Outil de cartographie et d'inventaire

Eric Chevigny
ENSCP
11 rue Pierre et Marie Curie 75005 Paris
eric-chevigny@enscp.fr

Kemal Ozcan
ENSCP
11 rue Pierre et Marie Curie 75005 Paris
kemal-ozcan@enscp.fr

Bernard Bellamy
ENSCP
11 rue Pierre et Marie Curie 75005 Paris
bellamy-bernard@enscp.fr

Résumé

La topologie et la configuration des éléments de réseau situés à la périphérie des backbones évoluent trop rapidement pour être connues de façon précise par l'administrateur réseaux. Ceci est d'autant plus vrai avec le développement de l'informatique mobile (connexion d'un ordinateur portable sur le réseau académique) et l'apparition de zones Wifi. Néanmoins, l'administrateur se doit de maîtriser la configuration et la topologie du réseau pour des raisons de sûreté de fonctionnement, de sécurité et de disponibilité (mauvaise configuration, connections non permises, surcharge du réseau due à un sous dimensionnement).

Afin d'apporter une solution à cette constatation, nous avons réalisé un outil logiciel de supervision à destination des administrateurs de réseaux Ethernet (IP over Ethernet) commutés. L'outil développé en Perl, java et PHP, permet de réaliser l'inventaire automatique et de définir la topologie logique (niveau 3 OSI) et physique (niveau 2 OSI) d'un réseau local (LAN) comportant des équipements hétérogènes dans un temps légèrement différé. Notre solution de supervision repose sur l'utilisation des protocoles SNMP, ARP, DNS, TCP et les informations définies dans les MIB II et MIB Bridges. L'outil ne nécessite pas de configuration préalable ni de plages d'adresses à parcourir. Le résultat peut être consulté à partir de n'importe quel poste client comprenant un navigateur web.

Mots clefs

supervision, topologie, cartographie, inventaire

1 Introduction

L'administrateur réseau, situé dans une petite structure est souvent seul pour gérer un parc de 500 à 600 machines.

Avec l'avènement des portables à un euro, des connexions wireless, le fait que les laboratoires appartiennent à plusieurs organismes de tutelles, il en découle que l'administrateur a bien du mal à faire remonter les informations vers lui pour gérer son parc. Des solutions existent mais elles sont souvent très coûteuses. (Hp open view, Tivoli, Lansurveyor, Landesk, etc..) Dans le monde du logiciel libre des développement de gestion de parc se développent actuellement, mais soit on doit rentrer les informations a la main (phpmyrezo, phpmycampus [1], GLPI [2]), soit ils nécessitent l'installation d'un client sur le poste pour l'analyser (OSCinventory [3]).

Là encore le monde la recherche n'est pas prêt. La règle est de ne pas toucher aux postes, chacun voulant être administrateur de sa machine.

Devant cette constatation et la situation historique du réseau à l'Ecole Nationale Supérieure de Chimie, nous avons été conduit à développer un outil d'inventaire et de cartographie. Inventaire on comprend facilement pourquoi. Mais pourquoi la cartographie ? La réponse est très simple les laboratoires ont une tendance non négligeable à disposer du réseau ethernet mis à leur disposition comme étant leur bien propre et à installer tout et n'importe quoi. Avec cet outil nous visualisons rapidement cet état de fait.

L'outil logiciel développé est à destination des administrateurs de réseaux Ethernet (IP over Ethernet) commutés, il permet de réaliser l'inventaire automatique du réseau et de définir la topologie logique (niveau 3 OSI) et physique (niveau 2 OSI) d'un réseau local (LAN) dans un temps légèrement différé sans nécessiter de configuration préalable ni de plages d'adresses à parcourir. Le résultat peut être consulté à partir de n'importe quel poste client comprenant un navigateur web.

2 Fondations

Notre solution implémentée en Perl, Java et PHP est basée principalement sur l'utilisation des protocoles SNMP [4], ARP [5], DNS [6], TCP [7] et les informations définies dans les MIB II [8] et MIB Bridges [9].

