IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

TP de sécurité Unix et réseau en salle réseau

UE Cryptographie et Applications

Printemps 2024

Disclaimer

Ce TP propose quelques tâches liées à l'administration sécurisée de machines Unix et de réseaux IP. Prenez le temps de lire (et comprendre) les consignes avant d'agir.

Avertissement : Ce TP vous prépare à être éventuellement administrateur système et réseau dans votre vie professionnelle future. Il est clair que les techniques exposées ici sont à utiliser selon l'éthique de cette profession et non dans des buts illicites !

Vous avez des suggestions, des idées pour améliorer ce TP ? Nous les attendons avec impatience !

Note pour les bidouilleurs indomptables : On vous aime, mais vous êtes priés de laisser les machines dans l'état où vous auriez aimé les trouver en arrivant ! S'il vous prend l'envie de changer la config de quoi que ce soit qui n'est pas indiqué dans le TP, pourquoi pas, faites-vous plaisir, mais veuillez rétablir la configuration initiale en fin de séance (même si c'est juste la couleur du fond d'écran). De même, il est interdit d'écrabouiller la configuration initiale des routeurs en mémoire flash (i.e. Interdiction de faire des `write mem`).

1 Prise en main du banc de test

Le réseau que nous allons utiliser tout au long de ce TP a la structure indiquée en figure 1 qui permet de simuler un réseau d'entreprise avec un filtrage de certains paquets IP.

On commencera par vérifier le câblage du réseau au cas où les encadrants se seraient trompés.

La configuration a déjà été faite... enfin... normalement... Il est plus prudent de vérifier (commandes `/sbin/ifconfig`, `ip address`, `/sbin/route -n`, `ip route`). Aussi, le cas échéant (mais c'est déjà fait), pour configurer le réseau de chaque machine, passer sous `root` et lancer le script de configuration avec :

```
cd /opt/TP/TPSec
./configure
```

En principe, le routeur a été également configuré par ce script à partir du fichier `configrouteur`. Vous pouvez vérifier cela depuis l'ordinateur relié au routeur (donc ne marche pas sur tous les ordinateurs, notamment ceux du milieu...) avec l'outil `gtkterm` (ou `minicom`) : dans cette console d'administration, vous pouvez taper des commandes pour le routeur (voir annexe A) :

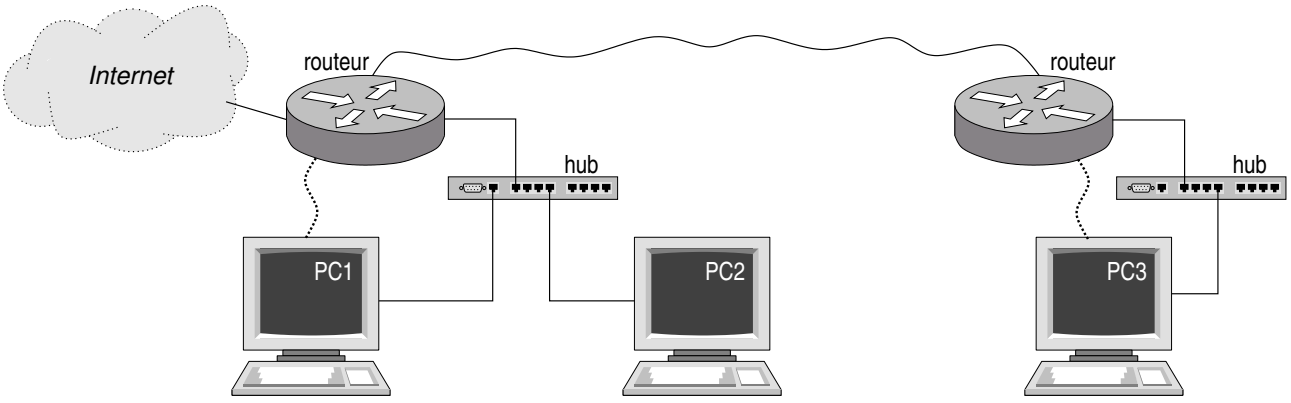


FIGURE 1 – Architecture du réseau expérimental

```
cisco> enable
Password:
cisco# show running-config
```

Récupérez l'adresse IP de votre machine et des machines des voisins (commande `/sbin/ifconfig`, `ip address`). Vérifiez les routes (`/sbin/route -n`, `ip route`). Vérifiez que vous pouvez joindre vos voisins avec quelques `ping`.

La connexion à Internet devrait également marcher grâce à de la translation d'adresse réalisée par le routeur de la salle.

2 Étude de protocoles non sécurisés

2.1 Concepts

L'exercice suivant va illustrer un autre problème avec les mots de passe : comment est-ce qu'ils sont échangés entre le client et le serveur.

Afin de se convaincre d'utiliser des protocoles sécurisés, nous allons observer en quoi consistent les protocoles «de base» existant sur Internet depuis l'origine des temps.

Dans un premier temps on peut lancer un programme `tcpdump` ou `wireshark` [1] et faire un `telnet` [2], un `ftp` ou utiliser un navigateur WWW entre 2 machines. Que constatez-vous ?

Suggestion d'un élève d'un TP précédent : demandez à votre binôme de commander quelque chose sur son site Internet préféré avec sa carte bancaire et son code secret et regardez pendant ce temps-là ce qui passe sur le réseau pour vérifier le niveau de sécurité. ☺

2.2 Utilisation de `dsniff`

On va automatiser le concept en utilisant le logiciel `dsniff` qui est capable de reconstruire le contenu d'une communication utilisant de nombreux protocoles.

Lancer sur le `PC2` un

```
dsniff -m -i eno2
```

et essayer de vous connecter par exemple du `PC3` vers le `PC1` avec un protocole non sécurisé style `telnet` ou `ftp` (ne tapez pas votre vrai mot de passe ☺). Que constatez-vous à la fin de la connexion ?

Variante : à défaut d'intercepter le code de carte bancaire du binôme qui se commande une pizza (qui, espérons-le, transitera dans du https), vous pourrez au moins connaître la pizza qu'il s'est choisit : sur le `PC2` lancez la commande `driftnet -i eno2` et observez ce qu'elle affiche lorsque votre binôme *surf sur le web* depuis `PC1` en HTTP (pas en HTTPS) .

2.3 Conclusion

Mettre à la poubelle les protocoles non sécurisés `telnet`, `ftp`, `rsh`,... et utiliser à la place `ssh`, `scp`, `sftp`,...

Imaginez les attaques faisables sur un réseau sans fil non sécurisé avec de l'espionnage de paquets ou des attaques par interception au milieu : sécurisez la configuration réseau et utilisez de toute manière des protocoles sécurisés.

3 Filtrage IP

La section précédente a montré clairement les limites des protocoles simples, mais non sécurisés. Une première limitation va être d'en restreindre l'usage depuis l'extérieur par exemple pour 2 raisons :

- si quelqu'un, même de bien intentionné, les utilise, il peut être espionné et quelqu'un pourra se connecter à sa place ;
- autant limiter au strict minimum les connexions extérieures afin de limiter un risque lié à l'exploitation d'un trou de sécurité dans un protocole inutilisé.

Le filtrage concernera sur la figure 1 la machine PC3.

1. Ce programme a un mode intéressant de reconstruction de sessions TCP plus compatible avec les cerveaux humains dans le menu `Analyze/Follow TCP Stream`.
2. Vous constaterez que nos serveurs telnet sont un peu longs à ouvrir une session. En effet ils tentent de faire une résolution de nom inverse sur l'IP du client pour en apprendre un peu plus sur lui... Or nos machines ne sont pas dans le DNS, dommage !

