LES RESEAUX TCP/IP

ARCHITECTURE, SERVICES, IMPLEMENTATION

ADMINISTRATION, SECURITE

Corrigé des exercices

Jacques PHILIPP

TABLE DES MATIERES

CHAPITRE 1

ARCHITECTURE ET NORMALISATION DES SYSTEMES DISTRIBUES

Les réponses aux nombreuses questions de cours sont dans l'ouvrage et ne sont donc pas traitées ici. Idem pour les commandes et services qu'il suffit de tester.

Ne sont donc rappelées que les questions dont on apporte les réponses.

1. EXERCICES

1°) Différences entre norme et standard.

Une norme est issue d'un organisme de normalisation, suite à un vote, après consultation des acteurs concernés (administrations gouvernementales, industriels, consommateurs).

Un standard est propriétaire et s'impose (de lui-même ?) à l'ensemble des acteurs.

2°) Définir les rôles du modèle de référence, des couches et protocoles.

Le modèle de référence permet de mettre en évidence les principes fondamentaux de conception d'applications basées sur l'architecture client serveur.

Il définit très précisément les fonctionnalités à respecter et à implémenter dans les différentes couches ainsi que les notions fondamentales de service (offert par une couche du modèle) et de protocole de dialogue entre des entités homologues du modèle

3°) Comme un modem gère-t-il les erreurs de transmission ?

Par code cyclique et bit de parité.

4°) Quelles sont les conséquences du théorème de Shannon.

Il indique une borne supérieure du débit en fonction des caractéristiques physiques du support et du contexte d'exécution de la transmission.

5°) Quelle est l'atténuation d'un fils téléphonique traditionnel.

Régénération nécessaire tous les 1830 m.

6°) Définir un support sériel et un support parallèle.

Les informations transmises peuvent l'être en série (principe de la voie de chemin de fer unique sur laquelle circulent tous les trains les uns après les autres) ou en parallèle (plusieurs voies sur lesquelles peuvent circuler, en parallèle, plusieurs trains).

7°) Topologies de l'architecture ethernet et protocole associé.

Architecture en bus avec le protocole non déterministe CSMA/CD.

8°) Rappeler les principes de l'architecture client-serveur.

Modèle de communication asymétrique dans lequel n clients peuvent faire des requêtes à p serveurs, base de tous les systèmes de communication.

CHAPITRE 2

RESEAUX LOCAUX

1. EXERCICES

1.1 Notions élémentaires

1°) Rappeler la définition d'un LAN, d'un MAN, d'un WAN.

Un réseau longue distance (WAN) peut être constitué de plusieurs réseaux métropolitains (MAN) eux même constitués de plusieurs réseaux locaux (LAN).

2°) Définir les contraintes de fonctionnement des réseaux locaux.

Faible coût, disponibilité totale, fiabilité, sécurité d'accès, de fonctionnement, des données, transparence, simplicité d'administration, efficacité, ergonomie, etc.

3°) La topologie physique d'un réseau ethernet correspond-t-elle à son architecture logique ?

Non en général car physiquement, une topologie en étoile masque une architecture logique en bus.

4°) Rappeler les règles de sécurité à respecter dans un environnement d'un réseau.

Sécurité d'accès aux données (mots de passe, clé publiques et privées, cryptage), tolérance de panne, sécurité de fonctionnement, sauvegarde des serveurs de données, surveillance des débits sur les lignes, recherche et prévention des goulets d'étranglement

5°) Différences entre un répéteur, un pont, un switch, et un routeur.

Le niveau dans le modèle : répéteur (couche physique), pont et switch (couche liaison), routeur (couche réseau).

6°) Définition d'une route.

C'est le chemin suivi par l'unité de donnée de protocole (par exemple un datagramme) pour arriver à destination.

1.2 Exercice récapitulatif

Le problème des Sioux et des signaux de fumée (E. Dijkstra -1974), complété par l'auteur, illustre, de façon amusante, quelques problèmes posés par le protocole du jeton et l'administration de réseau.

Imaginons que nous soyons au Far West. Une caravane composée de visages pâles, rangée circulairement est attaquée par des indiens Sioux. Comme chacun le sait, les combats cessent pendant la nuit. Toutefois, les sioux veulent être surs de retrouver leur proie le lendemain matin. Leur culte leur imposant d'invoquer périodiquement le Grand Manitou durant la nuit, il leur faut empêcher les visages pâles de s'enfuir pendant la prière.

Comme ce sont d'excellents logiciens, ils décident pendant leur guet d'invoquer le Grand Manitou selon le *rituel suivant* :

- ils constituent un *anneau virtuel* circulaire, leurs *yeux perçants* leur permettant de communiquer entre eux. Soit l'ensemble $S = \{S_1, S_2,..., S_n\}$ de n guetteurs en attente, ordonné de telle sorte que chaque guetteur S_k voit *uniquement son voisin de gauche* ou *prédécesseur* S_{k-1}. Cette chaîne est *bouclée* car S_1 a pour voisin de gauche S_n.

- à un instant donné, *un sioux au plus* doit prier, et sa prière dure un temps *fini* quelconque, éventuellement nul.

Pour mettre en oeuvre ce rituel, ils adoptent le *protocole* suivant :

- Chaque sioux S_k de l'ensemble $\{S_2,...,S_n\}$ peut commencer à invoquer le grand Manitou si son voisin S_{k-1} a son feu allumé alors que lui-même a son feu éteint ou l'inverse. Il allume ou éteint son feu (selon que le feu est déjà éteint ou allumé) dès qu'il cesse sa prière.
- Le sioux S_1 ne peut recommencer à prier que si les feux respectifs de S_n et le

sien sont simultanément soit allumés soit éteints. Pour cesser de prier, il allume ou éteint son feu selon qu'il est éteint ou allumé.

- Au départ, tous les feux sont éteints.
- S_1 initialise le processus à sa convenance. Il peut prier dès qu'il le souhaite et allume un feu dès qu'il terminé ses évocations. Le sioux S_2 *peut alors constater* qu'il détient *le droit de prier*, invoquer le Grand Manitou à sa convenance, allumer un feu dès qu'il a terminé, cédant ainsi le droit de prier au guetteur suivant. Le procédé est itératif jusqu'à ce que les n feux soient allumés. Le processus se poursuit par une suite d'extinctions de feux jusqu'à ce qu'ils soient tous éteints et ainsi de suite. Imaginez la terreur des visages pâles voyant ce spectacle insolite...

