R502
Supervision des réseaux

Sami Evangelista
IUT de Villetaneuse
Département Réseaux et Télécommunications
2023–2024

http://www.lipn.univ-paris13.fr/~evangelista/cours/R502

Contenu du module

R502 — Supervision des réseaux
http://www.lipn.univ-paris13.fr/~evangelista/cours/R502

- Volume horaire :
  - 2×3h de cours/TD
  - 6×3h de TP
  - 2h de contrôle
- Évaluation :
  - TP notés
  - contrôle

Plan

1. Introduction à la supervision des réseaux

2. SNMP : protocole de supervision

3. Centralisation de journaux avec syslog

4. Éléments de qualité de service

Supervision des réseaux — Une définition

Ensemble des techniques permettant de s'assurer du bon fonctionnement des éléments d'un réseau.

- superviser ⇔ surveiller
- tâche dévolue à l'administrateur du réseau
- éléments d'un réseau :
  - switchs, routeurs
  - ordinateurs
  - imprimantes
  - services (HTTP, FTP, DHCP, ...)
    ...
  - En gros, tout équipement qui a une @IP (plus éventuellement un port).

Pourquoi superviser ?

- Assurer la continuité des services.
  - Détecter les pannes pour remettre les équipements en service
- Mesurer les performances du réseau
  - Assurer une qualité de service
  - Analyser le trafic (reporting)
- Détecter/prévenir les attaques
  - ex : avec un système de détection des intrusions (IDS)

Exemples d'application de la supervision

1. Surveillance d'un serveur de mail
   - Un processus vérifie périodiquement l'état du disque dur du serveur de mail.
   - espace disque libre $< 1Go \Rightarrow$ il envoie un mail aux utilisateurs pour leur demander de faire le ménage dans leurs mails
2. Surveillance d'une imprimante
   - Toutes les 10 sec. on envoie un ping à une imprimante.
   - 5 pings successifs négatifs $\Rightarrow$ mail envoyé à l'administrateur
3. Blocage d'un attaquant
   - Sur la passerelle d'accès au réseau on journalise les tentatives de connexion SSH.
   - Un processus inspecte périodiquement le journal.
   - Il bloque les @IP (via l'ajout d'une règle de pare-feu) qui apparaissent trop fréquemment (p.ex., plus de 100 tentatives de connexion en une minute).
4. Prévision du trafic
   - On enregistre dans une base de données les dates des requêtes envoyées au serveur Web.
   - On peut ensuite en extraire des statistiques pour prévoir les pics d'affluence.

Approches de supervision

Supervision par interrogation (polling)

- à l'initiative du système de supervision
- Il interroge périodiquement les équipements supervisés sur leur état.
- ex : envoyer un ping toutes les minutes à l'imprimante

Supervision par remontée d'incidents

- à l'initiative des équipements supervisés
- En cas d'événement exceptionnel, l'équipement supervisé prévient le système de supervision.
- ex : si un service meurt, le système sur lequel il s'exécutait envoie un message au système de supervision

Approches de supervision — Comparaison

- La supervision par polling nécessite plus de bande passante et plus de traitements des deux côtés (superviseur et supervisé).
- Elle peut passer à côté d'incidents brefs (p.ex., si un service plante puis redémarre sans que le superviseur n'ait interrogé le supervisé entre-temps).
- Avec la méthode par remontée, le superviseur est prévenu dès que possible.
- Mais la supervision par polling peut détecter davantage de problèmes (p.ex., problème de réseau (équipement supervisé inaccessible), si la machine supervisée s'est arrêtée brusquement, ...).

  $\Rightarrow$ intérêt de combiner les deux

Comment superviser ?

- Outils généralistes
  - ping
  - traceroute
  - iperf
  - netstat
  - wireshark
  - ...
- Protocoles de supervision
  - CMIP (Common Management Information Protocol)
    - normalisé par l'OSI, peu utilisé
  - SNMP* (Simple Network Management Protocol)
    - simple à mettre en œuvre ⇒ très répandu
- Outils dédiés
  - Nagios*
  - Centreon
  - Zabbix
  - ...

⋆ : vu dans ce module

Un outil de supervision : Nagios

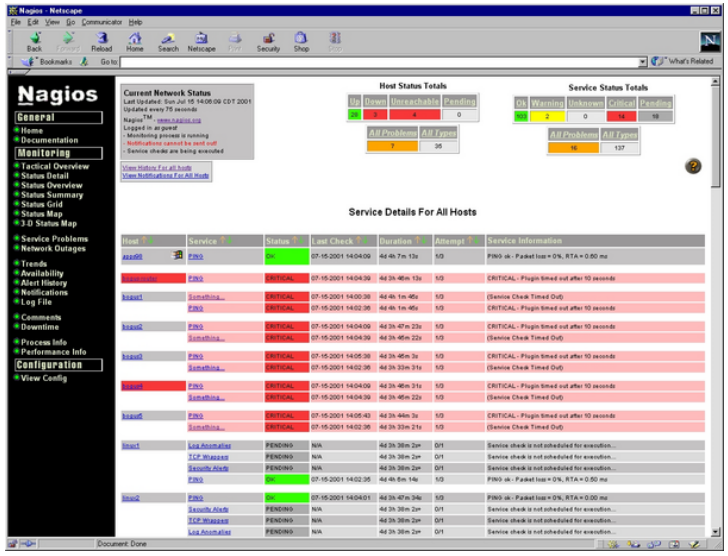
`https://www.nagios.com/`

- logiciel libre
- première version en 1996
- 2 composants :
  - un moteur qui exécute périodiquement des tâches de supervision ;
  - et une interface web pour visualiser les résultats collectés par le moteur.
- quelques fonctionnalités importantes :
  - possibilité de superviser
    - des équipements (p.ex., vérifier qu'une machine répond au pings)
    - des services (p.ex., vérifier qu'apache est actif sur un PC)
    - des ressources (espace disque, charge du processeur, ...)
  - cartographie du réseau
  - mise en places d'alerte (envoi de mail/SMS en cas de problème)
- possibilité de développer des plugins réalisant des tâches de supervision
- plusieurs forks : icinga, shinken, naemon

Plus de détails en TP.