3.1 Outil d'analyse de ports ouverts `nmap`

⚠ Attention : certains administrateurs système considèrent que le simple fait qu'on regarde quels sont les ports ouverts constitue une violation de leur vie privée...

Parmi les options utiles, `nmap -O` permet de deviner quel est le type d'une machine, `-sS` permet de faire un test de ports assez rapide, `-P0` permet de faire un test même si la machine ne répond pas au `ping`.

Essayer cet outil pour tester les ports IP ouverts sur diverses machines. Comparer par exemple `www.enst-bretagne.fr`, `www.telecom-bretagne.eu`, `www.emn.fr`, `www.imt-atlantique.fr`, `www.microsoft.com`.

On peut analyser des réseaux entiers comme `enstb.org/25` par exemple.

Il existe des interfaces graphiques (`xnmap`, `nmapfe`, `zenmap`) pour les amateurs de cliquodrome.

Éventuellement lancer un outil d'espionnage style `tcpdump` ou `wireshark` en même temps pour comprendre comment `nmap` peut fonctionner aussi rapidement malgré les temporisations protocolaires (TCP,...). En particulier on peut comprendre la différence entre les aspects `closed` et `filtered`. Dans le dernier cas ce n'est pas le protocole style TCP qui répond, mais le protocole ICMP annonçant l'interdiction. Ce sera pratique dans la suite pour mettre au point les filtrages.

3.2 Filtrage avec syntaxe à la CISCO

Cette partie nécessite un routeur CISCO ou un logiciel à syntaxe compatible tel que Zebra ou Quagga.

3.2.1 Syntaxe CISCO de base

Voir la documentation du constructeur, le cours de sécurité et les TPs précédents. À défaut, reportez-vous aux annexes A et B.

Pensez à l'utilisation de  [3] pour la complétion automatique et  [4] pour l'affichage de l'aide contextuelle.

On se connecte sur le CISCO par la liaison série en utilisant le logiciel `gtkterm` (ou `minicom`).

Tapez  [5] puis identifiez-vous. (Le mot de passe du routeur vous sera donné en séance.)

Ensuite, pour passer en mode d'administration sur le routeur :

```
enable
```

Pour afficher la configuration actuelle :

```
show running-config
```

et la configuration plus précise des listes de contrôle d'accès avec l'utilisation de chaque règle :

```
show access-lists
```

Pour entrer dans le mode de configuration interactive :

```
configure terminal
```

puis pour en sortir et valider la configuration tapée :

```
exit
```

Pour configurer chaque ligne de l'`access-list` 100 par exemple :

```
access-list 100 contenu d'une ligne d'access-list
```

Pour supprimer l'`access-list` 100 :

```
no access-list 100
```

Pour entrer dans la configuration d'une interface :

```
interface le-nom-de-l'interface
```

pour associer par exemple la liste de contrôle d'accès 100 à cette interface :

```
ip access-group 100 in-or-out
```

et en sortir :

```
exit
```

⚠ Une fois rajoutée une `access-list` à une interface, tout le trafic est bloqué tant qu'on ne l'a pas explicitement autorisé avec des règles de filtrage !

3. La touche `<TAB>` du clavier.
4. La touche `<?>` du clavier.
5. La touche `<TENTRÉE>` du clavier.

Note : Il est également possible de *rentrer* dans l'édition d'une access-liste avec `ip extended access-list 100`. Cela permet d'ajouter des lignes, voire d'en supprimer. Il est également possible de réordonnancer la liste avec `ip extended resequence 100 ...` (Je vous laisse chercher dans la doc Cisco.) Cependant, pour la suite des manipulations, il peut être plus simple de préparer ses access-listes dans un fichier texte, puis de faire un copier-coller à la souris lorsque l'on pense être ok...

3.2.2 Filtrage simple

Proposez un filtrage qui interdit les connexions `telnet` (port TCP 23), et autorise tout le reste.
En fonction du temps disponible, réalisez les filtrages des questions suivantes.

3.2.3 Scénarios de filtrage

1. Filtrage Web. On veut éviter que nos utilisateurs se connectent directement aux serveurs web externes, mais qu'ils utilisent le proxy de l'établissement (qui fait du cache). Pour cela, on interdit la sortie du trafic WWW (protocole http (port TCP 80) et https (port TCP 443)), et on autorise le trafic WEBCACHE (port TCP 8080) seulement vers le mandataire WWW de la salle (`proxy.tp-reseaux.enstb.org`, cette machine est définie dans la section **??**). En obliger l'usage permet de mieux exploiter ce cache global. [6] Un tel mandataire global pourrait aussi servir à filtrer du contenu [7].
2. Autres connexions sortantes. On autorisera toutes les connexions sortantes pour ne pas emprisonner les utilisateurs [8].
3. Connexions entrantes prohibées. En entrée on laissera le port `ssh`, le WWW et le DNS (`domain`). On interdira spécifiquement les `telnet`, `rsh` et autres `rlogin` ou `ftp`.

Pour trouver l'inspiration sur les noms des services et leur numéro, regarder dans `/etc/services` ou bien comme d'habitude chercher dans le WWW.

Dans la suite, on fera des expérimentations avec `nmap` pour vérifier depuis l'intérieur ou l'extérieur ce qu'on peut faire réellement.

Remarquez au passage l'intérêt d'avoir un compte `root` à l'extérieur ne serait-ce qu'un accès par modem à un fournisseur d'accès Internet pour faire de l'audit du réseau avec des outils comme `nmap`, etc.

3.2.4 Faire une configuration libertaire

Dans cette approche, on n'interdit que le trafic potentiellement dangereux ou non sécurisé. C'est bien pour les utilisateurs, mais dangereux, car le moindre trou de sécurité à la mode a le plus de chances d'être exploité.

3.2.5 Faire une configuration paranoïaque

On commence par tout interdire.
Seuls les services indispensables seront ouverts.

3.3 Utilisation d'iptable

Les manipulations précédentes étaient sur les routeurs. Mais on peut aussi mettre un *firewall* sur une machine terminale, c.à.d. pour protéger une machine si elle est raccordée à un réseau douteux.

Pour linux, cela se passe historiquement avec `iptables` (et `ip6tables`, voir `ebtables`).

Notez que depuis quelques temps déjà la communauté Linux migre progressivement de `iptables` vers `nftables`, considérées plus efficace et plus souple d'utilisation. [9]. La couche réseau dans le noyau est la même (*Netfilter*), mais la syntaxe est différente et vise à unifier les différents types de tables (IPv4 IPv6 Ethernet ...)

En fonction du temps disponible...

Refaire une manipulation similaire sur les postes terminaux avec `iptables` (voir annexe C).

Très pratiques les scripts `iptables-save` (affiche/sauve la configuration courante) et `iptables-restore` (recharge toute une configuration `iptables`).