1°) Le rituel précédemment décrit est-il respecté par ce protocole ? Justifier la réponse.

Un simple raisonnement par récurrence sur le nombre de guetteurs prouve la pertinence du protocole.

2°) Un sioux est-il obligé d'observer en permanence son voisin pour pouvoir prier ? (Justifiez la réponse). Quelles sont les conséquences de cette observation ?

Non. S'il l'était, un temps considérable serait perdu à ne faire que de l'observation (ou de l'attente) active.

3°) Déterminer les instants les plus adaptés durant lesquels les visages pâles peuvent s'enfuir ?

Pendant la prière bien sûr.

4°) Que se passe-t-il si un des sioux tarde à prier ? Peut on améliorer l'algorithme ? Le temps de prière doit il être borné ?

Perte de temps et blocage du protocole. L'amélioration de l'algorithme est suggérée dans la question : le temps de prière doit être borné.

5°) Que se passe-t-il si un des guetteurs est tué ? Proposer une modification de l'algorithme pour y remédier selon deux stratégies :

◊ le chef décide d'une solution à partir du moment où il réalise que le droit de prier ne circule plus (comment peut il le réaliser ?),

◊ les guetteurs mettent automatiquement en place une solution (à définir) à partir du moment où l'un d'entre eux réalise que le droit de prier ne circule plus.

La première stratégie permet de reconstruire un ou deux anneaux, à partir du moment où le chef constate la panne de fonctionnement, le temps maximum de prière, borné, de l'ensemble des guetteurs étant connu.

La deuxième stratégie est plus rapide puisque le guetteur qui suit le guetteur tué est le premier à identifier le problème et à y remédier.

6°) Que se passe-t-il si le chef est tué ? Proposer et discuter des avantages et des inconvénients d'au moins deux stratégies de remplacement du chef, l'une basée sur le remplacement automatique du chef par un guetteur (le premier, au hasard ?), l'autre basée sur l'élection d'un nouveau chef.

La question illustre le cas où il faut désigner un nouveau chef (un nouveau serveur), le chef étant hors service. Plusieurs stratégies sont possibles : au hasard (pas bon si le hasard fait mal les choses et qu'il choisi un serveur inadapté), déterminée : le chef défunt est remplacé par un nouveau chef, nommé (serveur de sécurité) ou élu. La question du réveil du chef endormi se pose puisqu'il ne doit pas y avoir deux chefs détenant l'autorité.

7°) Démontrer que le temps d'attente du droit de prier est borné. Conclusion ?

Déjà vu dans les questions précédentes.

8°) Que pensez-vous de l'efficacité de cet algorithme ? Justifier la réponse.

Algorithme relativement peu efficace puisque un seul jeton circule alors que la bande passante semble largement inutilisée.

9°) Que se passe si un sioux est mécréant et refuse de prier ?

Il bloque tout sauf s'il consent à respecter le protocole (allumer ou éteindre son feu dès réception du jeton, même sans prière).

10°) Le chef peut-il être aidé dans sa tâche par un sous chef ? Préciser son rôle et ses droits, si le chef est tué ou simplement blessé.

Le chef (serveur primaire) est doté d'un second (serveur de sécurité) qui devient naturellement le nouveau chef si nécessaire. La question du réveil du serveur primaire se pose puisqu'il ne doit pas y avoir deux chefs détenant l'autorité.

L'idée ici est de disposer d'un serveur de sécurité, doté d'une autorité suffisante pour se substituer temporairement ou définitivement au serveur primaire.

11°) Que se passe-t-il si deux guetteurs sont tués simultanément ? Indiquer deux remèdes différents.

Double rupture de l'anneau. Deux solutions : reconstituer un anneau sans les deux guetteurs ou deux anneaux réduits autonomes.

12°) Peut on imaginer une modification de l'algorithme qui permette à deux sioux de prier simultanément ?

Deux jetons au lieu d'un seul.

12°) Existe-t-il des applications de ce type d'algorithme ? Si oui, lesquelles ?

Le réseau Token Ring d'IBM (4Mo/s avec un seul jeton, 16Mo/s avec deux).

13°) Pour s'enfuir, les visages pâles sont prêt à tout, y compris à essayer de se *faire passer pour* un sioux.

a) Le présent algorithme garantit-il *l'authentification mutuelle* du couple de sioux (S_k,S_{k+1}). Justifier la réponse.

Non; rien n'a été prévu dans le protocole proposé. Un visage pâle peut donc tuer un sioux et se faire passer pour lui en prenant sa place (usurpation d'identité). Pendant son temps de prière (certes borné), un nombre certain de visages pâles peuvent s'enfuir. Il peut même gagner du temps en priant aussi longtemps que possible (temps borné tout de même).

b) Proposer un algorithme qui garantisse cette *mutuelle authentification*. On distinguera le cas particulier du chef.

Définir un code secret de transmission du jeton, connu des guetteurs seuls, et invisible pour les visages pâles (principe du serveur d'authentification), des clés secrètes ou publiques.

CHAPITRE 3

INTERNET ET LES SERVICES TCP/IP

1. EXERCICES

1.1 Services Darpa-BSD

1°) A partir de la commande *hostname* ou de la commande *uname*, identifiez le nom de domaine de votre station. Cet identificateur est-il arbitraire ?

Oui

2°) A partir du fichier *hosts*, identifiez l'adresse *IP* de votre poste de travail. Est-elle arbitraire ?

Non, elle doit obéir aux règles classiques d'attribution des adresses IP.

3°) Dans certains sites, le fichier *hosts* ne contient pas cette information. Essayer la commande *ypcat hosts*

Cas où un serveur NIS est installé.

D'autres serveurs peuvent être utilisés (DNS, NIS, NIS+, LDAP, etc.).

6°) A partir de la commande *ftp*, faire un transfert de fichiers entre deux stations dans deux situations différentes :

a) Vous n'avez pas de mot de passe sur le serveur de fichiers, (ne marche pas en principe).

b) Vous avez un mot de passe sur le serveur de fichiers (OK).