Interface web de Nagios

Vue en temps réel des services disponibles ou à l'arrêt.



Source : `https://fr.wikipedia.org/wiki/Nagios`, consultée le 10/09/2023.

Plan

1. Introduction à la supervision des réseaux
2. SNMP : protocole de supervision
3. Centralisation de journaux avec syslog
4. Éléments de qualité de service

2. SNMP : protocole de supervision
2.1 Introduction à SNMP
2.2 Les MIBs
2.3 La boîte à outils net-snmp
2.4 Structure des messages SNMP
2.5 De SNMPv1 à SNMPv3

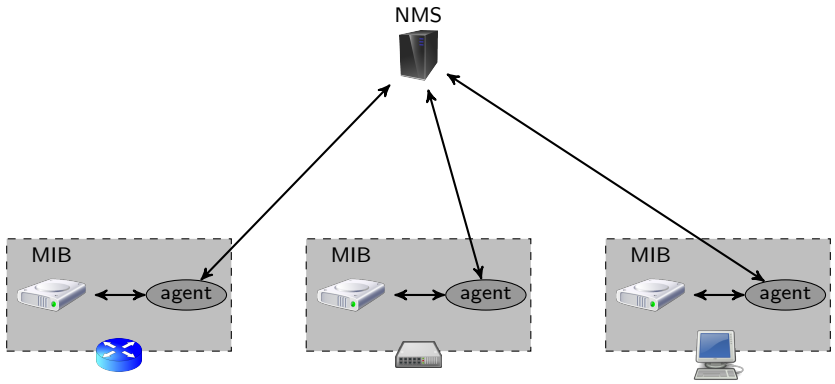
Présentation de SNMP

- SNMP = Simple Network Management Protocol
- protocole défini par l'IETF
- 3 versions définies dans différentes RFC :
  - v1 RFC 1065–1067 (1988) — toujours utilisée
  - v2c RFC 1901–1908 (1996) — version la plus répandue
    (Plusieurs v2 ont été définies : v2u, v2p, ... v2c est celle utilisée actuellement.)
  - v3 RFC 3411–3418 (2002)
- SNMP utilise (généralement) UDP pour le transport.
- Ports utilisés (par défaut) :
  - 161 = port utilisé par les agents SNMP
  - 162 = port de destination des notifications
- **Dans ce cours, on se basera sur la v2c.**
  (cf. section 2.5 pour une description des différences entre les trois versions)

Terminologie

- SNMP fait intervenir :
  - un **NMS** (Network Management System) — la plate-forme de supervision ;
  - et des **agents** — processus s'exécutant sur les équipements supervisés.
- Chaque agent a en mémoire un ensemble d'objets qui peuvent être lus/modifiés par le NMS.
- Ces objets donnent des informations sur l'état de l'équipement sur lequel s'exécute l'agent et peuvent être utilisés pour sa supervision.
- Ils forment une **MIB** : une Management Information Base.
- Les objets de cette MIB sont décrits dans le langage SMI (cf. les exemples dans la suite).
  SMI = Structure of Management Information
- Exemples d'objets :
  - états des interfaces réseau (activée/désactivée)
  - température interne
  - espace disque libre

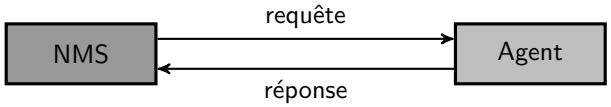
L'architecture SNMP



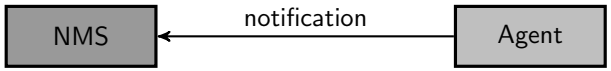
SNMP est mis en œuvre sur la plupart des équipements réseau (routeurs, switchs, ...) et systèmes d'exploitation.

Les opérations SNMP

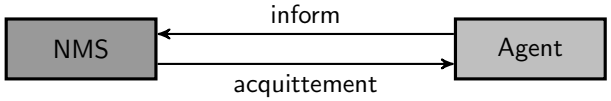
- les requêtes



  - permet de lire/modifier la (les) valeur(s) d'un (de plusieurs) objet(s)
- les notifications (ou traps)



  - permet de prévenir le NMS d'un événement particulier
    Exemple : l'OS va s'arrêter.
  - pas d'acquittement
- les informs



  - notification avec acquittement

Les requêtes SNMP

- **getRequest**
  récupère la valeur d'un (de plusieurs) objet(s)
- **setRequest**
  modifie la valeur d'un (de plusieurs) objet(s)
- **getNextRequest**
  récupère la valeur de l'objet suivant dans la MIB
- **getBulkRequest**
  récupère les valeurs d'un groupe d'objets consécutifs de la MIB

Les communautés SNMP

- Une communauté désigne un ensemble d'agents/NMS auxquels sont associés des droits.
- communauté $\approx$ mot de passe
- Trois types de droits sont associés à une communauté :
  - readOnly — droit en lecture sur les objets de la MIB
  - readWrite — droit en lecture/écriture sur les objets de la MIB
  - trap — droit d'envoyer des traps
- La communauté est présente dans tous les messages SNMP.
- L'échange se fait en clair (non chiffré) sur le réseau.

  $\Rightarrow$ En voyant passer des messages SNMP on peut découvrir les communautés.