6. Pour vos essais, pensez à configurer votre navigateur en conséquences (*Préférences / Réseau / Proxy*).
7. Dangereux pour l'ordinateur, voire pour le rendement de l'utilisateur... ☺
8. Sachant que de toute manière un utilisateur pourra toujours avoir un modem dans son bureau ou tirer un tunnel sur n'importe quel protocole autorisé et ouvrir un canal caché... Donc rester raisonnable dans les restrictions et plutôt convaincre les utilisateurs.
9. `https://wiki.nftables.org/wiki-nftables/index.php/Quick_reference-nftables_in_10_minutes`

stateless vs. stateful firewall Jusqu'à maintenant on a réfléchi au niveau des paquets. Le routeur prend sa décision seulement en considérant le paquet ; il ne garde pas d'information en mémoire concernant les paquets précédents : c'est un firewall stateless. Par contre, on peut vouloir des règles plus fines : p.ex. *«Interdire tous les paquets entrants, sauf s'ils sont en réponse à une requête sortante.»* Pour faire cela, il faut que le firewall ait l'intelligence suffisante pour comprendre qu'un paquet est une requête, et qu'un autre paquet est une réponse, et que telle réponse est liée à telle requête précédente. Il faut donc qu'il fasse du suivi de connexion et qu'il garde des informations en mémoire : c'est un firewall stateful. Dans le vocabulaire Linux, on parle de `conntrack` (connexion tracker). L'annexe C montre cela. Dans le monde cisco il faudrait passer par des `class-map type inspect ....` ou encore, pour des cas simples, utiliser le mot-celf `established` dans nos access-listes et si elles sont appliquées en `in` de nos interfaces.

3.4 IPv6

Bon, c'est bien gentil tout ça, mais on n'a configuré notre firewall que pour le protocol IPv4 (en principe obsolète depuis quelques dizaines d'années)... mais il y en a d'autre des protocoles de routage !

Refaites tout pour IPv6 !

(Non, je rigole, pas pour le TP. Mais dans la vraie vie...)

4 Courrier électronique

La section 4.1 décrit le fonctionnement du courrier électronique. Il convient de la lire (et de la comprendre) avant de se lancer dans les expérimentations proposées en 4.2.

4.1 Principe de fonctionnement du protocole SMTP

Le courrier électronique a été conçu pour offrir le même niveau de service et d'usage que le courrier papier. Tout le monde peut envoyer et recevoir des lettres à destination et de la part du monde entier. La poste ne contrôle pas si les utilisateurs sont autorisés ou non, ni si le contenu du courrier est légitime ou non. Par exemple, n'importe qui peut passer dans votre rue et glisser dans votre boîte aux lettres un courrier similaire à celui de votre banque (papier à entête, logo, registre de langage, etc.). Le système de courrier électronique a le même cahier des charges : assurer *l'acheminement* de messages, aussi les questions de légitimité des contenus, expéditeurs, confidentialités, etc. relèvent d'un autre niveau et d'autres outils (voir section **??**).

Autre propriété, le courrier électronique est un système décentralisé (contrairement aux systèmes de messagerie captifs proposés par les GAFAM, et supposément "plus intuitifs"). Comme avec le courrier papier, personne n'a besoin de *s'enregistrer* (auprès du service de la Poste de son État ou d'un organisme quelconque) pour envoyer un courrier, ni même en recevoir. (Le postier a tout de même besoin de voir figurer un nom de ville et de rue qu'il saura trouver ; avec le courrier électronique, il faut un serveur SMTP sur une adresse IP routable, voir dans le DNS.)

Le protocole SMTP (*Simple Mail Tranfert Protocol*) est utilisé pour l'envoi d'email entre un client et un serveur et pour l'échange d'email entre deux serveurs. Pour consulter sa boîte mail, on utilise généralement le protocole IMAP (*Internet Message Access Protocol*).

Le contenu d'un email comporte deux parties : les entêtes et le corps. Ces deux parties sont séparées par une ligne vierge. Comme sur un papier à entête, l'entête du courrier électronique contient le sujet du email, son expéditeur, son destinataire, une date, et d'autres informations comme la liste des serveurs par lesquels le email est passé, etc.

Ci-dessous, un dialogue typique entre un client email et un serveur SMTP. L'échange entre le client et le serveur se fait de manière textuelle. (Dans cet exemple, les lignes envoyées par le client sont affichées en italique, et les réponses du serveur en lettres droites.) Un être humain peut ainsi dialoguer avec un serveur sans logiciel client spécifique (même si c'est tout de même plus confortable). Notons que l'on a à faire ici à un serveur SMTP un peu verbeux, les implémentations plus modernes sont moins avenantes pour un être humain.

```
=== Connected to serveur.domain.tld
<-  220 serveur.domain.tld ESMTP Sendmail
 -> EHLO client.domain.tld
<-  250-serveur.domain.tld Hello client.domain.tld, pleased to meet you
<-  250-ENHANCEDSTATUSCODES
<-  250-PIPELINING
<-  250-EXPN
<-  250-VERB
<-  250-8BITMIME
```

```
<-  250-SIZE
<-  250-DSN
<-  250-ETRN
<-  250-AUTH DIGEST-MD5 CRAM-MD5
<-  250-DELIVERBY
<-  250 HELP
 -> MAIL FROM: user1@domain.tld
<-  250 2.1.0 user1@domain.tld... Sender ok
 -> RCPT TO: user2@domain.tld
<-  250 2.1.5 user2@domain.tld... Recipient ok
 -> DATA
<-  354 Enter mail, end with "." on a line by itself.
 -> From: user1@un-endroit.fr
 -> To: user2@ailleur.eu
 -> Subject: Sujet du message
 ->
 -> Corps du message...
 -> .
<-  250 2.0.0 k1QGh9xB020668 Message accepted for delivery
 -> QUIT
=== 221 2.0.0 serveur.domain.tld closing connection
```

Dans cet échange vous observerez deux grandes parties : tout d'abord des informations portées par *l'enveloppe* du message, puis ; après le mot-clef *DATA* le courrier lui-même (et donc "à l'intérieur" de l'enveloppe), avec son entête et le corps du message.

Le récepteur de l'email (celui dont l'adresse a été donnée via la commande `RCPT TO:` sur l'enveloppe) recevra le texte qui a été passé via la commande `DATA`. L'adresse passée à la commande `MAIL FROM:` n'est utilisé qu'en cas d'erreur lors de la livraison du message (un email d'erreur sera envoyé à cette adresse). L'adresse d'expédition contenue dans les entêtes de l'email (`From:`) peut être complètement différente et le serveur d'email ne fait, à priori, aucune vérification dessus. Or c'est cette adresse, et non celle donnée à la commande `MAIL FROM:` qui sera affichée par le logiciel de courrier du récepteur. Il est donc assez facile de tromper des utilisateurs.

De la même manière, on a deux champs pour indiquer l'adresse du destinataire... L'adresse passée à la commande `RCPT TO:` est la boîte email qui recevra le message. Par contre, le champ `To:` est l'adresse du destinataire qui sera affichée dans le lecteur d'email...

4.1.1 Acheminement du courrier

Considérons un utilisateur ordinaire, qui utilise un logiciel de messagerie électronique (MUA - *Mail User Agent*) qu'il a pris soin de configurer avec les indications techniques fournies par l'établissement qui lui a attribué une adresse électronique (typiquement l'adresse du serveur IMAP pour lire les emails, l'adresse du serveur SMTP pour envoyer des emails (ce n'est pas forcément le même), etc.)

Pour envoyer un email, le logiciel en question (MUA) se connecte donc au serveur SMTP, et comme dans l'exemple précédent, utilise les commandes `EHLO`, `RCPT TO:`, `MAIL FROM:`, `DATA`, etc. Le serveur SMTP va stocker tout ça dans sa file d'attente de traitement des emails en partance, puis le moment venu va dépiler cette file d'attente pour traiter les email. Il va alors faire attention à l'adresse du destinataire indiquée dans le `RCPT TO:`. La chaîne de caractère après le `@` est le domaine du destinataire. Notre serveur SMTP a besoin de connaître le nom du serveur SMTP qui gère ce domaine. Pour la connaître, il fait une requête DNS de type MX (*Mail eXchange*).