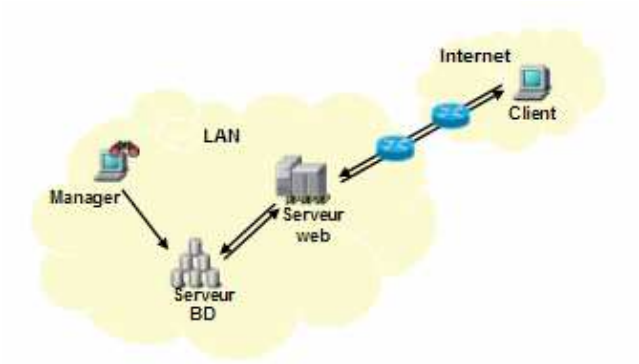

Figure 1 – *Architecture du système*

Le protocole **SNMP** [4] est le standard Internet de fait pour la gestion de réseau, définit par L'Internet Engineering Task Force (IETF). SNMP [4] est un protocole applicatif (niveau 7 sur le modèle OSI) de type question/réponse asynchrone. Il permet à un manager d'interroger les agents SNMP [4] présents sur les équipements constituant le réseau à partir des définitions des objets contenues dans les MIB. Aujourd'hui la majorité des équipements constituant la **backbone** des réseaux locaux sont compatibles SNMP [4]. Nous utilisons le protocole SNMP [4] pour récupérer les informations définies dans les MIB II [8] et MIB Bridges [9] implémentées par tous les équipements de réseaux compatibles SNMP [4]. Nous nous intéressons particulièrement aux informations contenues dans les tables : **IfTable**, **IpNetToMediaTable**, **IpRouteTable** définies dans la MIB II [8] et **Dot1dTpFdbTable** définies dans la MIB Bridges [9].

- L'objet **IfTable** contient des informations sur les interfaces du nœud. La table comprend notamment l'adresse physique de chacune des interfaces pour les nœuds possédant une adresse distincte sur chaque interface, ainsi que le statut opérationnel de celle-ci.
- L'objet **IpNetToMediaTable** contient une table connue sous le nom de cache **arp**, comme le protocole du même nom. La table **arp** est présente uniquement sur les ponts implémentant la couche de niveau 3 OSI ; principalement les routeurs. La table **arp** contient la correspondance adresse IP (logique) / adresse MAC (physique) et permet de faire le lien entre un inter-réseau et un réseau local. La table **arp** est un cache dynamique permettant de restreindre le nombre de requêtes **ARP** sur un réseau local.
- L'objet **IpRouteTable** contient la table de routage contenue dans les ponts de niveau 3 OSI (principalement les routeurs). Le champ **if** correspond au numéro de l'interface, **Route** correspond à l'adresse réseau du sous-réseau associée à l'interface, le champ **next Hop** indique l'adresse des nœuds directement ou indirectement connectés au niveau réseau (niveau 3 sur le modèle OSI) du pont interrogé. Le **type** indique si la connexion est directe ou indirecte et le champ **mask** correspond au masque réseau (**netmask**) de chacune des routes associées.
- L'objet **Dot1dTpFdbTable** contient la table de transmission du pont. Cette table connue sous l'acronyme **AFT** pour Adress Forwarding Table contient les correspondances adresse MAC / numéro de port tel que appris par le pont interrogé. Elle est construite dynamiquement par le pont et remise à jour périodiquement. La AFT permet d'aiguiller les trames au sein d'un réseau. Elle constitue la principale source d'informations pour réaliser l'algorithme de cartographie du réseau que nous avons implémenté.

Le protocole **ARP** [5] fait partie de la pile de protocoles TCP / IP et a pour vocation de réaliser la translation d'adresses entre un inter-réseau (WAN) et un réseau local (LAN). Il est situé entre les couches liaison (niveau 2 sur le modèle OSI) et réseau (niveau 3 sur le modèle OSI). Une utilisation détournée de ce protocole permet de réaliser l'inventaire des adresses physiques d'un LAN. Un LAN comporte un nombre limité de plages d'adresses IP contrairement aux adresses physiques (MAC) qui ne sont pas hiérarchisées. Si l'on connaît ces plages d'adresses, on peut réaliser l'inventaire des adresses IP valides et des adresses physiques correspondantes du LAN en balayant l'ensemble des plages à l'aide de requêtes ARP. L'utilisation du protocole ARP [5] nous permet de récupérer à l'aide d'une seule requête le couple adresse IP / adresse MAC.

Le protocole **DNS** [6] est un protocole applicatif. Il fait partie de la pile de protocoles TCP / IP et a pour vocation de réaliser la résolution de nom en adresses IP. La résolution se fait par interrogation d'un serveur DNS comprenant une base de noms. Une résolution inverse, nous permet d'obtenir le nom DNS à partir de l'adresse IP.

TCP [7] est un protocole de transport (niveau 4 sur le modèle OSI) avec connexion. Nous utilisons les caractéristiques de la négociation de connexion en trois étapes (free hands shake) du protocole TCP [7] dans le but de déterminer l'état des ports d'un hôte distant. L'analyse des résultats nous permet de déterminer le type d'hôte.