  $\Rightarrow$ à changer régulièrement

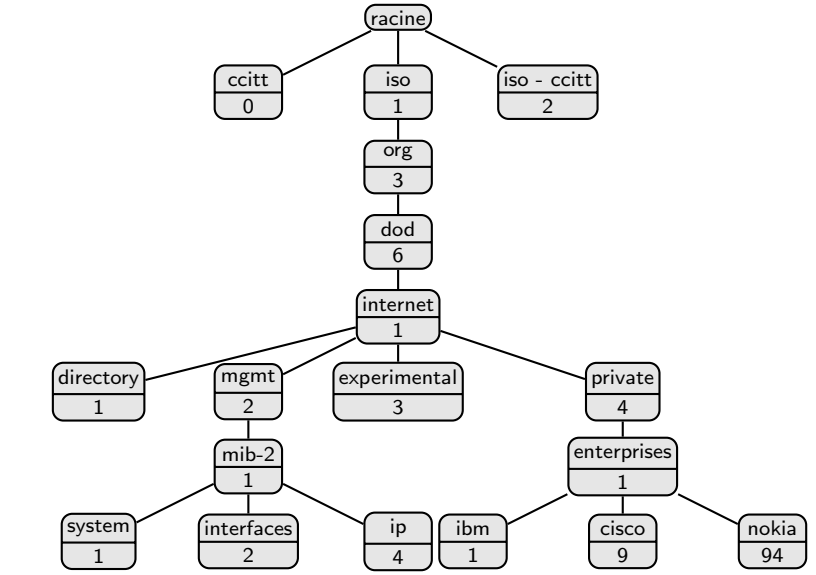
Plan

2. SNMP : protocole de supervision
2.1 Introduction à SNMP
2.2 Les MIBs
2.3 La boîte à outils net-snmp
2.4 Structure des messages SNMP
2.5 De SNMPv1 à SNMPv3

Structure des MIBs

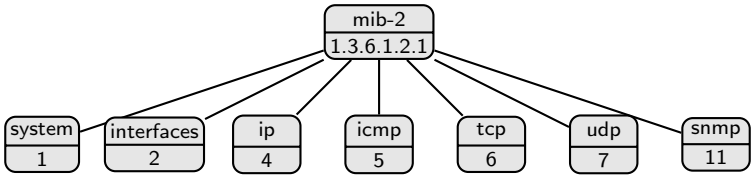
- MIB = base de données d'objets présente sur un équipement supervisé par un agent SNMP
- Les MIBs ont une structure arborescente.
- Cette structure est **normalisée** (en partie) et **extensible** (en partie).
- Un objet est un nœud de l'arbre.
- Chaque objet
  - a un nom symbolique et un numéro ;
  - et est identifié par un **OID** (object identifier) unique.
- OID d'un objet O = liste des numéros des objets sur le chemin de la racine à O séparés par des '.'
- Les agents ne possèdent pas tous la même MIB !

Représentation d'une MIB



OID = 1.3.6.1.2.1.1

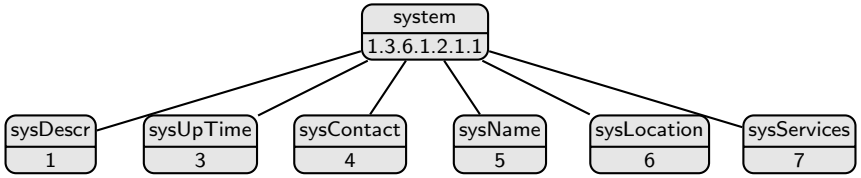
La branche mib-2



branche **normalisée** et **présente sur tous les agents SNMP** (en théorie !)

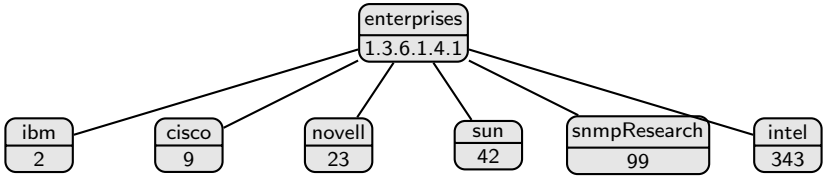
- system (1.3.6.1.2.1.1)
  informations générales sur l'équipement sur lequel se trouve l'agent
- interfaces (1.3.6.1.2.1.2)
  informations sur les interfaces réseau (ex : statuts des interfaces)
- ip (1.3.6.1.2.1.4)
  informations sur la pile IP (ex : contenu de la table de routage)
- tcp (1.3.6.1.2.1.6), udp (1.3.6.1.2.1.7), snmp (1.3.6.1.2.1.11)
  statistiques TCP, UDP et SNMP (ex : connexions TCP ouvertes)

La branche system



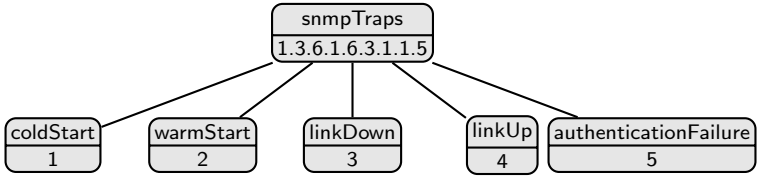
- sysDescr (1.3.6.1.2.1.1.1)
  description textuelle
- sysUpTime (1.3.6.1.2.1.1.3)
  temps (exprimé en $\frac{1}{100}$ de sec.) depuis lequel l'équipement est en marche
- sysContact (1.3.6.1.2.1.1.4)
  adresse mail de l'administrateur de l'équipement
- sysName (1.3.6.1.2.1.1.5)
  nom de l'équipement
- sysLocation (1.3.6.1.2.1.1.6)
  emplacement physique de l'équipement
- sysServices (1.3.6.1.2.1.1.7)
  services rendus par l'équipement

La branche enterprises



- Branche dans laquelle les entreprises peuvent insérer des objets spécifiques à leurs équipements.
- Chaque objet fils de enterprises est géré par la compagnie correspondante.
- Ces MIBs sont dites privées.

La branche snmpTraps



branche contenant la définition des notifications SNMPv2c standard
(Les notifications sont aussi définies comme des objets de la MIB.)

