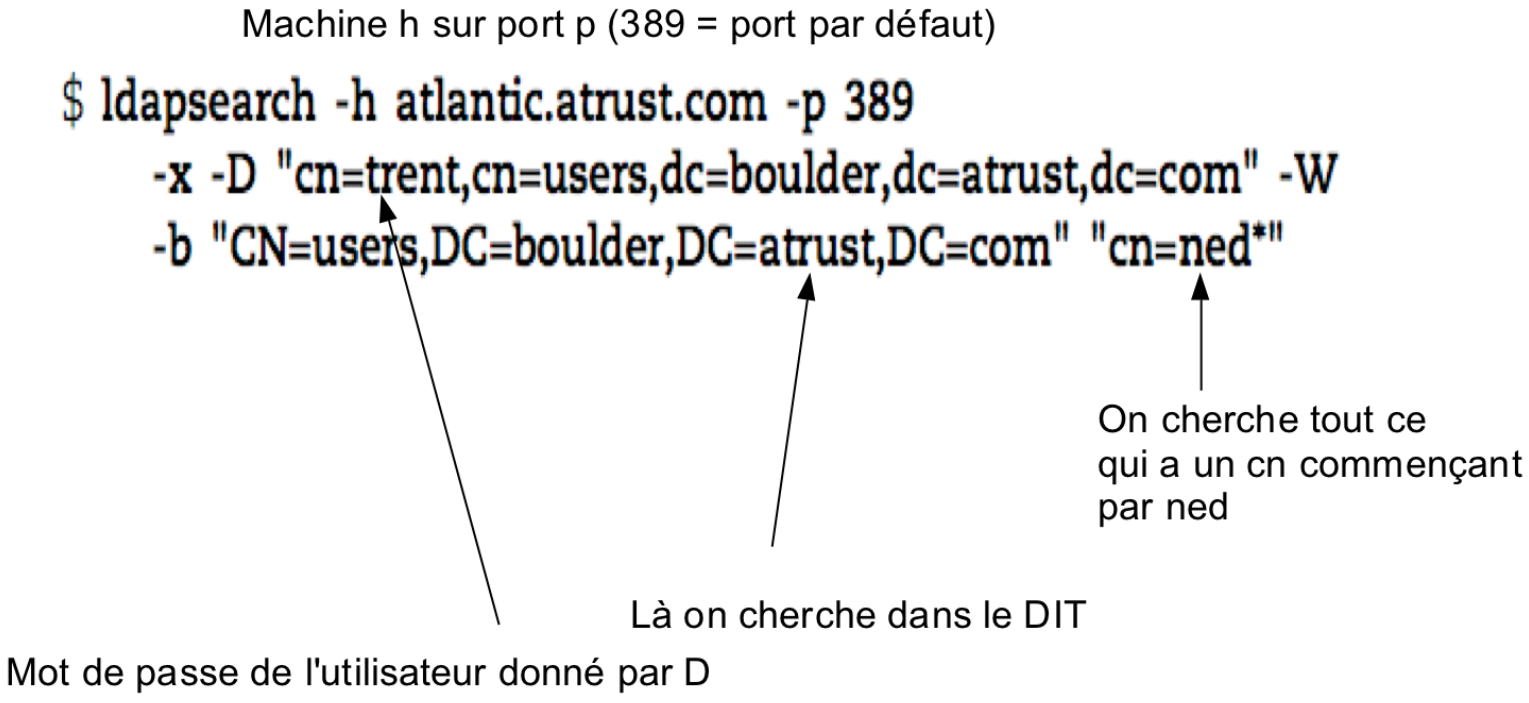
Ces ACLs s'expriment sous la forme canonique :

<target> <permission> <bind rule>

<target> : point d'entrée de l'annuaire auquel s'applique la règle
<permission> : permet ou refuse un type d'accès (lecture, écriture...)
<bind rule> : identifie le bindDN utilisé en connexion

<permissions>	<bind rules>
Read	Un utilisateur
Write	Un groupe d'utilisateur
Search	
Compare	
Selfwrite	
Add	
Delete	

Exemple d'accès aux données avec utilitaire ldapsearch



Comme un utilitaire, par ex. ssh, est raccordé à LDAP?

PAM: Pluggable Authentication Module

- Idée derrière PAM : une librairie dynamique pour les commandes/démons et un fichier de config par commande/démon. Délégation de l'authentification
- Seul le fichier de config (texte) doit être modifié – plus de recompilation!
- Comment savoir si une commande est pam-ifiée : ldd nom_commande | grep pamlib.so

```
root@stretch:/home/vagrant# ldd /usr/bin/passwd
        linux-vdso.so.1 (0x00007ffc6b282000)
        libpam.so.0 => /lib/x86_64-linux-gnu/libpam.so.0 (0x00007fdd6a421000)
        libpam_misc.so.0 => /lib/x86_64-linux-gnu/libpam_misc.so.0 (0x00007fdd6a21d000)
        libaudit.so.1 => /lib/x86_64-linux-gnu/libaudit.so.1 (0x00007fdd69ff5000)
        libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1 (0x00007fdd69dcd000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fdd69a2e000)
        libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007fdd6982a000)
        libcap-ng.so.0 => /lib/x86_64-linux-gnu/libcap-ng.so.0 (0x00007fdd69624000)
        libpcre.so.3 => /lib/x86_64-linux-gnu/libpcre.so.3 (0x00007fdd693b1000)
        /lib64/ld-linux-x86-64.so.2 (0x00007fdd6a83e000)
        libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007fdd69194000)
```

PAM : fichier de configuration

- Stockés dans le repertoire /etc/pam.d.
- Un fichier contient 3 parties en général : passwd, auth, session, account