7°) Ecrire un script de transfert de fichiers entre deux stations utilisant la commande *ftp*.

Deux fichiers doivent être créés : le fichier .netrc avec les droits convenables et le fichier des directives de la commande, comme indiqué dans le cours.

8°) Lancer ce script périodiquement (commande *crontab* (Unix/Linux),*winat* (Windows)). Ne pas oublier de valider le script avant de l'insérer, avec son chemin complet, dans la liste des services à ordonnancer (crontab).

10°) A partir de la commande *rcp*, faire un transfert de fichiers entre deux stations du réseau. Décrire son comportement en fonction du contenu du fichier hosts.equiv.

A priori ne fonctionne pas en l'absence du fichier hosts.equiv. D'une façon générale, il ne devrait être possible de faire des transferts sans fournir de mot de passe entre postes de travail que sur des réseaux ou des postes dits de confiance donc sécurisés.

11°) Même question en utilisant le fichier *.rhosts*

Permet à un utilisateur de définir des utilisateurs de confiance à l'insu de l'administrateur ce qui peut être dangereux. Le fichier .rhosts permet donc de provoquer une faille dans la sécurité du système.

13°) Afficher la liste des serveurs de fichiers distants (serveurs NFS). Comment pouvez vous y accéder ?

A partir des commandes *mount*, *df* et du contenu du fichier *exports*, on peut déterminer les droits des stations clientes en lecture et en écriture sur les objets partagés.

Depuis un système Windows, on utilise les propriétés du voisinage réseau.

1.2 Outils de communication divers

1°) Etablir un dialogue avec votre voisin en utilisant les commandes *talk*, *write* et *mesg*.

Ces commandes sont les ancêtres de MSN.

2°) Se protéger d'intrusions intempestives (mesg n).

1.3 Interface graphique

1°) A partir de la commande *echo*, établir l'adresse d'affichage de l'écran virtuel associé à votre poste de travail.

Il s'agit de la variable shell DISPLAY obtenue à partir de la commande :

echo $DISPLAY

5°) Parmi tous les processus actifs, identifier les processus clients et le serveur d'affichage (X:0.0)

CHAPITRE 4

L'ARCHITECTURE DU SYSTEME DE COMMUNICATION TCP/IP

1. EXERCICES

2°) A partir des appels système de la couche *socket*, écrire une application permettant à deux processus client/serveur distants un dialogue simple.

```
/************************************************************
*    SERVEUR
*    Module : server.c
*    Domaine: AF_INET
*    Mode   : DATAGRAMME
************************************************************/
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define  UDP_PORT  3004
#define ever (;;)
main()
{int                 s, ns  ; /* descripteurs des prises */
 struct sockaddr_in  addr, addr_tmp ;  /* socket address */
 int                 addrlen=sizeof(struct sockaddr_in);
 struct hostent      *hp ;
```

```
 char                buf[128] ;

 if (!(fork()))
 {if ((s=socket(AF_INET,SOCK_DGRAM,0)) == -1)
 {perror(" Server: error Socket"); exit(2);}

 bzero((char *) &addr,sizeof(addr));
 hp = gethostbyname("localhost");
 addr.sin_family = AF_INET ;
 addr.sin_port   = htons(UDP_PORT);
 bcopy(hp->h_addr, (char *)&addr.sin_addr, hp->h_length);

 if (bind(s,&addr,addrlen) == -1)
 {perror("Server error BIND");exit(2);}
#ifdef DEBUG
if (getsockname(s,&addr_tmp,&addrlen) == -1)
{perror("Server error GETSOCKNAME");exit(2);}
printf("Server: family = %d, sin_addr = %ld , length = %d \n"
    ,addr_tmp.sin_family,addr_tmp.sin_addr,addrlen);
#endif

for ever
{if(!(recvfrom(s,buf,sizeof(buf),0,&addr_tmp,&addrlen)))
 break;
 else
 {strcat ( buf, " coucou",sizeof(buf));
  sendto(s,buf,sizeof(buf),0,&addr_tmp,addrlen);
 }
}
close(s);
}
}
```

```
/*************************************************************
*    CLIENT
*    Module : client.c
*    Domaine: AF_INET
*    Mode   : DATAGRAMME
*************************************************************/
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define  UDP_PORT  3004
#define ever (;;)

main()
{int                  s  ; /* descripteur du socket */
 struct sockaddr_in   addr, addr_tmp ;  /* socket address */
 int                  addrlen=sizeof(struct sockaddr_in);
 struct hostent       *hp ;

 char                 buf[128] ;

 if ((s=socket(AF_INET,SOCK_DGRAM,0)) == -1)
 {perror(" Client: error Socket");exit(2);}

 bzero((char *) &addr,sizeof(addr));
 hp = gethostbyname("localhost");
 addr.sin_family = AF_INET ;
 addr.sin_port   = htons(UDP_PORT);
 bcopy(hp->h_addr, (char *)&addr.sin_addr, hp->h_length);

#ifdef DEBUG
```

```
 if ( getsockname(s,&addr_tmp,&addrlen) == -1)
 {perror("Client error GETSOCKNAME");exit(2);}

 printf("client: family = %d, sin_addr = %ld , length = %d\n"
 ,addr_tmp.sin_family,addr_tmp.sin_addr,addrlen);
#endif

 printf ("Donnez le message : ");
 gets(buf);
 sendto(s,buf,sizeof(buf),0,&addr,addrlen);
 bzero(buf,sizeof(buf));
 recvfrom(s,buf,sizeof(buf),0,&addr,&addrlen);
 printf("Message recu : %s\n",buf);
 close(s);
}
```

3°) Sous Unix/Linux, identifier les modules réseau installés.

Commandes probe, modprobe, etc.

4°) Evaluer les ressources mémoires utilisées pour les accès réseau.

netstat -m

5°) On considère la chaîne de modules permettant de gérer les périphériques standard de saisie et d'affichage avec des claviers de type qwerty ou azerty, qui permettent un affichage des caractères de la gauche vers la droite. Comment utiliser simplement cette architecture pour gérer un affichage des caractères saisis de la droite vers la gauche.

Le principe est simple : il suffit d'écrire un nouveau module pilote qui s'intercale dans la chaîne des modules existante et qui permute la position de l'affichage des caractères.