- coldStart (1.3.6.1.6.3.1.1.5.1)
  redémarrage de l'agent avec réinitialisation de la configuration
- warmStart (1.3.6.1.6.3.1.1.5.2)
  redémarrage de l'agent sans changement de configuration
- linkDown (1.3.6.1.6.3.1.1.5.3)
  interface désactivée
- linkUp (1.3.6.1.6.3.1.1.5.4)
  interface activée
- authenticationFailure (1.3.6.1.6.3.1.1.5.5)
  pb. d'authentification (p.ex., requête reçue avec une mauvaise communauté)

Le langage de description SMI — Exemple 1

Définition de l'objet `system`.

```
iso OBJECT IDENTIFIER ::= { 1 }
org OBJECT IDENTIFIER ::= { iso 3 }
dod OBJECT IDENTIFIER ::= { org 6 }
internet OBJECT IDENTIFIER ::= { dod 1 }
mgmt OBJECT IDENTIFIER ::= { internet 2 }
mib-2 OBJECT IDENTIFIER ::= { mgmt 1 }
system OBJECT IDENTIFIER ::= { mib-2 1 }
```

Le langage de description SMI — Exemple 2

Définition de l'objet `sysUpTime`, fils n°3 de l'objet `system`.

```
sysUpTime OBJECT-TYPE
    SYNTAX  TimeTicks
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The time (in hundredths of a second) since the
         network management portion of the system was last
         re-initialized."
    ::= { system 3 }
```

Le langage de description SMI — Exemple 3

Définition de l'objet `sysContact`, fils n°4 de l'objet `system`.

```
sysContact OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The textual identification of the contact person
         for this managed node, together with information
         on how to contact this person."
    ::= { system 4 }
```

Le langage de description SMI — Exemple 4

Définition de la notification `coldStart`, fils n°1 de l'objet `snmpTraps`.

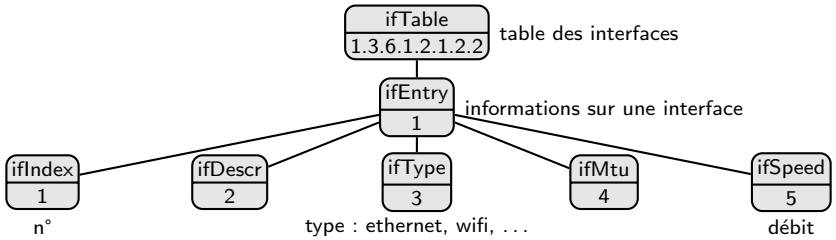
```
coldStart NOTIFICATION-TYPE
    STATUS  current
    DESCRIPTION
        "A coldStart trap signifies that the SNMP entity,
         supporting a notification originator application, is
         reinitializing itself and that its configuration may
         have been altered."
    ::= { snmpTraps 1 }
```

Objets scalaires et objets tabulaires

- objet scalaire ⇒ objet ayant une seule instance
  - par exemple : les objets de la branche `system` : `sysName`, `sysLocation`, `sysUpTime`, ...
  - L'OID de l'objet doit être suivi de ".0" dans les requêtes et réponses SNMP.
- objet tabulaire ⇒ objet ayant plusieurs instances
  - Les objets tabulaires sont définis avec le type `SEQUENCE`.
  - L'OID de l'objet doit être suivi de ".i" dans les requêtes et réponses SNMP avec `i` = numéro d'instance.

Exemple d'objet tabulaire : `IfTable`

`IfTable` : branche contenant les informations sur les interfaces réseau




```
ifTable OBJECT-TYPE
   SYNTAX SEQUENCE OF IfEntry ... ::= { interfaces 2 }
ifEntry OBJECT-TYPE
   SYNTAX IfEntry ... ::= { ifTable 1 }
ifDescr OBJECT-TYPE
   SYNTAX DisplayString (SIZE (0..255)) ... ::= { ifEntry 2 }
```

On a donc :

- 1.3.6.1.2.1.2.2.1.5.**1** = débit (`ifSpeed`) de la 1ère interface réseau
- 1.3.6.1.2.1.2.2.1.3.**2** = type de la 2ère interface réseau

2. SNMP : protocole de supervision
2.1 Introduction à SNMP
2.2 Les MIBs
2.3 La boîte à outils net-snmp
2.4 Structure des messages SNMP
2.5 De SNMPv1 à SNMPv3

Présentation de net-snmp

- boite à outils linux contenant :
  - des commandes d'interrogation d'agents
  - un agent SNMP
- principales commandes :
  - `snmpset`, `snmpget`, `snmpgetnext` et `snmpbulkget` : envoi de requêtes
  - `snmptrap` : envoi de notifications
  - `snmpinform` : envoi de informs
  - `snmpwalk` et `snmpbulkwalk` : exploration d'une branche de la MIB
    - Envoi de plusieurs getRequest (pour `snmpwalk`) ou getBulkRequest (pour `snmpbulkwalk`).
- Options requises par toutes les commandes :
  - `-v V`
    version de SNMP utilisée (V = 1, 2c ou 3)
  - `-c COMMUNAUTE`
    communauté utilisée

Utilisation de `snmpset` et `snmpget`

Pour récupérer la valeur de sysName :

```
$ snmpget -v2c -cpublic localhost .1.3.6.1.2.1.1.5.0
SNMPv2-MIB::sysName.0 = STRING: localhost.localdomain
```

Pour modifier sa valeur :

```
$ snmpset -v2c -cprivate localhost .1.3.6.1.2.1.1.5.0 s "pc-de-sami"
SNMPv2-MIB::sysName.0 = STRING: pc-de-sami
```

La valeur de certains objets n'est pas modifiable (p.ex., sysDescr) :

```
$ snmpset -v2c -cprivate localhost .1.3.6.1.2.1.1.1.0 s "pc-de-sami"
Error in packet.
Reason: notWritable (That object does not support modification)
Failed object: SNMPv2-MIB::sysDescr.0
```

On peut aussi récupérer (ou modifier) plusieurs objets avec une seule requête.

```
$ snmpget -v2c -cpublic localhost .1.3.6.1.2.1.1.4.0 .1.3.6.1.2.1.1.6.0
SNMPv2-MIB::sysContact.0 = STRING: sami.evangelista@iutv.fr
SNMPv2-MIB::sysLocation.0 = STRING: IUTV (0108)
```

Utilisation de `snmpgetnext` et `snmpwalk`

Avec un getNextRequest, on récupère l'objet suivant dans la MIB (via un parcours de l'arbre) :

```
$ snmpgetnext -v2c -cpublic -On localhost .1.3.6.1
.1.3.6.1.2.1.1.1.0 = STRING: Linux localhost.localdomain 4.17.19-200.fc28.x86
$ snmpgetnext -v2c -cpublic -On localhost .1.3.6.1.2.1.1.3.0
.1.3.6.1.2.1.1.4.0 = STRING: sami.evangelista@iutv.fr
```