L'implémentation de l'algorithme a été réalisé pour un environnement Windows XP ; cependant les choix d'utiliser des langages portables en se basant sur les standards de l'Internet fait que l'application peut être facilement portée, moyennant quelques modifications, sous d'autres environnements. Notre souhait était de réaliser une application autonome qui ne nécessite pas une

connaissance « a priori » du réseau à inventorier, si ce n'est le type à savoir un réseau Ethernet commuté.

La section 3 présente notre algorithme pour réaliser l'inventaire et définir la topologie du réseau. L'implémentation de l'algorithme et l'architecture du système sont présentées dans la section 4.

3 Présentation de l'algorithme

Nous avons implémenté un algorithme à partir des modèles proposés dans les travaux réalisés par Y. Breitbart et *al* [10] et B. lowekamp et *al* [11] pour réaliser l'inventaire des équipements actifs du réseau et en définir la topologie. L'étape d'inventaire consiste à recenser l'ensemble des équipements (stations de travail, serveurs, imprimantes réseau, routeurs, switchs) présents sur le réseau. Le résultat de cette étape se présente sous la forme d'une liste des équipements actifs sur le réseau. La découverte du réseau logique de niveau 3 (IP) se fait à partir des informations contenues dans la table de routage des routeurs ou des matériels comportant des fonctionnalités de routage. Le contenu des tables est récupéré à l'aide du protocole SNMP [4]. Le parcours exhaustif des plages d'adresses des sous réseaux composant le réseau logique à l'aide de requêtes ARP nous permet de récupérer le couple adresse IP / adresse MAC ; indispensable pour réaliser le lien entre les informations sur le réseau logique (IP) et physique (MAC). La dernière étape de l'inventaire consiste à récupérer des informations complémentaires sur les équipements à l'aide des protocoles DNS [6] et TCP [7].

L'étape de définition de la topologie a pour objectif de tracer une représentation graphique du réseau à partir des informations recueillies ; ceci nécessite au préalable, de trouver les liens physiques ou logiques existant entre les éléments du réseau. La découverte de la topologie du réseau physique est basée sur les données contenues dans les tables de transmission (AFT) de chacun des ponts administrables du réseau.

3.1 Inventaire du réseau

L'algorithme pour réaliser l'inventaire du réseau comporte sept étapes successives :

1. Récupérer la configuration réseau du manager
2. Rechercher les routeurs composant le domaine administré
3. Rechercher les sous réseau du domaine
4. Récupérer un premier ensemble de couples adresse IP / adresse MAC
5. Récupérer tous les couples adresse IP / adresse MAC des équipements actifs du domaine
6. Rechercher les noms des équipements
7. Identifier les équipements

Ne possédant aucunes informations préalables sur le réseau, sauf spécifications contraires par l'utilisateur ; la première étape de notre algorithme consiste à récupérer les informations de configuration de l'adaptateur de la station de gestion (manager) sélectionné pour découvrir le réseau. Si le système est convenablement configurée pour accéder au réseau, un certains nombre de paramètres sont obligatoirement présents tel que l'adresse **MAC** (adresse physique) de l'adaptateur, l'adresse **IP** et le masque de sous réseau (**subnet**) associé, l'adresse IP de la passerelle de proximité (**default Gateway**) et l'adresse IP du serveur de noms (**DNS**) pour le domaine administré. Suivant le système hôte, ces informations sont présentes soit dans un fichier de configuration (systèmes Unix et associés), soit dans une base de registre (systèmes Windows).

Dans un deuxième temps, l'algorithme recherche l'ensemble des routeurs, ou commutateurs offrant des fonctionnalités de routage, du domaine administré en commençant par la passerelle de proximité dont l'adresse est maintenant connue. Après avoir recherché le nom de communauté SNMP du routeur, notre algorithme recherche les adresses IP des routeurs voisins dans la table de routage à l'aide de requêtes SNMP. En absence de spécifications de la part de l'utilisateur l'algorithme utilise des paramètres par défaut (nom de communauté et port SNMP). On récupère les adresses IP valides contenues dans le champ **ipRouteNextHop** de l'objet **ipRouteTable** (la table de routage) du groupe ip de la MIB II [8] pour chacun des routeurs administrables découverts. Le champ **ipRouteType** doit également être positionné à 3 (route directe) ou 4 (route indirecte). La recherche se fait itérativement jusqu'à ce qu'aucun routeur présent dans la liste des routeurs ne soit administrable. Ceci constitue la condition d'arrêt de notre algorithme.