CHAPITRE 5

LA FAMILLE DES PROTOCOLES TCP/IP

1. EXERCICES

1.1 Adressage IP et protocoles associés

4°) Pourquoi est-il nécessaire de faire "une résolution d'adresse" pour que deux stations puissent dialoguer ?

L'utilisateur fournit le plus souvent un nom de domaine qu'il convient d'associer à une adresse IP pour pouvoir communiquer; les en têtes des datagrammes ne contenant que les adresses IP de l'émetteur et du destinataire.

6°) Quel est le nom du protocole de résolution d'adresse du monde *TCP/IP* ? arp

7°) Sous quel nom de commande est-il accessible ? arp

8°) Quels sont les fichiers de configuration de l'adressage *IP* ? En déduire une cartographie du réseau en cours d'utilisation en indiquant le nom des stations, leur adresse *IP* et leur adresse physique.

Les fichiers hosts, networks, ou ceux de configuration des serveurs DNS, NIS, LDAP, etc.

10°)Indiquer le principe de fonctionnement du protocole ICMP ? Quel est son rôle? Avec quelle commande est-il implémenté ?

Principe du ping-pong : l'émetteur envoie un datagramme à une station (phase ping). Cette dernière, quand elle est active renvoie simplement ce datagramme (phase pong).

11°)Quels sont les problèmes posés par l'installation d'une carte réseau ?

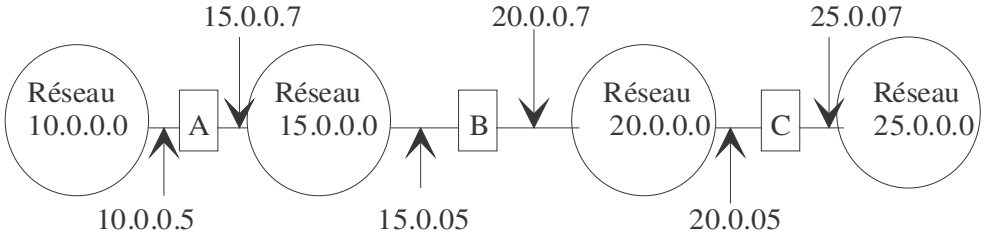
Il faut valider dans l'ordre les aspects matériels, le pilote, puis configurer le poste de travail pour l'intégrer dans un réseau ce qui est la réponse à la question suivante.

11°)Définissez une stratégie complète d'installation d'un réseau constitué de deux stations utilisant les protocoles *TCP/IP* au moins à savoir :

- ◊ problèmes d'installation,
- ◊ test de fonctionnement,
- ◊ stratégie d'adressage,
- ◊ mise en fonctionnement.

1.2 Routage

3°) Soit le réseau suivant :



A partir de la commande *netstat*, vous déduisez que les tables de routage du routeur B sont les suivantes :

Station du réseau	**Adresse du prochain routeur**
15.0.0.0	émission directe
20.0.0.0	émission directe
10.0.0.0	10.0.0.5
25.0.0.0	20.0.0.5

Votre réseau fonctionne-t-il correctement ?

Non. Le réseau 10 demeure inaccessible. Il faut remplacer l'adresse 10.0.0.5 par 10.0.0.7.

4°) Quels différences y a-t-il entre un algorithme de routage statique et un algorithme de routage dynamique ?

Des tables statiques ne sont pas modifiées au cours du temps. Elles sont donc utilisées avec des routes fixes.

Les tables dynamiques évoluent en fonction du comportement du réseau (protocoles IGRP, HELLO, etc.) et permettent de choisir la route la plus adaptée (notion de meilleur chemin) à chaque instant.

5°) Quand doit-on les utiliser ?

Routage statique sur un réseau statique (plutôt un réseau Intranet), routage dynamique autrement.

6°) Existe-il des protocoles capable de déterminer un "meilleur chemin ? Selon quels principes ? Comment sont ils appelés ? Indiquer des exemples.

IGRP,HELLO, SPF, EIGRP

7°) Qu'est ce qu'un sous-réseau. Quelle est son utilité ?

- Définir des réseaux publics ou privés constitués d'un nombre précis de poste de travail ce qui permet d'affiner le nombre de postes de travail des réseaux de classe A, B, C traditionnelles et d'améliorer les performances du routage.

- Regrouper des postes d'une entité.

9°) Qu'est ce qu'un système autonome ?

Un système doté d'une source d'autorité d'administration du réseau qui permet de définir une stratégie d'exploitation et d'administration.

10°) Différences entre un protocole de type EGP et un protocole de type IGP.

Les protocoles de type EGP sont utilisés par les routeurs entre des réseaux Intranet et Intranet.

Les protocoles de type IGP sont utilisés dans les routeurs des réseaux Intranet.

11°) Quelles sont les commandes et serveurs utilisés pour la gestion du routage. Quels sont les protocoles de routage sous jacents ? Indiquer des situations d'utilisation.

Les commandes classiques telles ping, route, netstat, ifconfig, ipconfig, arp, nslookup.

Le gestionnaire des services réseaux doit être actif ainsi que les serveurs correspondants (DNS, ICMP, ARP,…).

1.3 Protocoles de transport

1°) Quels sont les protocoles de transport essentiels utilisés dans le monde TCP/IP ? UDP, TCP.

2°) Indiquer leurs caractéristiques essentielles ainsi que les raisons de ces dernières ? classe 0 pour le protocole UDP et les réseaux Intranet, classe 4 pour le protocole TCP et les échanges de données garantis fiable sur Internet.

3°) Rappeler les principes de l'adressage au niveau transport. Dans quels fichiers peut-on trouver les adresses correspondantes ?

Les protocoles de transport utilisent des numéros de ports définis dans le fichier *services*.

Les serveurs accessibles sont décrits de le fichier *(x)inetd.conf*

4°) Différence entre une *trame*, un *datagramme* et un *segment*.

Ce sont les unités de données de protocoles des niveaux liaison (trame), réseau (datagramme), transport (segment).

1.4 Protocoles de haut niveau

1°) Rappeler le principe du protocole RPC. Quels sont les problèmes potentiels posés par son utilisation en cas de dysfonctionnement ?

Le protocole RPC permet d'effectuer des requêtes d'exécution de procédures sur des systèmes distants, nécessaires à l'exécution d'applications locales. Les conséquences d'un dysfonctionnement sont le blocage des applications locales ayant effectuées les requêtes distantes.