(Option `-On` = sortie numérique (affiche les OID plutôt que les noms))

`snmpwalk` permet de retrouver tous les objets dont l'OID est prefixé par un OID donné en argument :

```
$ snmpwalk -v2c -cpublic -On localhost .1.3.6.1.2.1.1
.1.3.6.1.2.1.1.1.0 = STRING: Linux localhost.localdomain 4.17.19-200.fc28.x86
.1.3.6.1.2.1.1.2.0 = OID: .1.3.6.1.4.1.8072.3.2.10
.1.3.6.1.2.1.1.3.0 = Timeticks: (661096) 1:50:10.96
.1.3.6.1.2.1.1.4.0 = STRING: sami.evangelista@iutv.fr
.1.3.6.1.2.1.1.5.0 = STRING: pc-de-sami
...
```

Remarques sur `snmpwalk` et `snmpbulkwalk`

- `snmpwalk` et `snmpbulkwalk` sont des commandes permettant d'explorer une branche de la MIB.
- Mais ce ne sont pas des requêtes du protocole.
- Elles utilisent des requêtes **getNextRequest** (pour `snmpwalk`) et **getBulkRequest** (pour `snmpbulkwalk`) pour leur exploration.
- Les seules requêtes SNMP sont : set, get, getNext, et getBulk.

Plan

2. SNMP : protocole de supervision
2.1 Introduction à SNMP
2.2 Les MIBs
2.3 La boîte à outils net-snmp
2.4 Structure des messages SNMP
2.5 De SNMPv1 à SNMPv3

Le schéma d'encodage TLV

- schéma d'encodage utilisé pour tous les messages SNMP
- Tout champ est décomposé en trois parties :

Type	Length	Value
1 o.	1 o.	N o.

- Type (1 o.) = type du champ (code normalisé)
- Length (1 o.) = nb. d'octets dans la partie Value
- Value (N o.) = valeur réelle du champ

Les types primitifs

- type primitif ⇔ Le champ Value ne peut pas être redécomposé en sous-champs.

Type	Code hexadécimal
Integer	0x02
String	0x04
Null	0x05
OID	0x06
IpAddress	0x40
Counter	0x41
Gauge	0x42
TimeTicks	0x43

Exemples

- `02 01 04` = un entier (02) codé sur un octet (01) et valant 4
- `04 02 79 6F` = une chaîne de caractères (04) de deux (02) octets valant "yo" (`79 6F` en ASCII)

Les types composite

- type composite ⇔ Le champ Value est une suite de sous-champs (eux-mêmes encodés en TLV).

Type	Code hexadécimal
sequence	0x30
getRequest	0xA0
getNextRequest	0xA1
response	0xA2
setRequest	0xA3
getBulkRequest	0xA5
informRequest	0xA6
trap	0xA7

- Le contenu d'un message est donc encodé en TLV.

Exemple

- `30 07 02 01 04 04 02 79 6F` = une séquence (30) de 7 octets contenant l'entier 4 et la chaîne de caractères "yo"

Encodage des OID

Règle de codage des 2 premiers nombres de l'OID

- Les deux premiers nombres $x.y$ d'un OID sont codés sur 1 octet valant $40x + y$.
- Exemples : la valeur de l'OID 1.3.4.5 sera : `2B 04 05`
  - `2B` $= 2 \times 16 + 11 = 43 = 40 \times \mathbf{1} + \mathbf{3}$

Règle de codage des grands nombres (> 127)

- Problème : comment coder 400 dans un OID ?
  - 400 est codé sur deux octets : `01 90` en hexadécimal.
  - Or `01 90` pourrait être interprété comme deux nombres : 1.144.
- Solution utilisée :
  - Seuls les 7 bits de poids faible sont utilisés pour coder les nombres.
  - Le $8^{\text{ème}}$ bit (de poids fort) restant vaut
    - 0 si l'octet est le dernier apparaissant dans le nombre
    - 1 si l'octet suivant fait aussi partie du nombre.
- Exemple : $400 = 11\ 0010000_2$

  $\Rightarrow$ 400 sera codé **1**0000011 **0**0010000 en binaire, soit `83 10` en hexa.

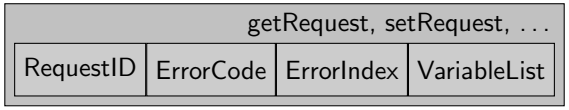
Structure générique

- Les messages SNMP ont tous la même structure.
- Tous les champs sont encodés selon le schéma TLV.
- Un message est encodé comme une séquence (au sens TLV) de 3 éléments.

- Version = 0 pour SNMPv1, 1 pour SNMPv2c, 3 pour SNMPv3
- Community = chaîne de caractères contenant la communauté.
- PDU = contenu du message

Le PDU SNMP

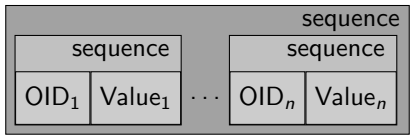
- Structure du PDU identique quel que soit le type du message.
- On utilise les types composite getRequest, setRequest, ... pour déterminer le type du message.



- RequestID = identifiant de requête
  - choisi aléatoirement par l'émetteur de la requête
  - et recopié dans la réponse
- ErrorCode = code d'erreur
  - toujours à 0 (pas d'erreur) dans une requête
- ErrorIndex = numéro de l'OID dans VariableList qui a causé l'erreur (si ErrorCode $\neq 0$)
- VariableList = séquence de couples (OID, valeur)

Encodage de la liste d'objets

- Un message SNMP contient toujours une liste (séquence) d'OID avec leurs valeurs.
- Chaque couple (OID, valeur) est lui-même encodé comme une séquence.



Valeurs associées aux objets envoyés

- Dans les requêtes set, les valeurs sont celles que la requête tente d'affecter aux objets.
- Dans les requêtes get, les valeurs sont toujours vides (type Null).
- Dans les réponses (qu'elle que soit la requête envoyée) les valeurs sont celles des objets de la requête (après modification dans le cas d'un set).