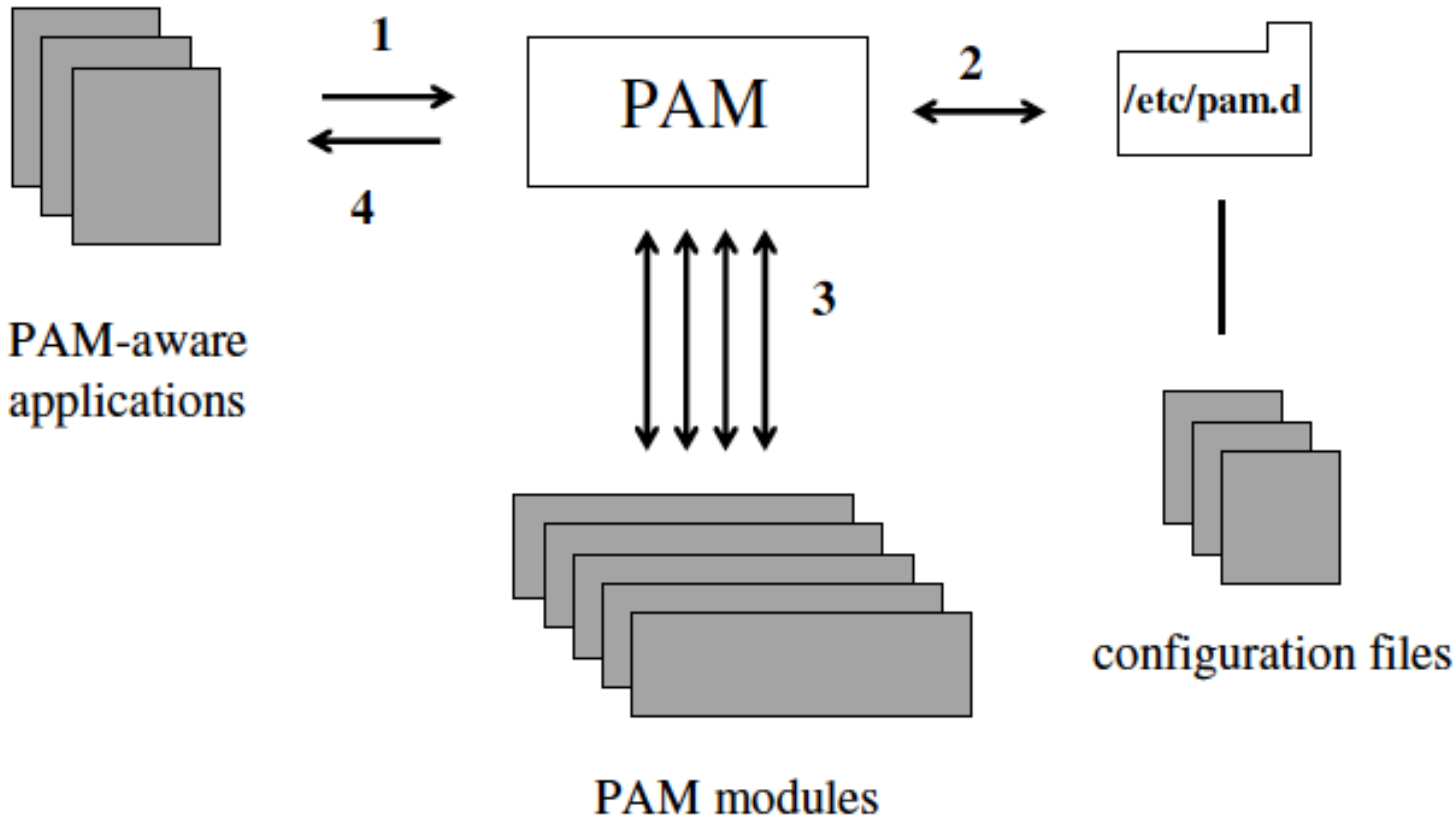
› Passwd : changement de mot de passe
› Auth : vérification du couple (nom, mdPasse)
› Account : vérification des éléments du compte (horaires, droit, ...)
› Session : mise en place de la session

PAM : fichier de configuration

- Chaque partie contient des lignes du type
  auth sufficient pamrootok.so
- En fin de ligne, on trouve un module PAM qui va retourner : succès ou erreur
- Le « control-flag » qui vaut sufficient, required,requisite, optional, etc
- A chaque invocation, retour d'une seule valeur : succès ou erreur

```
guillaumesmbp2:pam.d urvoy$ ls
authorization           authorization_lacont    login                   screensaver             smbd
authorization_aks       checkpw                 login.term              screensaver_aks         sshd
authorization_ctk       chkpasswd               other                   screensaver_ctk         su
authorization_la        cups                    passwd                  screensaver_la          sudo
guillaumesmbp2:pam.d urvoy$ more passwd
# passwd: auth account
auth       required       pam_permit.so
account    required       pam_opendirectory.so
password   required       pam_opendirectory.so
session    required       pam_permit.so
```

Architecture PAM



© David Morgan 2006

NSswicth

- Name Service Switch
- Wikipedia : autorise le remplacement des traditionnels fichiers Unix de configuration (par exemple /etc/passwd, /etc/group, /etc/hosts) par une ou plusieurs bases de données centralisées (ldap, etc)

```
root@stretch:/etc/pam.d# more /etc/nsswitch.conf
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch functionality.
# If you have the `glibc-doc-reference' and `info' packages installed, try:
# `info libc "Name Service Switch"' for information about this file.

passwd:         compat
group:          compat
shadow:         compat
gshadow:        files

hosts:          files dns
networks:       files

protocols:      db files
services:       db files
ethers:         db files
rpc:            db files

netgroup:       nis
```