2°) En faisant une analogie avec le modèle OSI, quel est la couche correspondante associée au protocole RPC ?

La couche session.

3°) Mêmes questions avec le protocole XDR.

La couche XDR prend en charge le problème de la représentation de l'information sur différents postes de travail. En cas de disfonctionnement,

cette information peut devenir inexploitable car non correctement interprétée.

4°) Quels sont les applications immédiates de ces deux protocoles ?

Les services NFS, XDR, etc.

1.5 Protocoles d'administration de réseau

1°) Pourquoi est-il nécessaire de définir un protocole d'administration de réseau ?

Pour pouvoir administrer n'importe quel "objet administrable" à distance.

2°) Existe-t-il aujourd'hui un standard sur le marché ? Oui, SNMP.

4°) Qu'est-ce qu'un agent SNMP ?

Un poste de travail fournissant des informations aux gestionnaires SNMP.

5°) Rappelez les principes de fonctionnement du protocole SNMP.

Les objets à administrer sont gérés par un serveur d'informations décrite selon la norme ASN1 de la base de données (MIB) qui les transmet à la communauté des gestionnaires SNMP.

6°) Quelle est l'architecture utilisée ?

Le protocole UDP est utilisé avec les numéros de ports 161, utilisé pour les requêtes d'administration et 162 pour les notifications d'évènements issus des agents SNMP.

7°) Que représente le serveur SNMP ? le client SNMP ?

Le serveur SNMP fournit l'information au gestionnaire SNMP qui en est client.

8°) Comment appelle-t-on la base de données des objets SNMP ?

La MIB.

9°) Comment un objet de la base est-il accessible ?

A partir de l'arbre de nommage.

10°) Une station administrée peut-elle être protégée ? A quelle niveau ?

Toutes station administrable peut être protégée localement des gestionnaires SNMP, la granularité pouvant être le poste ou un de ses objets administrables.

11°)Les informations associées au protocoles SNMP sont elles "utilisables directement" ?

Pas vraiment car l'information de base est fournie en mode texte. C'est pourquoi il vaut mieux utiliser une interface graphique et les outils propriétaires associés.

CHAPITRE 6

OUTILS D'ADMINISTRATION DU MONDE TCP/IP

1. EXERCICES

1.1 Administration élémentaire

1°) A partir de la commande Unix/Linux *ps*, identifier les différents fournisseurs de services système et réseau.

L'exécution des commandes

ps, *ps –e*, *ps –ealf*

affiche les différents serveurs actifs.

3°) Sous Unix/Linux, utiliser le gestionnaire des services (processus *init*) pour changer de mode de fonctionnement (mono-utilisateur, multi-travaux). On analysera en détail le fichier *inittab*. Idem sous Windows à partir des onglets de la commande *msconfig*.

Il suffit, à partir d'un niveau donné, d'exécuter la commande

init n

n étant le niveau d'exécution souhaité.

4°) Intégrer le démarrage d'un nouveau service qui souhaite la bienvenue à l'ensemble des utilisateurs du réseau. Ne pas oublier la procédure d'arrêt et de redémarrage.

Il faut intégrer un script supplémentaire dans le répertoire associé au niveau d'exécution souhaité. Sa dénomination implique l'ordre de son exécution, par exemple le script S99essai suivant :

#!/bin/bash

echo "Bonjour"

La procédure K99essai d'arrêt pourrait être la suivante :

#!/bin/bash

echo "au revoir"

1.2 Sauvegardes

■ **La commande tar**

1°) Effectuer une sauvegarde de votre répertoire sous */tmp.`logname`* où *`logname`* est votre identificateur de connexion.

cd

tar cvf /tmp/sauve.`logname` .

2°) Restaurer la sauvegarde d'un de vos voisins dans le sous-répertoire *tempo* (à créer) dans votre répertoire de travail.

cd

mkdir tempo

tar xvf /tmp/sauve.nom_voisin

3°) A partir de 2 fichiers *image_tar*, faire un fichier **.tar* puis le compresser avec la commande *compress* (fichier **.tar.Z*).

tar cvf cible.tar f1 f2 | compress

4°) Compresser les 2 fichiers *image_tar* de la question précédente et fabriquer un fichier **.Z.tar*. Comparer sa taille à celui de la question précédente. Conclusion.

compress f1

compress f2

tar cvf cible.tar.Z f1.Z f2.Z

La commande ls –l indique une taille supérieure du fichier contenant les compressions successives.

5°) A partir des commandes Unix/Linux *tar*, *rsh* et *dd*, effectuer la sauvegarde d'un répertoire local sur une station distante puis sa restauration.

C'est l'exemple du § 5.4 du chapitre.

6°) Même question avec un répertoire distant sauvegardé puis restauré localement.

Une solution est de faire un montage distant par NFS puis d'effectuer la sauvegarde directement.

CHAPITRE 7

MISE EN OEUVRE DES SERVICES

1. EXERCICES

1.1 Base de données réseau

1°) Définir la base de données réseau.

La base de données réseau est constituée de toutes les informations nécessaires au lancement et à l'administration des différents services réseaux à gérer au niveau du transport et du routage.

2°) De quels fichiers est-elle constituée dans le monde *TCP/IP* ? Quelles en sont les données représentées ?

Gestion des adresses IP, des routeurs, des services DNS et des fichiers de configuration des différents protocoles utilisés…Bref, beaucoup de monde.

3°) Stratégie de choix des adresses *IP* dans une entreprise, selon que son réseau est ouvert ou non sur le monde extérieur.

Il faut choisir la classe du réseau, définir les adresses privées, les masques de sous réseaux, définir les services à installer et les serveurs ad hoc (puissance, sécurité, sauvegarde, etc..), demander des adresses publiques, installer les protocoles de protection et de sécurité

4°) Rôle du fichier *hosts.equiv*.

Définition de réseau de confiance, de groupes de confiance (machines, utilisateurs), d'utilisateur de confiance sur un réseau donnée, sur une machine donnée.

5°) Après consultation du fichier *hosts.equiv*, lancer les r-commandes usuelles sur des cas simples. Demander ensuite à votre ingénieur système favori de

supprimer (s'il existe) ou d'installer (s'il n'existe pas) le fichier *hosts.equiv* et relancer les r-commandes précédentes. Conclusions ?