Par exemple, essayez la commande `host -t MX imt-atlantique.net`

Ensuite, notre serveur SMTP se connecte au serveur SMTP du destinataire, et tente de lui relayer le courrier en question. Le serveur destinataire va (éventuellement) l'accepter et le délivrer dans la boîte email de l'utilisateur destinataire.

À priori il n'y a pas besoin de passer par d'autre serveur entre l'expéditeur (From) et le destinataire (To). Par contre, un domaine peut avoir plusieurs serveurs SMTP en redondance, ou même avoir plusieurs équipements en série par lesquels transitent les emails (par exemple pour détecter les spams ou fichiers attachés douteux). Un domaine peut éventuellement confier la gestion de son email à un prestataire trier. Cependant, tout ceci est transparent pour l'expéditeur qui se contente d'utiliser l'adresse renvoyée par la requête DNS.

4.1.2 Sécurisation

À cause d'un certain nombre d'abus sur Internet, les serveurs d'emails sont de plus en plus protégés. D'autant plus que, contrairement au courrier postal, c'est quasiment gratuit.

On va par exemple essayer de se protéger contre le *relai ouvert* : typiquement, si vous mettez en place un serveur d'emails c'est pour *vos* utilisateurs. Par conséquent, vos utilisateurs (c.à.d. dans le ou les domaines que vous gérez) sont soit expéditeurs soit destinataires des emails qui passent par votre serveur. On peut ainsi détecter et filtrer les tentatives de *relay* : quelqu'un de l'extérieur qui adresse des emails à l'extérieur en passant par vous... ça n'a à priori rien à faire sur votre serveur, c'est filtré et cela lève des alarmes auprès des administrateurs du serveur. Tout serveur de mail qui relaierait des mails en provenance de tout l'Internet serait rapidement exploité par des spammeurs et se retrouverait très rapidement inscrit dans les listes noires anti-spam. (Ces listes noires sont parfois employées comme mesure de protection, avec beaucoup d'effets de bord négatifs, mais comme il y a un certain business...) Cette protection contre le relais est extrêmement importante (mais largement insuffisante pour lutter contre les spams ou le fishing).

On rajoute souvent une couche de chiffrement TLS par-dessus les protocoles réseau SMTP ou IMAP (soit directement à la connexion, soit en "escaladant" une connexion traditionnelle en clair vers du TLS). Cela permet d'éviter les écoutes passives d'une manière générale, et lorsqu'un utilisateur se connecte à son serveur, ses identifiants sont alors protégés par ce canal sécurisé. Cela permet aussi au client d'authentifier le serveur, et lorsque les emails sont relayés entre serveurs, TLS permet l'authentification réciproque.

On peut également configurer les firewalls de façon à ce que les postes de l'établissement ne puissent accéder qu'au serveur email de l'établissement et pas à des serveurs externes. Ceci évite qu'une machine de l'établissement contaminée par un virus ne se rendre coupable de spam et arrose le reste du monde... Car l'établissement porte alors la responsabilité d'une telle attaque.

D'autres vérifications sont également couramment pratiquées par les serveurs de mail : vérification de l'existence du domaine de l'expéditeur, etc. Enfin, des techniques plus évoluées sont en cours de discussion/standardisation/déploiement tels que SPF [10] (*Sender Policy Framework*), DKIM [11] (*DomainKeys Identified Mail*) qui connaît un certain succès, etc.

4.2 Faire du faux courrier électronique ?

On pourrait expérimenter le fonctionnement de SMTP pour comprendre comment procèdent les spammeurs. Notez que la charte d'usage du réseau à IMT Atlantique stipule que c'est interdit. Notez également que la loi dite *LOPSI2*, article 226-4-1, punit de 1 an d'emprisonnement et de 15000€ d'amende le délit *d'usurpation d'identité sur Internet* (c'est du pénal...).

Par conséquent, si vous testez l'envoi de faux courriel, faites-le sur votre propre adresse électronique, soit sur celle d'une personne pleinement consentante... [12]

Constatez que le protocole SMTP est très simple comme l'indique le S du nom. Le serveur attend des connexions TCP sur un numéro de port standard (25). Le protocole est textuel ; un humain peut taper les commandes au clavier et jouer le rôle du client à la main. Il est donc très simple d'envoyer du courrier électronique avec un simple : `telnet` *`un-serveur-de-mail`* `smtp`
et de dialoguer directement en SMTP [13]. (Notez que ce n'est pas beaucoup plus compliqué de causer directement en IMAP, et c'est franchement trivial en POP, mais ça a moins d'intérêt pour faire des virus qui génèrent du spam.)

Est-ce gênant ? Oui et non : le courrier standard n'offre pas plus de garanties et on peut faire de fausses lettres papier ou des lettres anonymes encore plus facilement en postant discrètement dans une boîte aux lettres.

Il existe des mécanismes d'authentification et de chiffrement des échanges indépendamment du transport (voir § **??**).

4.2.1 Expérimentations

Pour ces essais, on vous propose d'utiliser un serveur SMTP spécifique : `smtp.tp-reseaux.enstb.org`. Il s'agit d'un serveur interne à notre salle de TP [14], qui ne procède pas à beaucoup de filtrage ; il est donc plus "pédagogique". [15] Ce n'est pas une raison pour ne pas l'utiliser de manière sérieuse et responsable !

Sur ce serveur d'emails a été créée une série de comptes utilisateurs. Chacun des 9 PCs de la salle dispose de trois comptes Alice Bob Carol : `alice1` `bob1` `carol1` `alice2` `bob2` `carol2` `alice3` `bob3` `carol3` `…@tp-reseaux.enstb.org`

10. `http://www.openspf.org/`
11. `http://www.opendkim.org/`
12. Une autre option est d'utiliser un répondeur de courrier automatique `http://www.bortzmeyer.org/repondeurs-courrier-test.html`
13. `HELP` permet d'avoir le mode d'emploi en ligne. Malheureusement cette commande est souvent désactivée sur les serveurs modernes.
14. Pour les curieux, c'est `postfix` pour la partie SMTP, avec `dovecot` pour la partie IMAP, les deux étant interconnectés en LMTP, les comptes utilisateurs étant gérés du côté dovecot.
15. Gérer son propre serveur d'email n'est pas forcément compliqué, mais c'est vrai que ça prend du temps. `https://www.bortzmeyer.org/mon-serveur-messagerie.html`

Ouvrez le maileur *Thunderbird* sur votre PC, normalement vous avez accès aux trois comptes emails pré configurés. [16] Vous pouvez faire quelques échanges d'emails entre ces différents comptes pour tester ce logiciel.

Lorsque vous recevez un mail (un faux ou un prétendu vrai), prenez la peine de regarder les entêtes en détail ! Dans les menus de Thunderbird : `Affichage` / `entêtes` / `Complet`. [17] Prenez le temps de lire chaque ligne, et d'essayer de deviner ce que cela peut vouloir dire (en informatique, chaque détail compte).

Pour comprendre les choses, il vaut mieux essayer à la main. Nous pourrions ouvrir une socket TCP avec `telnet` comme évoqué précédemment et taper les commandes SMTP à la main, mais nous allons utiliser un outil en ligne de commande : `swaks`. Cet outil est dédié au test et à la maintenance de serveur SMTP ; c'est justement ce que nous faisons. Il gère les petits problèmes bas niveau de mise en forme des commandes SMTP, et nous permet de contrôler finement les choses et de tenter des manipulations exotiques via un grand nombre d'options. (Il y a d'autres outils (comme `sendmail` `msmtp` …), mais celui-ci à ma préférence. Si d'autres vous paraissent "mieux", n'hésitez pas.) Listez la documentation (`man swaks`). La documentation est très riche, très pédagogique. Je vous suggère notamment la lecture de la section *TERMS AND CONVENTIONS*, et plus précisément la définition de *Envelope DATA Headers Body*.