Codes d'erreur

Quelques codes courant :

- `0x00` noError — pas d'erreur
- `0x01` tooBig — réponse trop longue pour être transportée
- `0x06` noAccess — pas d'accès à l'objet demandé (p.ex., la communauté n'a pas le droit de modifier l'objet demandé)
- `0x07` wrongType — la valeur fournie dans une requête set n'a pas le bon type
- `0x11` notWritable — tentative de modification d'une variable accessible en lecture seule

Exemple de message getRequest


```
▸ Frame 7: 99 bytes on wire (792 bits), 99 bytes captured (792 bits) on interface 0
▸ Ethernet II, Src: Dell_3e:67:dd (b8:ac:6f:3e:67:dd), Dst: Cisco_67:90:40 (1c:1d:86:67:90:40)
▸ Internet Protocol Version 4, Src: 192.168.15.24, Dst: 192.168.15.100
▸ User Datagram Protocol, Src Port: 46864, Dst Port: 161
▾ Simple Network Management Protocol
    version: v2c (1)
    community: commRO
  ▾ data: get-request (0)
    ▾ get-request
        request-id: 1713239524
        error-status: noError (0)
        error-index: 0
      ▾ variable-bindings: 2 items
        ▸ 1.3.6.1.2.1.1.4.0: Value (Null)
        ▸ 1.3.6.1.2.1.1.6.0: Value (Null)
```

2 objets demandés : sysContact et sysLocation

- message getRequest envoyé par la commande :

```
$ snmpget -v2c -ccommRO 192.168.15.100 \
     1.3.6.1.2.1.1.4.0 \
     1.3.6.1.2.1.1.6.0
```

- getRequest ⇒ Les objets apparaissant dans le message ont la valeur Null.

Exemple de message setRequest


```
▸ Frame 5: 115 bytes on wire (920 bits), 115 bytes captured (920 bits) on interface 0
▸ Ethernet II, Src: Dell_3e:67:dd (b8:ac:6f:3e:67:dd), Dst: Cisco_67:90:40 (1c:1d:86:67:90:40)
▸ Internet Protocol Version 4, Src: 192.168.15.24, Dst: 192.168.15.100
▸ User Datagram Protocol, Src Port: 33710, Dst Port: 161
▾ Simple Network Management Protocol
    version: v2c (1)
    community: commRW
  ▾ data: set-request (3)
    ▾ set-request
        request-id: 657359147
        error-status: noError (0)
        error-index: 0
      ▾ variable-bindings: 2 items
        ▸ 1.3.6.1.2.1.1.4.0: 61646d696e406963692e6672
        ▸ 1.3.6.1.2.1.1.6.0: 49555456
```

2 objets à modifier : sysContact et sysLocation

admin@ici.fr en hexadécimal

IUTV en hexadécimal

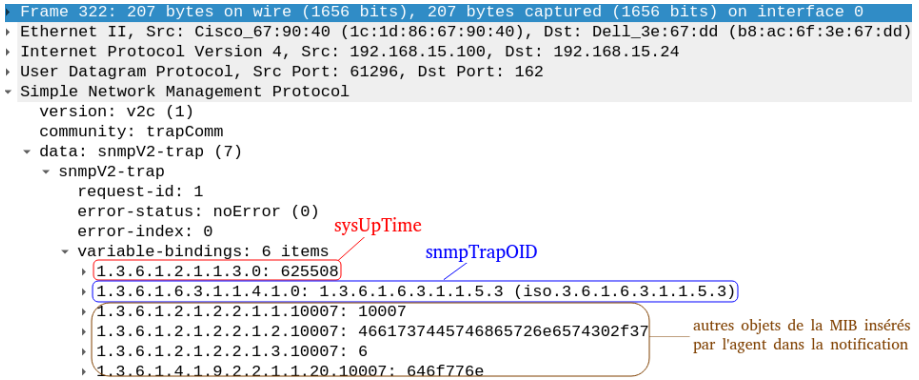
- message setRequest envoyé par la commande :

```
$ snmpset -v2c -ccommRW 192.168.15.100 \
     1.3.6.1.2.1.1.4.0 s admin@ici.fr \
     1.3.6.1.2.1.1.6.0 s IUTV
```

Encodage des notifications

- Quand un agent envoie une notification, la liste d'objets contient toujours :
  - objet n°1 : OID = 1.3.6.1.2.1.1.3 (sysUpTime)
    temps depuis lequel l'équipement est en marche
  - objet n°2 : OID = 1.3.6.1.6.3.1.1.4.1 (snmpTrapOID)
    OID de la notification envoyée
- L'agent peut aussi placer dans la notification d'autres objets pouvant donner des informations utiles au NMS.
  Par exemple :
  - sysName pour avoir le nom de l'équipement
  - sysContact pour avoir l'adresse éléctronique de l'administrateur
  - nom de l'interface désactivée dans le cas d'une notification linkDown

Exemple de notification



- 1er objet : (sysUpTime, 625508)
  ⇒ équipement démarré depuis 625508 centièmes de seconde
- 2ème objet : (snmpTrapOID, 1.3.6.1.6.3.1.1.5.3)
  ⇒ notification de type linkDown (1.3.6.1.6.3.1.1.5.3, voir diapo 26).

⇒ notification envoyée suite à la désactivation d'une interface réseau

Plan

2. SNMP : protocole de supervision
2.1 Introduction à SNMP
2.2 Les MIBs
2.3 La boîte à outils net-snmp
2.4 Structure des messages SNMP
2.5 De SNMPv1 à SNMPv3

De SNMPv1 à SNMPv3

Différences entre SNMPv1 et SNMPv2c

- nouveau type de message : inform
- nouveau type de requête : getBulkRequest
  - Avec la v1, pour récupérer plusieurs objets consécutifs de la MIB : plusieurs getNextRequest consécutifs (⇒ nombreux aller-retours).
  - Un getBulkRequest permet d'en demander plusieurs en même temps.
- structure des messages de notification simplifiée

Pas de modifications importantes entre la v1 et la v2c.