Ces commandes fonctionnent simplement sur un réseau de confiance avec des ordinateurs de confiance. Elles deviennent plus complexes d'utilisation si la confiance ne règne pas. Elles peuvent même ne pas fonctionner du tout.

6°) Après suppression du fichier *hosts.equiv*, constituez vous un fichier ~/.rhosts. Relancer les commandes précédentes. Conclusions.

Le fichier .rhosts permet à un utilisateur de définir des utilisateurs de confiance venant d'une machine de confiance qui pénétreront dans le système avec son identité. La présence de ce fichier peut représenter une faille dans la politique de sécurité des accès.

7°) Quel est le rôle du gestionnaire de services *inetd* ? Analyser la liste des processus actifs puis lancer une connexion *telnet*. Analyser à nouveau la liste des processus actifs. Conclusions ?

Le gestionnaire des services réseau lance la plupart des serveurs d'objet réseau dynamiquement. Il suffit de consulter la liste des services réseau actifs avant le lancement d'un nouveau service, pendant et après.

8°) En utilisant l'architecture du gestionnaire de services *inetd*, comment pouvez vous simplement intégrer un nouveau service accessible aux différents utilisateurs.

Il suffit de développer les codes serveur et client, de choisir un numéro de port, et d'intégrer les informations correspondantes dans les fichiers services (paramétrage du client) et inetd.conf (paramétrage des serveurs).

9°) Que faut-il faire si on veut installer un service de nom de domaine ?

Cette opération est complexe et délicate. Le plus simple est d'utiliser une interface graphique genre Webmin, de définir les relations nom de domaine et adresse IP ainsi que les relations inversées. Il faut ensuite définir le serveur maître primaire, un éventuel serveur secondaire. Il faut aussi définir les passerelles, les serveurs DNS externes interrogés.

L'usage des outils tels ifconfig, ipconfig, netstat, ping, nslookup… est fortement recommandé.

10°) Enumérer les serveurs nécessaires au fonctionnement des couches basses.

Les couches basses sont gérées à partir du gestionnaire des services réseaux, du serveur de résolution d'adresse, éventuellement d'un serveur DHCP, d'un

serveur DNS (non indispensable s'il en existe un par ailleurs ou si d'autres outils sont utilisés comme le service NIS, ou simplement des fichiers hosts.

11°) Quels sont les scripts de démarrage des services réseau.

Dans le monde Unix/Linux, ces scripts sont dans le répertoire /etc ou dans un des ses sous répertoires (/etc/init.d/initxx.d), numérotés par ordre croissant.

Dans le monde Windows, il suffit d'exécuter l'utilitaire msconfig pour accéder aux paramètres de démarrage des services.

12°) L'ordre de démarrage des différents processus est-il important ? Pourquoi ?

Cet ordre est fondamental dans la mesure où il est faut éviter "de mettre la charrue avant les bœufs" ou plus précisément, éviter de démarrer des services dont le bon fonctionnement dépend d'autres services non encore démarrés.

1.2 NFS et NIS

1°) Quel sont les services de niveau inférieur nécessaires aux protocoles NFS et NIS ? Rappeler leur rôle.

Le protocole de niveau session RPC et celui de présentation XDR.

Le protocole RPC gère la synchronisation des requêtes locales et distantes.

Le protocole XDR gère les problèmes liés à la représentation non universelle de l'Information.

2°) Quel sont les processus nécessaires au fonctionnement d'un serveur NFS? Rappeler leur rôle.

Ce sont les serveurs d'objets (nfsd), les serveurs de cache réseau (biod), les serveurs de montage (mountd).

3°) Quel sont les processus nécessaires au fonctionnement d'un serveur NIS ? Rappeler leur rôle.

Le gestionnaire du domaine NIS (domainname), le serveur de listes (ypserv), le client des listes (ypbind).

On peut ajouter à cette liste simple l'utilitaire rpcinfo, le gestionnaire de port (portmapper), les utilitaires de maintenance des listes d'objets (yppush, ypxfr, ypinit, ypcat).

4°) La sémantique des appels systèmes d'entrée/sortie est elle conservée par NFS ? Qu'en pensez vous ?

Non! Les problème induits pour effectuer des requêtes d'entrées/sorties dans un réseau sont nombreux : gestion des accès concurrent, gestion complexe des synchronisations distantes des opérations d'entrées/sorties, gestion de la représentation interne des objets. Et ils ne sont qu'imparfaitement résolus par le protocole NFS.

5°) Quel est le rôle du fichier `exports` ?

Ce fichier contient la liste des objets exportables ainsi que les listes de confiance associées (utilisateurs de confiance, postes de confiance, etc.).

6°) Vous voulez monter la racine du serveur NFS A sur celle du serveur NFS B et réciproquement. Citez précisément l'ordre des séquences à réaliser pour réaliser ce montage croisé. Que se passe-t-il si une des stations est brutalement arrêtée ?

L'idéal serait que chacun des serveurs ait démarré avant l'autre ce qui ne peut être réaliste.

On procède donc de la manière suivante :

- démarrage successif des postes A et B et de leurs serveurs d'objets respectifs,

- montage distant de B sur A et montage distant de A sur B

Le "plantage" de l'un provoque le "plantage" de l'autre ("plantage" mutuel…).

7°) A votre avis, vaut-il mieux faire un montage NFS *hard* ou *soft* ? Pourquoi ?

Aucun n'est parfait. Le montage hard est bloquant car il réitère la requête jusqu'à sa satisfaction. Le montage soft s'arrête simplement si la requête n'est pas satisfaite et provoque alors le blocage de la requête cliente.

8°) Rôle du service NIS.

Serveurs des listes d'objets distribués de la base de données réseau (utilisateur, nom de domaine, adresses IP, alias de messagerie, groupes, etc.).

9°) Avez-vous intérêt à disposer de plusieurs serveurs NIS sur un même réseau ? Quelles règles de sécurité devez vous respecter ? Cette approche

est-elle une règle générale dans les systèmes client-serveur. Citez d'autres exemples d'utilisation d'un tel concept.