Ainsi, fabriquer un email en ligne de commande donnerait quelque chose comme :

```
swaks --server smtp.tp-reseaux.enstb.org
      --from alicia22@tp-reseaux.enstb.org
      --to caroline33@tp-reseaux.enstb.org
      --h-From: alicia22@tp-reseaux.enstb.org
      --h-To: caroline33@tp-reseaux.enstb.org
      --h-Subject: "Hello"
      --body "Test"
```

Bien évidemment, remplacez les différents champs par des informations adaptées pour vos tests ; ou alors, admirez les messages d'erreur. L'outil vous affiche dans votre terminal le détail de ses échanges avec le serveur SMTP. Vous observez déjà à ce niveau l'utilisation des adresses d'expéditeur et destinataires, au niveau de l'enveloppe d'une part, et au niveau des entêtes du courrier d'autre part (comme avec du courrier papier en fait).

Dans un premier temps, essayez un email "normal", sans tenter de choses exotiques. Ensuite, introduisez des variations entre l'enveloppe et l'entête.

Soyez attentif au résultat sur le message reçu. Regardez les différents champs des entêtes, tels qu'ils apparaissent au destinataire. Mais l'expéditeur peut lui-même en renseigner au moment de l'envoi, et ce de manière purement déclarative. (Regardez la documentation pour voir comment faire.) Repérez par exemple l'impact du nom de machine annoncé lors du EHLO (et donc, relisez le `man swaks`, pour trouver l'option qui va bien).

4.3 Conclusion

Ne pas croire tous les messages que vous recevez, même de personnes connues...

16. Note : vous n'en n'avez pas besoin pour les manipulations dans ce TP, mais les mots de passe sont de la forme `"alice1secret"` `"bob1secret"`….

17. Dans *Zimbra*, faites un clic droit sur le mail et choisissez *Montrer l'original*.

Annexe A

Quelques commandes de manipulation des routeurs

Ce travail utilise des routeurs Cisco 1841. Chaque routeur est équipé d'un certain nombre de ports d'entrées-sorties. Évoquons rapidement le port `console` pour l'administrer via un terminal RS232[18], et le port `aux` pour brancher des équipements auxiliaires. Intéressons nous plus spécifiquement aux ports réseau. Les routeurs disposent de deux ports Ethernet numérotés 0/0 et 0/1. Ils sont équipés également de deux slots pour insérer des cartes d'extension. Ces slots portent les numéros 0 et 1. Dans chaque routeur, une carte d'extension est installée sur le slot 0 et comporte deux ports pour la communication via une ligne série. Ils sont numérotés 0/0/0 et 0/0/1.

La documentation complète de configuration de ces routeurs est disponible sur Internet [**?**]. Dans le cadre de ce travail, nous nous limitons à quelques commandes qui sont décrites dans ce chapitre. Pour une description plus complète des commandes, voir [**?**].

Les routeurs fonctionnent dans trois modes différents : les modes *exec*, *exec privilégié* et *global configuration*. Le mode *exec* permet d'exécuter quelques commandes de base, mais sans modifier la configuration du routeur. Le mode *exec privilégié* permet de modifier certains paramètres du routeur et d'accéder à des commandes complémentaires. Le mode *global configuration* permet lui de modifier complètement la configuration du routeur. On peut voir le mode `exec` comme étant destiné à l'utilisateur normal tandis que les modes `exec privilégié` et `global configuration` sont destinés au gestionnaire du routeur.

A.1 Mode exec

Ce mode permet essentiellement de visualiser l'état des routeurs et d'exécuter quelques commandes simples. C'est le mode par défaut dans lequel on se trouve après s'être connecté sur le routeur. À tout instant, le routeur peut indiquer les commandes utilisables en réponse à la touche `?`. Cette touche peut également être utilisée pour obtenir de l'aide sur les paramètres d'une commande.

```
Exec commands:
 <1-99>          Session number to resume
 clear           Reset functions
 disable         Turn off privileged commands
 disconnect      Disconnect an existing network connection
 enable          Turn on privileged commands
 exit            Exit from the EXEC
 lock            Lock the terminal
 login           Log in as a particular user
 logout          Exit from the EXEC
 name-connection Name an existing network connection
 ping            Send echo messages
 resume          Resume an active network connection
 set             Set system parameter (not config)
 show            Show running system information
 systat          Display information about terminal lines
 terminal        Set terminal line parameters
 traceroute      Trace route to destination
 where           List active connections
 access-enable   Create a temporary Access-List entry
 access-profile  Apply user-profile to interface
 connect         Open a terminal connection
 help            Description of the interactive help system
 mls             exec mls router commands
 mrinfo          Request neighbor and version information from a multicast
                 router
 mstat           Show statistics after multiple multicast traceroutes
 mtrace          Trace reverse multicast path from destination to source
 pad             Open a X.29 PAD connection
 ppp             Start IETF Point-to-Point Protocol (PPP)
 rlogin          Open an rlogin connection
```

18. Un PC doté du logiciel `minicom` et relié via son port série au port console joue ce rôle de terminal d'administration.

```
 slip           Start Serial-line IP (SLIP)
 telnet         Open a telnet connection
 tunnel         Open a tunnel connection
 udptn          Open an udptn connection
 x28            Become an X.28 PAD
 x3             Set X.3 parameters on PAD
```

La liste ci-dessus reprend les commandes disponibles en mode `exec`. Les plus intéressantes dans le cadre de ce travail sont :

help : aide en ligne.

exit : quitte la session avec le routeur.

enable : permet de passer en mode `exec privilégié`.

ping : équivalent de la commande Unix du même nom. Faire `ping` sans paramètres pour voir la liste des paramètres.

traceroute : équivalent de la commande Unix du même nom. Faire `traceroute` sans paramètres pour voir la liste des paramètres.

show : obtenir des informations sur le routeur.

La commande `show` comprend de nombreux paramètres. Parmi ceux-ci, les plus utiles en mode `exec` sont `show running-config` qui permet de visualiser la configuration actuelle du routeur et `show interfaces` pour visualiser les différentes interfaces. D'autres paramètres sont accessibles en mode `exec privilégié`.

Exemple de configuration de l'interface FastEthernet 0/0 d'un routeur Cisco :

```
cisco1# show interface FastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
 Hardware is AmdFE, address is 0001.429d.6a01 (bia 0001.429d.6a01)
 Internet address is 192.168.2.41/24
 MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
 reliability 255/255, txload 1/255, rxload 1/255
 Encapsulation ARPA, loopback not set
 Keepalive set (10 sec)
 ARP type: ARPA, ARP Timeout 04:00:00
 Last input 00:00:00, output 00:00:00, output hang never
 Last clearing of "show interface" counters never
 Queueing strategy: fifo
 Output queue 0/40, 0 drops; input queue 0/75, 18 drops
 5 minute input rate 1000 bits/sec, 1 packets/sec
 5 minute output rate 1000 bits/sec, 1 packets/sec
 145463 packets input, 26204491 bytes, 0 no buffer
 Received 134695 broadcasts, 0 runts, 0 giants, 0 throttles
 1 input errors, 1 CRC, 1 frame, 0 overrun, 0 ignored
 0 input packets with dribble condition detected
 168756 packets output, 16028646 bytes, 0 underruns
 28 output errors, 2 collisions, 1 interface resets
 0 babbles, 0 late collision, 29 deferred
 28 lost carrier, 0 no carrier
 0 output buffer failures, 0 output buffers swapped out
```

Une autre variante intéressante de la commande show est `show ip route`. Cette commande permet de visualiser le contenu de la table de routage du routeur.