Dans l'hypothèse où le premier routeur de la liste ne serait pas administrable. Le logiciel tente de trouver un routeur coopérant après un parcours exhaustif, à l'aide du protocole **ARP**, de la plage d'adresses constituant le sous réseau auquel appartient la station de gestion (manager). Pour identifier un routeur administrable, l'algorithme teste la présence de la **MIB Bridges** [9] et les objets **sysServices** et **ipForwarding** de la MIB II [8]. La présence de la MIB Bridges [9] permet de différencier les routeurs et commutateurs administrables des autres équipements. L'objet **sysServices** donne des informations sur le niveau (OSI) de service offert par le noeud. L'objet **ipForwarding** indique si l'équipement transmet les

paquets qu'il reçoit (1) où s'il rejette les paquets qui ne lui ont pas adressés (2).

La troisième étape récupère tous les sous réseaux du domaine administré. Pour chacun des routeurs administrables obtenus précédemment, on récupère l'adresse des sous réseaux des interfaces actives**.** Les sous réseaux visibles depuis un routeur sont déterminés à l'aide des champs **ipRouteDest** et **ipRouteMask** de la table **ipRouteTable** du groupe ip de la MIB II [8]. Le champ **ipRouteType** de cette même table peut être utilisé pour déterminer si le sous-réseau est connecté directement ou non au routeur. La valeur de l'objet **ifOperStatus** de la table **ifTable** doit être contrôlée au préalable ; si l'interface est active celle-ci est positionnée à 1 (up).

La quatrième étape récupère les adresses MAC et IP associées contenues dans le cache **arp** de chacun des routeurs. On récupère, à l'aide de requêtes SNMP, pour chacun des routeurs découverts précédemment la table **ipNetToMediaTable** contenue dans le groupe ip de la MIB II [8]. Cette table nous donne la correspondance entre les adresses physiques (adresses MAC) et les adresses IP pour chacun des segments de réseau issus des interfaces du routeur.

La cinquième étape collecte les adresses de tous les éléments actifs en parcourant la plage d'adresses de chacun des sous réseaux découverts précédemment à l'aide de requêtes **ARP**. On réalise un parcours exhaustif dans le but d'obtenir l'ensemble des couples adresse IP / adresse MAC correspondant aux équipements actifs du réseau. Cette étape vient compléter les informations collectées à l'étape précédente.

Après avoir récupéré les adresses MAC et IP de l'ensemble des machines constituant le réseau administré, la sixième étape récupère les noms DNS et les noms et domaines netBIOS si ils existent. Pour chacune des adresses IP contenues dans liste de nœuds on réalise une résolution inverse de nom auprès du serveur DNS du domaine dont l'adresse IP a été découverte au cours de la première étape et une requête de nom netBIOS auprès du système distant. A l'issue de cette étape on possède l'ensemble des adresses IP et MAC ainsi qu'un nom DNS et netBIOS, s'ils existent, des nœuds actifs sur le réseau.

Enfin la dernière étape de notre algorithme identifie le type d'équipement correspondant aux adresses des éléments actifs découverts précédemment. L'objectif ici, est de différencier les nœuds du réseau (routeurs, switchs, hubs) entre eux et de les séparer des hôtes (stations terminales, imprimantes réseaux, serveurs). Connaissant l'adresse IP d'un nœud nous réalisons une série de tests sur les services offerts pour chacun des équipements. Les noeuds du réseau sont identifiés à l'aide de requêtes SNMP tel que décrit ci-dessus à l'étape numéro deux. La méthode pour identifier les hôtes repose sur l'émission de segments TCP SYN sur des ports particuliers. Cette méthode nous permet de différencier les stations de travail appartenant à la famille des systèmes Windows, des stations appartenant à la famille des systèmes Unix et dérivés ainsi que les imprimantes réseau.

3.2 Cartographie du réseau

L'algorithme de définition de la topologie utilise les résultats obtenus précédemment au cours de l'inventaire pour réaliser une représentation graphique du réseau. Il réalise un ''instantané'' (snapshot) de la topologie du réseau tel qu'il est configuré dans les ponts le constituant au moment ou les tables de transmission (AFT) sont récupérées.