Oui bien sûr. L'idée est de disposer d'au moins un deuxième serveur de sécurité qui, vis-à-vis des requêtes clientes, se comporte de la même façon que le serveur de base. La seule différence (essentielle) est que ce serveur de sécurité ne détient la source d'autorité que par délégation, avec des droits minimum. On retrouve ce concept avec des services d'architectures similaires (serveur primaire et secondaire (DNS), serveur maître et de sécurité (domaine microsoft), serveur maître et esclave (NIS), serveur primaire et secondaire (messagerie), etc.).

10°) Que se passe-t-il si vous voulez ajouter une station sur le réseau ? Indiquez la séquence des instructions à exécuter pour l'ajouter dans la liste des stations.

Quand il n'existe aucun serveur de l'information, il faut l'implémenter sur tous les postes de travail ce qui est long, laborieux, peu sécurisé, peu fiable….

Avec un serveur adéquat (NIS par exemple), il suffit d'ajouter l'information dans le fichier (centralisé) ad hoc, par exemple /etc/hosts, puis de mettre à jour la base de données réseau, comme indiqué dans le cours :

cd /var/yp/nom_domaine

make liste

La liste modifiée est alors dispatchée sur le réseau.

11°) Comment pouvez vous connaître le nom des serveurs NIS ?

Commande ypwhich

12°) Comment pouvez vous en choisir un autre ?

Commande ypset

13°) Pouvez vous ajouter une nouvelle liste (personnalisée) à l'ensemble des listes classiques associées au service NIS ? Par quelles commandes ?

Oui, il suffit de respecter la structure de la base de données NIS (format dbm).

14°) Que pensez vous de la notion de groupe de stations ?

Délicate à manier et peu performante aujourd'hui.

15°) Installer des quotas d'utilisation sur un système de fichier exporté.

Il faut créer le fichier quota et utiliser les commandes edquota, repquota, quotaon, quotaoff, quotacheck

1.3 Messagerie

1°) Rappeler les noms des principaux agents utilisateurs que vous connaissez ainsi que les systèmes respectifs d'implémentation. Protocole associé.

mail, xmail, mailx, msn, netscape communicator, outlook, outlook express, eudora…

2°) Quel est l'*agent de transport* associé au service de messagerie. Protocole associé.

L'agent de transport, dans le monde Unix/Linux est implémenté par le serveur sendmail, qui respecte le protocole SMTP.

3°) Indiquer son paramétrage pour qu'il soit réveillé toutes les 3 minutes.

4°) Que penser vous du fichier `sendmail.cf` ? Comment vous y prendriez vous pour le configurer.

Les règles de grammaire permettant de décrire l'ensemble des processus d'adressage et de nommage des différents systèmes de messagerie sont très complexes (monde Apple, Unix/Linux, Windows, IBM, Netware, LDAP, etc.). Les autres concepts (bureau de poste, postier, agent de relais de messagerie, etc.) complètent ces règles qu'il convient de parfaitement maîtriser (2s minimum) ou d'adapter à partir d'un fichier correct, existant.

5°) Quels sont les droits d'accès au répertoire des boites aux lettres publiques ? Pourquoi ?

Ecriture autorisée pour écrire un courriel, ainsi que lecture autorisé pour sa lecture, tout en évitant les abus (lecture, modification, suppression) des courriels d'autrui.

6°) Pensez vous qu'il judicieux de partager un serveur de messagerie sur un réseau comportant un nombre important de stations ? Dans l'affirmative, comment vous y prendriez vous ?

Bien sur, chacun des poste de travail devenant un simple client d'un serveur de messagerie accessible par tous les clients dont les boites aux lettres y sont centralisées.

1.4 Installation

1°) Rappeler la commande de configuration d'un contrôleur ?

ifconfig, ipconfig

2°) Quelles sont ses principales options ? Rappeler leur utilité.

Choix du réseau (internet, XNS)

3°) Quand son utilisation est-elle nécessaire ?

Pour paramétrer un contrôleur

4°) Rappeler la définition d'une station sans disque et d'un terminal graphique XWindow. Quelles en sont les contraintes d'installation et d'administration.

Une station sans disque est une station qui télécharge l'ensemble de ses informations et dont le disque (virtuel) doté de son système n'est accessible que par le réseau.

Un terminal X est un terminal graphique qui accède aux données centralisées d'un serveur.

CHAPITRE 8

ADMINISTRATION SECURISEE

1. EXERCICES

1.1 Gestion des utilisateurs

1°) Créer des utilisateurs sur un système local. On s'attachera à définir leur répertoire de travail, leur groupe(s) de travail, leurs privilèges d'administration et de programmation.

Il faut définir soigneusement les règles d'identification des utilisateurs, des groupes, leur mot de passe, les profils utilisateurs, les règles de sécurité et de propriété, en utilisant une interface ad hoc.

2°) Essayer de créer un utilisateur synonyme de l'administrateur.

Cette opération est manuelle le plus souvent. Elle nécessite de créer un utilisateur dont toutes les caractéristiques sont identiques à celle de l'administrateur à l'exception de son nom et de son mot de passe. Sous Unix Linux, il faut donc modifier à la main les fichiers passwd, group, et shadow.

3°) Verrouiller la commande *su*.

Dans le monde Unix/Linux, il faut créer un groupe d'utilisateurs autorisés à utiliser la commande, par exemple su, puis modifier le groupe d'appartenance de la commande (commande chgrp).

4°) L'utiliser et constater son audit par le système.

Il faut créer le fichier sulog dans le répertoire /var/adm.

5°) Mêmes questions dans un réseau Intranet fonctionnant en *Workgroup*, *Domaine* (NIS ou microsoft).

La question ici est de définir un unique administrateur du réseau avec un identificateur et un mot de passe unique. On procède en deux temps : définition sur le serveur de la liste des utilisateurs d'un utilisateur synonyme de l'administrateur, puis diffusion de la liste sur le réseau. Remarquons que l'administrateur du réseau créé n'a pas intérêt à perdre son mot de passe…

1.2 Attributs de sécurité

1°) Déplacez vous dans l'arborescence d'UNIX pour étudier les répertoires /, `/etc`, `/usr`, `/home` et essayer d'y détruire des fichiers au hasard (vous n'êtes pas *root* quand vous exécutez cette commande!!).