Exemple de table de routage sur un routeur Cisco :

```
cisco1# show ip route
R 192.168.4.0/24 [120/1] via 192.168.2.76, 00:00:04, FastEthernet0/1
  [120/1] via 192.168.3.201, 00:00:04, FastEthernet0/0
R 192.168.5.0/24 [120/1] via 192.168.2.100, 00:00:04, FastEthernet0/1
C 192.168.3.0/24 is directly connected, FastEthernet0/0
C 192.168.2.0/24 is directly connected, FastEthernet0/1
S 192.168.9.0 [1/0] via 192.168.3.20
```

Cette table de routage s'interprète de la façon suivante. La lettre dans la première colonne indique le type de route (R : route annoncée par le protocole RIP, C : réseau directement connecté, S : route statique). L'examen de la ligne «`R 192.168.5.0 [120/1] via 192.168.2.100, ...`» nous fournit les renseignements suivants :

— le protocole de routage utilisé pour annoncer cette route est RIP
— le réseau destination est le 192.168.5.0 (masque de 24 bits)
— la distance administrative est de 120
— la métrique de la route est 1
— le slot du routeur permettant de rejoindre ce réseau est le 192.168.2.100
— le slot sur lequel on doit envoyer les paquets est le FastEthernet 0/1

La distance administrative n'est pas utilisée dans le cadre de ce travail, nous nous contenterons de la métrique associée à la route. Celle-ci correspond au nombre de «sauts», de routeurs, qu'il faut traverser pour arriver au réseau destination.

A.2 Mode exec privilégié

Toutes les commandes possibles de l'IOS sont accessibles en mode `exec privilégié` y compris de passer en mode `global configuration`. Le passage en mode `exec privilégié` se fait avec la commande `enable`. (On en ressort avec `disable`).

Lorsque le routeur se trouve en mode `exec privilégié`, son prompt devient «`#`».

A.3 Mode global configuration

On peut entrer dans ce mode grâce à la commande `configure terminal`. Ce mode permet de changer la configuration courante du routeur. On quitte le mode `global configuration` en utilisant la commande `exit`.

Ce mode permet de réaliser les opérations principales relatives à la configuration du routeur et des protocoles de routage. Parmi les nombreuses commandes disponibles dans ce mode, vous devrez utiliser notamment les commandes de configuration des interfaces du routeur ainsi que les commandes de configuration des protocoles de routage.

A.3.1 Commandes de configuration d'interfaces

La configuration des interfaces se fait interface par interface. Pour configurer l'interface FastEthernet 0/0 d'un routeur, utilisez les commandes suivantes :

interface FastEthernet 0/0 : permet d'accéder à la configuration de l'interface FastEthernet 0/0

ip : ensemble des commandes de configuration du protocole IP sur l'interface courante. `ip ?` vous fournira la liste des commandes possibles, mais seules quelques unes d'entre elles sont utiles pour le TP. Parmi celles-ci :

ip address *adresse masque* : permet de spécifier l'adresse IP de cette interface ainsi que son masque de sous-réseau

shutdown : permet de désactiver une interface. La commande à utiliser pour activer une interface est `no shutdown`. Sur un routeur Cisco, la commande `no` *`commande paramètres`* permet de supprimer l'effet d'une commande antérieure. Par exemple, pour supprimer une adresse IP insérée précédemment, il suffit d'utiliser la commande `no ip address 192.168.1.1 255.255.255.0`

exit : permet de revenir au niveau précédent de configuration.

La configuration des interfaces série est un petit peu différente de celle des interfaces Ethernet. Pour configurer l'interface série d'un routeur Cisco, faites les commandes suivantes :

interface Serial 0/0/0 : permet d'accéder à la configuration de l'interface `Serial 0/0`

encapsulation : ensemble de commandes permettant de spécifier le type de protocole utilisé sur une interface. Dans le cadre du TP, vous utiliserez le protocole HDLC sur l'interface série.

clock rate : permet de configurer la fréquence d'envoie de données sur une ligne série. Cette commande doit donc être exécutée coté DCE (*Data Circuit Equipment*, équipement connecté sur la ligne)

Ces commandes étant effectuées, il faut ensuite assigner une adresse IP et activer l'interface. Ceci est faite en utilisant les mêmes commandes que pour la configuration d'une interface Ethernet.

A.3.2 Commandes de configuration du routage

Après avoir configuré toutes les interfaces du routeur, il est possible de manipuler notamment les tables de routage. Deux types de routage sont utilisables. Le premier est le routage statique où les tables de routage sont configurées manuellement sur chaque routeur du réseau. Le second est le routage dynamique où un protocole de routage est utilisé pour distribuer les routes dans l'ensemble du réseau.

Le routage statique est supporté avec notamment les commandes `ip routing` et `ip route`. La première active les possibilités de routage. La deuxième prend quatre paramètres : le réseau destination, le masque de ce réseau, l'adresse IP du routeur passerelle et enfin la métrique associée à cette route. Par exemple, la commande

```
cisco1(config)# ip route 192.168.1.0 255.255.255.0 192.168.3.240 10
```

permet d'insérer une route pour le réseau `192.168.1.0` avec un masque de 24 bits et en utilisant le routeur 192.168.3.240 et avec une métrique de 10. Pour supprimer cette route de la table de routage, il suffit d'utiliser la commande

```
cisco1(config)# no ip route 192.168.1.0 255.255.255.0 192.168.3.240 10
```

L'utilisation des protocoles de routage dynamique se fait en activant le protocole de routage par l'intermédiaire de la commande `router`. Par exemple, `router rip` active la version 1 du protocole RIP sur le routeur. Une fois le protocole de routage activé, le routeur est prêt à accepter les messages RIP venant d'autres routeurs du réseau. Un routeur n'annoncera des routes avec le protocole RIP que si chaque route à annoncer est explicitement spécifiée. Cela se fait avec la commande `network`. Par exemple, les commandes ci-dessous permettent au routeur Cisco1 d'annoncer à l'ensemble du réseau qu'il parvient à joindre le réseau 192.168.0.0.

```
cisco1(config-if)# ip address 192.168.0.1 255.255.255.0
cisco1(config-if)# exit
cisco1(config)# router rip
cisco1(config-router)# network 192.168.0.0
```

Outre l'annonce de routes, il est possible de configurer certains paramètres opérationnels du protocole RIP. Par défaut, un routeur RIP envoie ses vecteurs de distance toutes les trente secondes. Ce délai peut être modifié grâce à la commande `timers basic`.

A.4 tips & tricks

Voici quelques combinaisons de touches (non documentées) Cisco.

A.4.1 Sur la ligne de commande

- `Ctrl-A` : déplace le curseur en début de ligne
- `Ctrl-E` (end) : déplace le curseur en fin de ligne
- `Ctrl-B` (backward) : déplace le curseur de un caractère à gauche
- `Ctrl-F` (forward) : déplace le curseur de un caractère à droite
- `Esc-B` : recule d'un mot
- `Esc-F` : avance d'un mot

A.4.2 Pour stopper un ping

Lorsque l'on tape un mot non reconnu comme une commande, Cisco l'interprète comme un nom de machine et fait un ping 30 fois dessus... ça peut être long.