Les équipements de niveau 2 (OSI) sont totalement transparents du point de vue des équipements de niveau 3 et des équipements terminaux. Par conséquent la découverte des liens physiques entre les équipements participant au réseau ne peut se faire sans la participation de commutateurs (switchs) administrables. Le rôle d'un commutateur est de diriger les flux de données sur un réseau de niveau 2. Pour éviter l'engorgement du réseau, le commutateur filtre les trames entre les segments de réseau auxquels il est connecté. Pour ce faire, il écoute toute l'activité sur chaque segment de réseau auquel il est connecté et stocke en mémoire les adresses physiques (MAC) associées au numéro d'interface des trames qu'il reçoit, constituant les tables de transmission (AFT). Certains commutateurs participent avec leurs voisins à l'algorithme du ***Spanning Tree*** dont l'objectif est de créer un arbre de recouvrement total sur l'ensemble du sous réseau en éliminant les interconnexions redondantes à l'origine de boucles infinies. Ainsi le réseau s'adapte dynamiquement, sans que l'intervention d'un administrateur ne soit nécessaire, aux modifications de la configuration du réseau. Si un lien tombe, une nouvelle route sera configurée. Récemment, Son Myung-Hee et *al* [12] ont proposé un algorithme de découverte de la topologie physique d'un réseau à partir des informations issues du protocole de Spanning Tree.

Notre algorithme repose sur l'utilisation des tables de transmission maintenues dynamiquement par les commutateurs administrables et les caractéristiques du graphe obtenu à l'aide de l'algorithme du ***Spanning Tree*** pour définir la topologie. Il commence par rechercher les connexions entre les ponts administrables composant le réseau. Les nœuds non administrables sont associés à des hôtes. Pour se faire on commence par récupérer les données issues de l'inventaire, les tables de transmission (AFT) et les tables des interfaces de tous les ponts

administrables à l'aide de requêtes SNMP. Les tables de transmission sont contenues dans l'objet **dot1dTpFdbTable** du groupe **dot1dTp** de la MIB Bridges [9] pour les commutateurs. Les tables des interfaces sont contenues dans l'objet **IfTable** de la MIB II [8]. Notre algorithme tente d'associer un équipement à chacun des ponts administrables constituant le réseau ; puis recherche les incohérences possibles issues de l'association. Si tous les ponts « voient » l'équipement et le pont auquel il est associé temporairement sur le même port alors il n'y a pas d'incohérence. Dans le cas contraire l'équipement et le pont sélectionné ne peuvent être connectés. La procédure décrite ci-dessus est réalisée pour chacun des ponts contenus dans l'inventaire. Après avoir listé les ports des ponts connectés à d'autres ponts, sur lesquels aucun autre équipement ne peut être connecté, la recherche des connexions entre les pont administrables et les hôtes se fait suivant le même algorithme. Ceci permet d'éviter les erreurs d'association dues à la présence d'hôtes partiellement effacés des tables de transmissions des ponts du réseau et ne pouvant plus être associés par l'algorithme aux nœuds auxquels ils sont réellement connectés. Les tables de transmission étant remises à jour en moyenne toutes les 30 minutes (age-time). Pour finir l'algorithme recherche la présence de hubs ou de switchs non administrables. Le port d'un pont administrable, non connecté à un autre pont, comportant plusieurs hôtes révèle la présence d'un hub ou d'une série de hub. Les hubs et switchs non administrable sont modélisés par un « hub virtuel ».

Le graphe obtenu est une représentation non pas des connexions physiques réelles, mais de l'arbre de recouvrement du réseau au moment de la découverte ; les connexions redondantes éliminées par l'algorithme du *Spanning Tree* n'apparaissent pas. L'algorithme implémenté gère les connexions multiples (**trunks**) et les boucles pouvant être présentes sur un réseau tel que celui de L'Ecole Nationale Supérieure de Chimie de Paris (ENSCP).

3.3 Limitations

Le modèle sur lequel nous nous sommes basé pour implémenter notre algorithme comporte quelques limitations. Notre solution basée sur le protocole SNMP [4] nécessite la collaboration au minimum d'un routeur ou un équipement équivalent ; ainsi qu'un maximum de switchs administrables. Les éléments non administrables tel que les hubs ou les switchs non administrables sont devinés par notre algorithme. Cependant nous ne sommes pas capable de différencier un ensemble de machines connectées à une série de hubs ou de switchs non administrables ou le même ensemble connecté sur un seul hub. Les segments comportant plusieurs machines connectées sur un même port d'un switch administrable sont symbolisés par un « hub virtuel », celui-ci pouvant recouvrir plusieurs configurations réelles. De même le modèle que nous utilisons pour représenter le réseau associe une machine à une interface physique. Il se peut par conséquent qu'une machine possédant plusieurs adaptateurs réseau apparaisse sur le schéma comme des hôtes distincts.

4 Implémentation

Les administrateurs du réseau pouvant être amenés à consulter les informations depuis des sites différents, comme par exemple au cours d'une intervention de dépannage sur un poste utilisateur, le choix de l'architecture a été de développer dans un système multi tiers composé de trois entités : un manager, un serveur et un client (Figure 1).