Différences entre SNMPv2c et SNMPv3

- Mécanismes de sécurité ajoutés
  - authentification
  - chiffrement des données

Plan

1. Introduction à la supervision des réseaux

2. SNMP : protocole de supervision

3. Centralisation de journaux avec syslog

4. Éléments de qualité de service

Les journaux/logs

- fichiers (généralement texte) dans /var/log
- Ils contiennent des descriptions d'événements générés par les processus (p.ex., démarrage d'un service) ou par le noyau (p.ex., gestion des périphériques).
- L'analyse des journaux est primordiale pour la maintenance et la sécurité.
  - fournit un déroulé des événements qui peuvent expliquer le problème
- Problème : nombreux fichiers de log qui peuvent être présents sur différentes machines.

Plan

3. Centralisation de journaux avec syslog
3.1 Le protocole syslog
3.2 Syslog sous linux : le service rsyslog

Syslog

- protocole de la couche applicative
- Syslog utilise (généralement) UDP pour le transport.
- défini dans les RFC 3164 et 5424
- port par défaut : 514
- objectif : centralisation de journaux sur des serveurs syslog par l'échange de messages de log
- Comme SNMP, syslog est implanté dans la plupart des OS et équipements réseau.

Structure des messages syslog

Dans la RFC3164

- Partie priorité
  - catégorie (ou *facility*) : code de 0 à 23 (voir diapos suivantes)
  - gravité : code de 0 à 7 (voir diapos suivantes)

  La priorité associée à un message est $8 \times$ catégorie $+$ gravité. On la note généralement sous la forme `catégorie.gravité`.
- Partie en-tête
  - date et heure de la génération du message
  - nom ou @IP de la machine ayant généré le message
- Partie message
  - un texte brut donnant des informations supplémentaires

Dans la RFC5424

- plus d'info. dans la partie en-tête (p.ex. nom et PID du processus ayant généré le message)
- corps du message plus structuré

Les catégories

24 catégories parmi lesquelles

- 0 = kernel — messages du noyau
- 1 = user — messages de l'espace utilisateur
- 2 = mail — messages du système de messagerie
- 3 = daemon — messages des services
- 4 = auth — messages d'authentification
- 5 = syslog — messages généré par le service syslog

Les niveaux de gravité

8 niveaux

- 0 = emergency — système inutilisable
- 1 = alert — intervention requise
- 2 = critical — conditions critiques
- 3 = error — erreur de fonctionnement
- 4 = warning — erreurs potentielles
- 5 = notice — information importante (mais pas une erreur)
- 6 = informational — information de moindre importance
- 7 = debug — message de débuggage

Plan

3. Centralisation de journaux avec syslog
3.1 Le protocole syslog
3.2 Syslog sous linux : le service rsyslog

Le service rsyslog

- Que fait-il ?
  réception de messages syslog émis par les processus locaux ou par d'autres serveurs syslog, puis éventuellement :
  - écriture du message dans un fichier
  - redirection du message vers un autre serveur
  - écriture dans un tube (⇒ permet l'exécution d'une commande)
  - ...
- fichier de configuration principal : `/etc/rsyslog.conf`
- Ce fichier contient des règles de la forme :

  `sélecteur action`

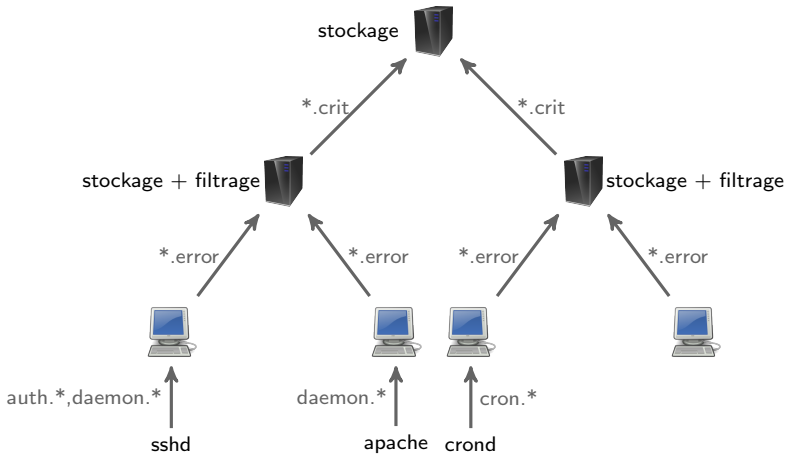
  - sélecteur : détermine les messages (selon leurs catégories et gravités) sur lesquels s'applique cette règle
  - action : ce que rsyslog fait de ces messages

  (Plusieurs règles peuvent s'appliquer pour un même message.)

Exemples de règles


```
# tous les messages de la catégorie auth avec un niveau de gravité <=
# error sont écrits dans le fichier /var/log/authentification.log
auth.error          /var/log/authentification.log
# tous les messages de la catégorie auth
auth.*              /var/log/authentification.log
# tous les messages des catégories auth et kernel
auth.*;kernel.*     /var/log/important.log
# tous les messages ('-' <=> écritures non synchrones)
*.*                 -/var/log/messages
# tous les messages sauf ceux de la catégorie auth
*.*;auth.none       -/var/log/messages
# tous les messages avec une gravité strictement égale à critique
*.=crit             /var/log/critical.log
# les messages de la cat. mail avec un niveau de gravité > notice
mail.*;mail.!notice    -/var/log/messages
# redirection vers 10.0.0.1 avec UDP (un seul @) et TCP (deux @)
syslog.*            @10.0.0.1
syslog.*            @@10.0.0.1
# messages de cat. cron redirigés vers le tube /var/run/rsylog/pipe
cron.*              |/var/run/rsylog/pipe
```

Exemple d'architecture syslog



Les tubes nommés

- Quand on fait `cmd1 | cmd2` l'OS crée un tube anonyme.
- C'est un fichier de type particulier (type pipe) dans lequel `cmd1` va écrire et `cmd2` va lire.
- On peut aussi créer des tubes nommés avec `mkfifo`.

```
$ mkfifo /tmp/test-fifo
$ ls -l /tmp/test-fifo
prw-rw-r-- 1 sami sami 0 10 déc.  10:53 /tmp/test-fifo
```

- C'est un moyen de communication entre les processus.
- Si un processus écrit (resp. lit) dans un tube il faut qu'un autre processus soit en train de lire (resp. d'écrire) dans le tube. Sinon il est en attente. Il faut qu'il y ait un "rendez-vous" entre les deux processus.
- Par exemple :

```
$ echo salut >> /tmp/test-fifo

    j'attends qu'un
    processus lise dans le fifo

$
```

```
$ cat /tmp/test-fifo | \
  while read data
  do echo $data ; done
salut
$
```

La commande logger

Elle permet d'envoyer un message au serveur syslog local.