Pour l'interrompre : `Ctrl-Shift-6` deux fois.

Suivant les versioons d'IOS, c'est parfois : `Ctrl-Shift-6` puis `x`

Annexe B

Firewall Cisco

Auteur : *Ronan Keryell*

B.1 Filtrage par adresse

— Empêcher des paquets avec source interne falsifiée de rentrer
— Altruisme : empêcher des paquets de sortir avec une adresse non locale
— Faire confiance à certaines machines extérieures. ⚠ adresses sources falsifiées

Exemple sur Cisco :

```
interface Ethernet0
 ip address 194.214.157.2 255.255.255.0
 ip access-group 100 in
 media-type 10BaseT
! Effacer l'access-list avant de commencer
no access-list 100
! Les adresses locales
access-list 100 deny ip 192.54.148.0 0.0.0.255 any
access-list 100 deny ip 192.54.172.0 0.0.0.255 any
access-list 100 deny ip 192.54.173.0 0.0.0.255 any
access-list 100 deny ip 127.0.0.0 0.255.255.255 any
! Adresses privées du RFC 1597
access-list 100 deny ip 10.0.0.0 0.255.255.255 any
access-list 100 deny ip 172.16.0.0 0.15.255.255 any
access-list 100 deny ip 192.168.0.0 0.0.255.255 any
! École des Mines, site de Paris
access-list 100 permit ip 192.54.165.0 0.0.0.255 any
access-list 100 permit ip 194.214.158.0 0.0.0.255 any
```

B.2 Filtrage par port

— Filtre les services associés à des ports UDP ou TCP
— Interdiction de certains services entrants, mais autorise en sortie
— Empêche certains services de passer directement sans mandataire du bastion (adresses sources falsifiées)
— Note : pour afficher la liste des correspondances port-service connues par votre cisco : `show ip port-map`.

Exemple sur Cisco :

```
no service udp-small-servers
no service tcp-small-servers
!pour le cri (R.Keryell) roazhon, chailly - dmi.ens.fr, trefle.ens.fr
access-list 100 permit tcp host 129.199.96.17 host 192.54.172.242 eq 6000
access-list 100 permit tcp host 129.199.96.17 host 192.54.172.200 eq 6000
access-list 100 permit tcp host 129.199.96.11 host 192.54.172.242 eq 6000
access-list 100 permit tcp host 129.199.96.11 host 192.54.172.200 eq 6000
access-list 100 deny udp any any eq echo
access-list 100 deny tcp any any eq echo
access-list 100 deny tcp any any eq 11
access-list 100 deny tcp any any eq 15
access-list 100 deny udp any any eq bootps
access-list 100 deny udp any any eq tftp
access-list 100 deny tcp any any eq 87
access-list 100 deny tcp any any eq 95
access-list 100 deny tcp any any eq sunrpc
access-list 100 deny udp any any eq sunrpc
access-list 100 deny tcp any any eq 144
access-list 100 deny udp any any eq snmp
access-list 100 deny udp any any eq xdmcp
access-list 100 deny tcp any any eq exec
```

```
access-list 100 deny udp any any eq biff
access-list 100 deny udp any any eq who
access-list 100 deny tcp any any eq cmd
access-list 100 deny udp any any eq syslog
access-list 100 deny tcp any any eq lpd
access-list 100 deny udp any any eq rip
access-list 100 deny tcp any any eq 2000
access-list 100 deny tcp any any eq 2001
access-list 100 deny tcp any any eq 2002
access-list 100 deny tcp any any eq 2003
access-list 100 deny udp any any eq 2049
access-list 100 deny tcp any any eq 2049
access-list 100 deny tcp any any eq 6000
access-list 100 deny tcp any any eq 6001
access-list 100 deny tcp any any eq 6002
access-list 100 deny tcp any any eq 6003
access-list 100 permit ip any any
```

Annexe C

Firewall Linux

Auteur : *Ronan Keryell*

C.1 NetFilter (IPTables)

`http://www.netfilter.org/`
— À partir de GNU/Linux 2.4
— ∃ interfaces graphiques
`http://online.securityfocus.com/infocus/1410`
`http://expansa.sns.it/knetfilter/`
— Garde un état des paquets passés (suivi de connections TCP (sens, segmentation,...), FTP, DNS, ICMP,...)
— Filtrage par caractéristiques de paquets IP, adresses MAC
— Enregistrements paramétrables (*log*)
— Traduction d'adresses (NAT) et mandataires (*proxy*) transparents
— Contrôle de fréquence de certains paquets (*scan*, dénis de service,...)
— Test sur un processus émetteur/récepteur (*uid*, *gid*,...)
— Opérations possibles dans l'espace utilisateur (`libipq`)
— Extensible

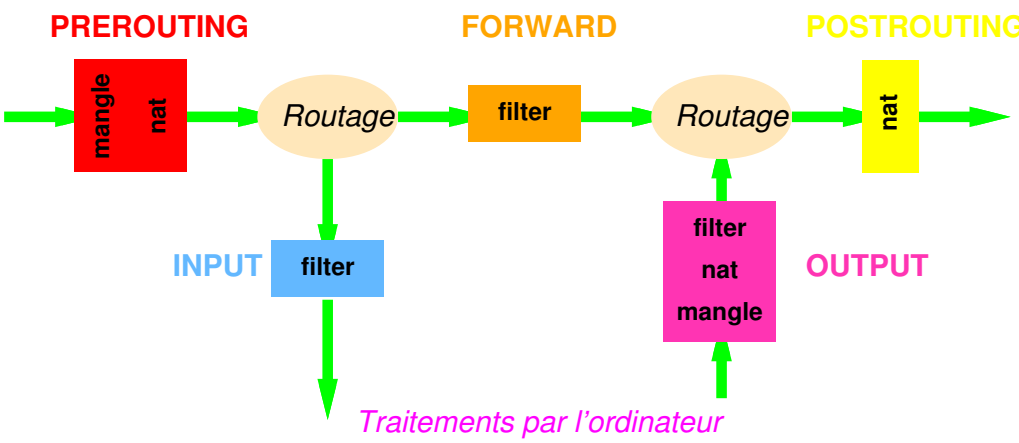
C.2 NetFilter pour quoi faire ?

Permet par exemple de :
— Faire un pare-feu avec ou sans état
— Cacher un réseau derrière une seule machine (NAT) si pas assez d'adresses publiques
— Réaliser des mandataires transparents évitant d'avoir à reconfigurer les programmes utilisateurs (HTTP, H.323, FTP,...)
— Marquer des paquets avec différentes qualités de service (QoS pour `tc` et `iproute2`,...)
— Manipuler à foison les paquets (ToS,...)