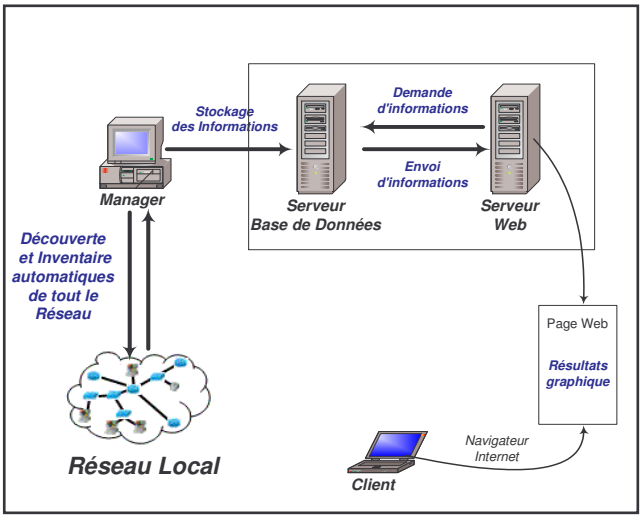

Figure 2 – *Description du système*

Le manager implémenté en Perl réalise l'inventaire et définit la topologie du réseau suivant l'algorithme décrit précédemment. La partie manager est composée d'un interpréteur PERL et d'un ensemble de modules orientés objet. Les résultats sont stockés dans une base de données sur un serveur de données Mysql.

La base de données conserve les informations issues de l'inventaire pour chacune des interfaces des équipements découverts : adresse MAC, adresse IP, nom DNS, type de matériel... Ainsi que les informations sur le graphe (nœuds, arcs) généré par le module de cartographie et affiché par le client.

La partie serveur composée d'un serveur de données Mysql, d'un serveur de pages Web Apache (ou IIS), d'un interpréteur et de modules PHP, génère dynamiquement les

pages destinées au client à partir des informations recueillies auprès du serveur de données.

Le client est constitué de pages HTML générées dynamiquement à partir de scripts PHP côté serveur. La page principale comporte un Applet java chargé de représenter le réseau administré découvert par le manager.

La représentation se fait sous la forme d'un graphe. Les nœuds du graphe représentent les nœuds (routeurs, swichs, hubs) ou les hôtes (station de travail, imprimantes …) constituant le réseau. Les connexions entre les nœuds sont modélisées par un arc. Le type de nœud (routeur, switch, station de travail Unix ou Windows, serveur, imprimante …) représenté par une icône est aisément reconnaissable. Enfin chaque nœud est désigné par un alias qui peut être le nom DNS s'il existe, le nom netBIOS, l'adresse IP, voire l'adresse MAC. Les échanges entre la partie serveur et le client se font via le protocole HTTP.

Il est également possible de consulter le résultat de l'inventaire des équipements du réseau sous la forme d'un listing.

L'utilisation de code mobile (Applet java) permet à un utilisateur de consulter les informations à l'aide d'un navigateur à partir de n'importe quel poste possédant une connexion Internet sans nécessité d'installation d'un logiciel.

Le paramétrage de l'application se fait au travers d'un fichier de configuration en mode texte (inventaire.conf).

```
# Fichier de configuration pour l'inventaire dynamique
# d'un réseau de type Ethernet
# Routeur par défaut :
DefaultGateway = 10.0.0.1
# nom de communauté (séparés par un point virgule) - par défaut: public
community = public;enscp
# port SNMP (requêtes) - par défaut: 161
port = 161
# Paramètres de connexion à la base de données :
# nom DNS ou adresse IP du serveur de base de données - par défaut: localhost
dbServer = 127.0.0.1
# Port du serveur:
dbPort = 3306
# type de base de données :
dbType = mysql
# login :
dbUser = "votre login"
# Pass :
dbPass = "votre mot de passe"
```

Figure 3 – exemple de fichier de configuration

Tous les paramètres sont facultatifs ; l'application recherche d'elle-même les informations dont elle a besoin pour réaliser l'inventaire sur la machine hôte ou utilise des paramètres par défaut (nom de communauté et port SNMP).

S'il le désir, l'utilisateur peut néanmoins spécifier les plages d'adresses IP à parcourir. Cette fonctionnalité additionnelle est intéressante dans le cas d'un réseau fortement sécurisé dans lequel aucun routeur ne collabore : ce qui constitue une condition de réussite de la découverte automatique du réseau par notre algorithme.

4.1 Distribution

Notre projet logiciel à destination des administrateurs de réseaux est ouvert. Le développement se poursuit et l'application s'enrichit régulièrement de nouvelles fonctionnalités.