- option -p pour préciser la priorité (par défaut user.notice)
  (Remarque : impossible d'envoyer des messages de la catégorie kernel)
- option -n pour envoyer le message à un serveur syslog distant

Exemples :

```
logger -p cron.error "une erreur cron s'est produite"
logger -n 10.0.0.1 -p auth.notice "nouvel connexion"
```

Remarques

Quels messages stocker/centraliser ?

- compromis entre les infos que l'on veut garder et les performances
- par exemple, éviter de mettre sur les clients syslog :

```
*.*     @@10.0.1.100
```

  $\Rightarrow$ réseau saturé avec tous les messages de log (même les plus insignifiants comme ceux de débuggage)
- Mais envoyer uniquement les messages les plus graves n'est pas forcèment très utile pour résoudre les problèmes.

Gestion des fichiers de logs

- utilisation du service logrotate pour organiser une rotation des logs et/ou éviter qu'ils n'occupent trop de place
- Exemple de règle logrotate :
  - stocker uniquement les logs des 4 derniers mois dans des fichiers séparés (`/var/log/syslog.1`, ... `/var/log/syslog.4`)

Plan

1. Introduction à la supervision des réseaux

2. SNMP : protocole de supervision

3. Centralisation de journaux avec syslog

4. Éléments de qualité de service

Qualité de service (QoS) — Une définition

Capacité d'un réseau à véhiculer des flux dans de bonnes conditions.

- flux = séquence de paquets d'une source vers une destination
- Critères de QoS : fiabilité, délai, gigue, bande passante.

Critères de QoS

Soit un flux entre deux points A et B du réseau.

Fiabilité

- Taux d'erreur tolérable du flux.

Délai (ou latence)

- Temps mis par un paquet pour traverser le réseau de A à B.
- Principales causes :
  - temps de traversée des équipements (routeurs)
  - temps de propagation (élevé pour les réseaux longues distances)

Gigue

- Variation dans les délais d'acheminement.
- utilisation de la variance statistique
- gigue $= 0 \Leftrightarrow$ délai d'acheminement constant

Bande passante

- Débit max. de bout en bout entre A et B.

Exigences de QoS

Critères de QoS pour quelques applications :

Application	Fiabilité	Délai	Gigue	BP
Mail	Haute	Faible	Faible	Faible
FTP	Haute	Faible	Faible	Haute
Web	Haute	Moyenne	Faible	Moyenne
VoIP	Faible	Haute	Haute	Faible
Conf. vidéo	Faible	Haute	Haute	Haute

- Mail, FTP et web : applications de transfert de fichiers. faibles exigences sur les critères temporels (délai et gigue). forte exigence sur la fiabilité
- VoIP et Conf. vidéo : applications **temps-réel**. fiabilité secondaire. fortes exigences sur les critères temporels

Le SLA (Service Level Agreement)

- Contrat passé entre
  - un fournisseur de service (p.ex., un fournisseur d'accès à Internet)
  - et un client (p.ex., une entreprise)

  et définissant
  - le type de service fourni
  - et les exigences de QoS attendues.
- Les exigences de QoS peuvent être fournies sous forme de **métriques**. Par exemple :
  - Bande passante garantie de 100 Mb/s.
  - Taux de perte de paquets < 0.001.
  - Délai max de bout en bout $< 100ms$.

  ...

Les métriques IPPM

- IPPM = IP Performance Measurement
- Groupe de l'IETF chargé de la définition
  - de métriques réseau pour les réseaux IP ;
  - et de méthodes de calcul de ces métriques.
- Quelques métriques :
  - RFC 2679 : A One-way Delay Metric for IPPM
    mesure du délai d'acheminement
  - RFC 2680 : A One-way Packet Loss Metric for IPPM
    mesure du taux de perte (dans un seul sens)
  - RFC 2681 : A Round-trip Delay Metric for IPPM
    mesure du délai aller-retour

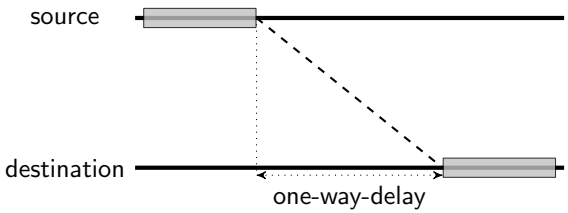
- La plupart des métriques IPPM sont dépendantes du type de paquet.
  - protocole de transport
  - ports source et destination
  - taille du paquet
- Quand une métrique est spécifique à un type de paquet, son nom est préfixé par `typePaquet`.

  $\Rightarrow$ Le nom de la métrique est alors `typePaquet-metrique`.
- Lors de la réalisation d'une mesure on doit préciser le type de paquet utilisé.

  Par exemple : `typePaquet` $\rightarrow$ `TCP-src1550-dst80-size800`
  (paquet TCP de 800 octets envoyé du port 1550 au port 80)

One-Way-Delay = délai dans un seul sens

- Le One-Way-Delay donne une indication plus fine que le Round-Trip-Delay (délai aller-retour).
  - Possibilité de routage asymètrique.
  - Pour de nombreuses applications (p.ex., streaming multimedia), on s'intéresse au délai dans un seul sens.
- Mesure le temps passé par le paquet sur le réseau :



- Nécessite une synchronisation des horloges (via GPS, par exemple).