C.3 Concepts de NetFilter

`http://www.netfilter.org/documentation/tutorials/blueflux/`
— 2 parties :
  — NetFilter dans le noyau qui permet d'appeler des fonctions (*hooks*) avec les paquets
  — Infrastructure de sélection des paquets `iptables` pour interagir avec au niveau noyau et utilisateur
— Les paquets passent par des « chaînes » : flot de données classique (sorte de sous-programme)
— Dans chaque chaîne on applique les règles d'une ou plusieurs tables dans l'ordre jusqu'à trouver une règle vérifiée ou la règle par défaut (sorte d'instructions)

C.4 Routage standard à travers les chaînes



C.5 Chaînes dans NetFilter

Les paquets passent par
— `INPUT` : avant d'être utilisés par l'ordinateur local
— `OUTPUT` : après être générés par l'ordinateur local
— `FORWARD` : lors du transit par l'ordinateur local (routage)
— `PREROUTING` : dès réception sur une interface pour un premier paquet (ouverture de connexion)
— `POSTROUTING` : juste avant émission sur une interface pour un premier paquet (ouverture de connexion)

C.6 Tables dans NetFilter

— 3 tables de base pour mieux structurer les tâches
— `filter` table par défaut
  — Dévouée au pur filtrage des paquets
  — S'applique dans les chaînes `INPUT`, `FORWARD` et `OUTPUT`
— `nat`
  — Orientée traduction d'adresse
  — S'applique dans les chaînes `PREROUTING`, `OUTPUT` et `POSTROUTING`
— `mangle`
  — Pour modifications de paquets spécifiques
  — Œuvre du côté de `PREROUTING`, `OUTPUT`

C.7 Cibles dans une règle NetFilter

— Définit la cible à prendre dans la table courante si une règle est vérifiée
— Peut être
  — Appel à une chaîne définie par l'utilisateur
  — Cibles spéciales
    — `ACCEPT` : le paquet continue
    — `DROP` : le paquet part à la poubelle
    — `QUEUE` : le paquet continue sa vie vers l'espace utilisateur
    — `RETURN` : revient après la règle appelante (≈ sous-routine)
  — Cibles étendues pour marquer les paquets, faire du log, de la traduction d'adresse, renvoyer des paquets de réponse,...

```
# Met en place du masquage sur le paquet partant via ppp0 :
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
# Cache notre réseau local derrière notre réseau public :
iptables -t nat -A POSTROUTING -o eth0 -j SNAT \
         --to-source 193.50.97.144-193.50.97.147
# Fait du mandataire transparent pour le traffic WWW :
iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 \
         -j DNAT --to-destination 193.50.97.250:8080
```

— On peut définir une règle par défaut par chaîne

C.8 Utilisation avec iptables

— Commande permettant de manipuler le système de filtrage à partir d'un script
— Administration des chaînes
  — `-L`, `--list` : affiche les règles d'une chaîne
  — `-N`, `--new-chain` : créer une nouvelle chaîne
  — `-F, --flush` : vide toutes les règles d'une chaîne
  — `-X`, `--delete-chain` : détruire une chaîne vide
  — `-P`, `--policy` : définit le comportement par défaut (cible) d'une chaîne
  — `-Z`, –zero : met à 0 les compteurs associés à toutes les chaînes
— Administration des règles
  — `-A`, `--append` : rajoute une ou plusieurs règles à la fin d'une chaîne
  — `-D`, `--delete` : supprime une ou plusieurs règles dans une chaîne
  — `-R`, `--replace` : remplace une règle dans une chaîne
  — `-I`, `--insert` : insère une ou plusieurs règles dans une chaîne
— Administration des tables
  — `-t` précise la table à considérer

C.9 Exemple de pare-feu avec iptables

Exemple de pare-feu protégeant un réseau privé derrière une adresse publique

```
#!/bin/sh
# L'accès au monde extérieur :
INET_IP="194.236.50.155"
INET_IFACE="eth0"

# Le réseau local (privé RFC 1918)
LAN_IP="192.168.0.2"
LAN_IP_RANGE="192.168.0.0/16"
LAN_BCAST_ADRESS="192.168.255.255"
LAN_IFACE="eth1"

# Mon ego :
LO_IFACE="lo"
LO_IP="127.0.0.1"

IPTABLES="/usr/sbin/iptables"

# Par défaut (paranoïaque) : tout à la poubelle !
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP

# Crée une chaîne pour les mauvais paquets TCP :
$IPTABLES -N bad_tcp_packets

# Crée des chaînes spécifiques pour faire traverser ICMP, TCP et UDP :
$IPTABLES -N allowed
$IPTABLES -N icmp_packets
$IPTABLES -N tcp_packets
$IPTABLES -N udpincoming_packets

# Règles pour les mauvais paquets TCP :
$IPTABLES -A bad_tcp_packets -p tcp ! --syn -m state --state NEW \
 -j LOG --log-prefix "New not syn:"
$IPTABLES -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j DROP

# Règles pour les bons paquets TCP :
$IPTABLES -A allowed -p TCP --syn -j ACCEPT
$IPTABLES -A allowed -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A allowed -p TCP -j DROP

# Règles pour TCP :
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 21 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 22 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 80 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 --dport 113 -j allowed

# Règles pour UDP :
$IPTABLES -A udpincoming_packets -p UDP -s 0/0 \
 --destination-port 53 -j ACCEPT

# Règles pour ICMP :
$IPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j ACCEPT
$IPTABLES -A icmp_packets -p ICMP -s 0/0 --icmp-type 11 -j ACCEPT

# Teste les mauvais paquets TCP qui nous sont destinés :
$IPTABLES -A INPUT -p tcp -j bad_tcp_packets
```

```
# Accepte les connexions internes :
$IPTABLES -A INPUT -p ALL -i $LAN_IFACE -s $LAN_IP_RANGE -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LO_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LAN_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $INET_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LAN_IFACE -d $LAN_BCAST_ADRESS -j ACCEPT

# Filtre les connexions depuis l'extérieur :
$IPTABLES -A INPUT -p ALL -d $INET_IP -m state \
    --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -p TCP -i $INET_IFACE -j tcp_packets
$IPTABLES -A INPUT -p UDP -i $INET_IFACE -j udpincoming_packets
$IPTABLES -A INPUT -p ICMP -i $INET_IFACE -j icmp_packets

# Enregistre les paquets bizarroïdes :
$IPTABLES -A INPUT -m limit --limit 3/minute --limit-burst 3 \
 -j LOG --log-level DEBUG --log-prefix "IPT INPUT packet died: "

# Filtre sur la fonction de routage :
$IPTABLES -A FORWARD -p tcp -j bad_tcp_packets

$IPTABLES -A FORWARD -i $LAN_IFACE -j ACCEPT
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Enregistre les paquets bizarroïdes :
$IPTABLES -A FORWARD -m limit --limit 3/minute --limit-burst 3 \
 -j LOG --log-level DEBUG --log-prefix "IPT FORWARD packet died: "

# On est gentil avec les autres :
$IPTABLES -A OUTPUT -p tcp -j bad_tcp_packets
$IPTABLES -A OUTPUT -p ALL -s $LO_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $LAN_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $INET_IP -j ACCEPT

# Enregistre une action locale malicieuse :
$IPTABLES -A OUTPUT -m limit --limit 3/minute --limit-burst 3 \
 -j LOG --log-level DEBUG --log-prefix "IPT OUTPUT packet died: "

# Fait de la traduction d'adresses vers l'extérieur :
$IPTABLES -t nat -A POSTROUTING -o $INET_IFACE \
 -j SNAT --to-source $INET_IP
```

C.10 Exemple pour supprimer le pare-feu

Remettre à ”zéro” le firewall de Linux signifie supprimer toutes les règles de toutes les tables, et choisir comme politique par défaut d'accepter tous les paquets. Donc, un simple `iptables -f` ne suffit pas...

Voici comment pousser une config à ”zéro” dans iptables :

```
iptables-restore << EOF
*mangle
:PREROUTING ACCEPT
:INPUT ACCEPT
:FORWARD ACCEPT
:OUTPUT ACCEPT
:POSTROUTING ACCEPT
COMMIT
*nat
:PREROUTING ACCEPT
:POSTROUTING ACCEPT
:OUTPUT ACCEPT
COMMIT
```

```
*filter
:INPUT ACCEPT
:FORWARD ACCEPT
:OUTPUT ACCEPT
COMMIT
```