Après avoir posé la première pierre d'un système de supervision, notre projet devrait s'orienter vers la gestion de parc. Nous sommes en effet partis de la constatation que les administrateurs connaissaient bien les matériels réseau et systèmes composant l'infrastructure du système d'information. En contre partie la nature et la configuration des équipements connectés à la périphérie des backbones sont très souvent méconnues. Or ceci est indispensable pour un service réseau dont l'objectif principal est de fournir un service fiable aux utilisateurs.

Suivant l'intérêt porté par la communauté à notre projet présenté aux JRES 2005, nous envisageons de diffuser librement notre application.

4.2 pré requis pour l'installation

Vous devez pour faire fonctionner cet outil, disposer des éléments suivant sur votre serveur

Un serveur apache ou IIS

Une base de donnée Mysql

Les langages Perl ,
PHP ,
et Java

Et enfin de la librairie WinPcap.

5 Résultats

Notre application a été testée sur le réseau hétérogène de L'Ecole Nationale Supérieure de Chimie de Paris (ENSCP). Le réseau de l'ENSCP est un réseau de type **Ethernet commuté** ; il comprend environ 500 machines hôtes, composé à 90% de machines Windows (de Windows 95 à Windows Server 2003) le reste étant essentiellement des environnements de type UNIX et Mac

OS, et d'une trentaine de nœuds (router, switchs, hubs) constituant la **backbone** du réseau. Il est divisé en trois zones principales : une **DMZ**, une zone **NAPT** et une zone WiFi

- La **DMZ** du réseau de l'ENSCP est constituée d'une **classe C** et la zone **NAPT d'une classe A** ;

- La zone **NAPT** est elle-même subdivisée en une dizaine de **subnets.** Une machine présente dans un subnet ne peut « voir » que les machines présentes dans le même subnet. L'échange entre subnets nécessite l'intervention d'un routeur ; même si les machines sont situées dans le même segment physique de réseau.

La séparation de la **DMZ** et de la zone **NAT**, ainsi que la subdivision de la zone **NAT** en subnets sont réalisées par un switch – routeur **Foundry** (switch comportant des fonctionnalités de niveau 3 OSI). La zone **NAPT** est inaccessible depuis l'extérieur du fait qu'elle utilise des adresses IP non routables.

Pour réaliser les tests nous avons placé le manager dans la zone NAPT. Les tests de cartographie réalisés montrent que l'application découvre automatiquement l'ensemble des machines actives et définit la topologie correcte de la couche 2 et 3 OSI du réseau. De plus, environ quatre vingt dix pour cent des hôtes sont correctement identifiés.

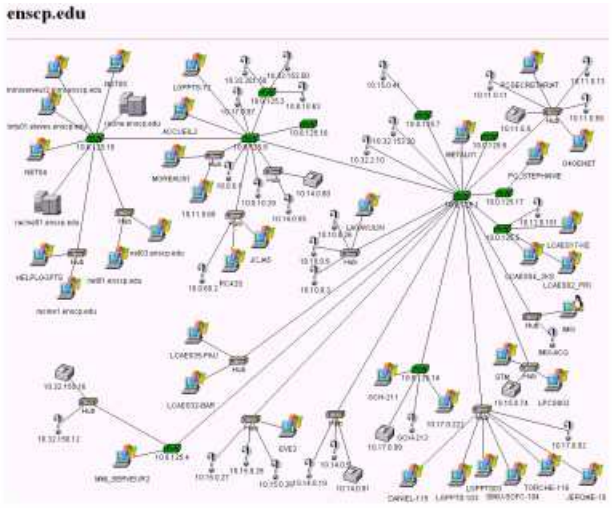

Figure 3 – *Exemple de cartographie*

L'application a également été testée avec succès sur un réseau d'entreprise beaucoup plus modeste la société TR2E. Aucune information n'a été donnée a l'application les seuls informations du serveur DHCP ont suffit à trouver l'ensemble du réseau

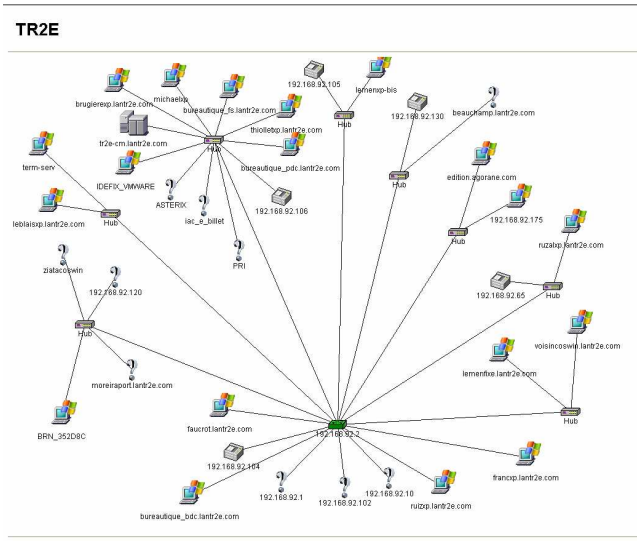

Figure 3 – Exe*mple de cartographie*
(Avec l'aimable autorisation de la société TR2E)

Nous avons testé notre solution sur des réseaux locaux Ethernet comportant des équipements réseau de différentes marques tel que Cisco, Hp, 3Com, D-Link et Netgear.