Si le système est bien configuré, il n'y aura pas de surprise tous ces fichiers système étant protégés. Sinon, le pire est à redouter.

2°) Rappeler les fonctions des commandes Unix/Linux `chmod`, `chgrp`, `umask`, `basename`, `file`, `du`, `size`, `od`.

3° a) Définir le masque de création de fichiers 777.

b) Créer un fichier *liste* à partir de la commande ls -l > liste

c) Quels sont ses droits ?

d) Créer un répertoire *tempo* et constater ses droits. Conclusion.

Ce masque de protection est paranoïaque car il interdit tout accès à un fichier ou répertoire.

4°) Refaire la question précédente avec le masque 000. Conclusion.

Ce masque est une passoire qui autorise toutes les fantaisies à n'importe qui.

5°) Aller chez votre voisin et essayer de lui détruire un de ses fichiers ou répertoires (de préférence inutile). Conclusion ?

Si vous pouvez détruire un fichier de votre voisin, dites vous qu'il peut faire la même chose chez vous. Vous avez donc intérêt à vous protéger (commande chmod). Eventuellement, prévenir l'administrateur qu'il y a un problème de sécurité dans le système.

6°) Que pensez vous du masque 022 ?

Ce masque est raisonnable et ouvert (tout accès en lecture est autorisé à quiconque, tout accès en écriture interdit à quiconque sauf au propriétaire.

7°) Afficher les caractères non imprimable d'un fichier texte (commande od).

8°) Quels sont les attributs de sécurité et de propriété du répertoire */tmp*. Essayer d'y détruire un fichier au hasard. Conclusion ?

On constate que ce répertoire est d'accès public en lecture et en écriture mais qu'il est, en principe (bit t) impossible de détruire les fichiers des autres utilisateurs ce qui correspond à des règles de coexistence pacifique.

9°) Comment positionner le bit *t* sur le répertoire */tmp* ?

chmod 4777 /tmp

1.3 Montage

1°) Analyser le fichier des descriptions des systèmes des fichiers à monter sur votre système. Analyser en détail les attributs de sécurité (*ro*, *rw*, bit *s*, etc.).

Utiliser les commandes mount, df, consulter le fichier fstab.

2°) Vous n'êtes pas administrateur. Essayer de *démonter* ou de *monter* des systèmes de fichiers au hasard. Idem si vous êtes administrateur.

En principe impossible, pour des raisons de sécurité, sauf peut être pour des périphériques personnels sur bus USB. Utiliser les commandes précédentes.

1.4 Cheval de Troie

Essayer de programmer un cheval de Troie avec un (faux) script (par exemple *su*).

L'idée est de laisser traîner un script dans un répertoire d'accès public qui piège un utilisateur qui voudrait changer d'identité (l'administrateur par exemple), qui s'exécuterait si ledit utilisateur exécute en priorité les scripts du répertoire courant. Ce dernier l'évite par modification dans sa variable PATH de la position du répertoire courant (le plus à droite possible) de telle sorte que

PATH=$HOME:/bin:/….:.

1.5 Démarrage et arrêt du système

1°) Quel est le répertoire essentiel utilisé au démarrage du système ?

/etc

2°) Analyser le fichier *inittab* (Linux/Unix) et le résultat d'exécution de la commande *msconfig* (microsoft).

Voir le cours.

3°) Si possible, sur un terminal en mode texte, tuer le processus de *login*. Constater qu'il se relance "spontanément".

Les lignes correspondantes dans le fichier inittab indiquent une exécution en mode "respawn" (tâche "perpétuelle").

4°) Visualiser les répertoires et fichiers de démarrage.

Il faut visiter les répertoires /etc, /etc/init.d et chacun de ses sous dossiers.

Sous Windows, exécuter msconfig et analyser les différents onglets (services au démarrage, service réseau, fichier*.ini, etc.).

5°) Un arrêt et un redémarrage d'une ou de deux stations seront réalisés en fin de session. Analyser le démarrage à froid puis le démarrage à chaud avec les commandes adéquates.

Sous Unix/Linux, plusieurs commandes similaires possibles : init, boot, reboot, shutdown…

Le plus simple est d'utiliser init

init 0	arrêt
init 1	mode maintenance
init 6	redémarrage en mode par défaut

Sous Windows, utiliser msconfig pour désactiver ou activer un service au démarrage.

6°) Modifier le fichier de démarrage de session pour y ajouter une fonction *logout* exécutée automatiquement lors de la déconnexion et effaçant l'écran.

Indication : définir la fonction *logout* puis utiliser la commande UNIX *trap*.

1.6 Sauvegarde distante

Analyser la commande Unix/Linux :

tar cvfb 20 - . | rsh host_distant dd of=/dev/rmt0 obs=20b

La première partie de la commande effectue, sur la sortie standard, la sauvegarde du répertoire courant.

Le flot de données est redirigé (tube) comme entrée de la commande d'exécution distante (rsh) du transfert (dd) vers le média de sauvegarde (/dev/rmt0), avec un facteur de groupage adéquat (20).

1.7 Antivirus et pare-feu

Sous Windows, installer les logiciels du domaine public Anti-Vir, Kerio Firewall, etherreal pour analyser des captures.

L'idée ici est simplement de télécharger un anti-virus du domaine public et un pare feu d'utilisation gratuite, de les paramétrer pour sécuriser un poste de travail.

L'utilitaire du domaine public etherreal permet de capturer et d'analyser des séquences de navigation sur le net.

CHAPITRE 9

KERBEROS

1. EXERCICES

1°) Rappeler les principes élémentaires de chiffrement (clé publique, clé privée).

Revoir les concepts rappelés.

2°) Quels sont les objectifs d'un système comme Kerberos.

Définir un juge de confiance d'authentification ce qui sécurise les transactions sur des réseaux intranet ou sur internet.

3°) Définir un ticket initial, un ticket de session, un serveur de ticket garanti.

Le ticket initial est fourni par le juge. Après authentification par le serveur de ticket garanti, il donne droit aux tickets de sessions nécessaires.

4°) Rappeler l'architecture de Kerberos.

Les points précédents.

5°) Installer et valider Kerberos sur un serveur de téléchargement Unix/Linux ou Windows.

L'idée ici est simplement d'installer une base de donnée kerberos et de la paramétrer, seul moyen de valider ses connaissances.