Malgré la mention "SNMP compliant" affichée, certains matériels destinés à une diffusion grand public ne comportent pas toutes les informations requises par notre algorithme.

6 Conclusion

L'application développée est un outil essentiel à la maintenance et la planification d'un réseau. Basée sur l'utilisation des protocoles SNMP [4], ARP [5], DNS [6], TCP [7] et les informations définies dans les MIB II [8] et MIB Bridges [9] ; elle permet de réaliser l'inventaire et la cartographie d'un réseau LAN hétérogène. L'inventaire régulier du réseau administré permet de connaître à tout moment la configuration réelle de celui-ci, d'identifier des connexions non autorisées ou de détecter les pannes. La connaissance de la topologie physique et logique permet à l'administrateur de détecter les points névralgiques du réseau, les modifications non testées, prohibées ou non documentées qui peuvent mettre à mal le réseau et de planifier les évolutions futures.

Références

L'implémentation de notre algorithme d'inventaire et de cartographie de réseau LAN réalisé en Perl importe un certain nombre de modules librement diffusés conformément aux dispositions de la licence Perl ainsi que la librairie WinPcap développée par l'Université Polytechnique de Turin. Le module Net::SNMP réalisé par

David M. Town est une implémentation en Perl du protocole SNMP. Le module Net::DNS réalisé par Michael Fuhr et Chris Reinhardt est un resolver DNS. Le module Net::NBName réalisé par James Macfarlane est un client de noms NetBIOS. Le module Net::Ping réalisé par Andreas Karrer, Paul Marquess, Russell Mosemann, Rob Brown, Colin McMillen et Scott Bronson, permet de tester l'état (actif / inactif) d'un hôte distant sur un réseau à l'aide des protocoles ICMP, TCP et UDP. Le module DBI écrit par Tim Bunce est une interface d'accès aux bases de données. Enfin le module Win32::Netpacket réalisé par J-L Morel est une interface orientée objet vers l'API driver packet.dll. l'API driver packet.dll fait partie du package WinPcap. La librairie WinPcap permet la capture de paquets sous Windows. L'API driver packet.dll offre un ensemble de fonctions pour analyser le trafic réseau, envoyer des paquets sur le réseau, obtenir une liste des adaptateurs réseau et obtenir différentes informations sur un adaptateur.

Bibliographie

[1] http://phpmycampus.cnrs-gif.fr/

[2] http://glpi.indepnet.org/

[3] http://ocsinventory.sourceforge.net/

[4] J. Case, M. Fedor, M. Schoffstall et J. Davin, *A Simple Network Management Protocol (SNMP)* Internet-RFC 1157, Mai 1990.

[5] David C. Plummer *An Ethernet Address Resolution Protocol*, Internet RFC-826, novembre 1982.

[6] P. Mockapetris, *Domain Names* Internet RFC-1034 & 1035, Novembre 1987

[7] Transmission Control Protocol, RFC-793, septembre 1981.

[8] K. McCloghrie et M. Rose, Management Information Base for Network Management of TCP/IP-based internet : MIB II. Internet RFC-1213, Mars 1991.

[9] E. Decker, P. Langille, A Rijsinghani et K. Mc Cloghrie, *Definitions of Managed Objects for Bridges*. Internet RFC – 1493, Juillet 1993.

[10] Y. Breitbart, M. Garofalakis, B. Jai, C. Martin, R. Rastogi et A. Silberschatz, *Topology Discovery in Heterogeneous IP Networks: The NetInventory System*. IEEE INFOCOM, 2000.

[11] B. Lowekamp, D.R. O'Hallaron, et T.R. Gross, *Topology Discovery for Large Ethernet Networks*. ACM / SIGCOMM, 2001.

[12] S. Myung-Hee, J. Bheom-Soon, K. Byung-Chu et L. Jae-Yong, *Physical Topology Discovery for Metro Ethernet Networks.* ETRI Journal, Volume 27, numéro 4, Août 2005.