

Sécuriser son poste Linux

Olivier Hoarau (olivier.hoarau@funix.org)

V1.3 du 6.01.04

1	Historique du document.....	3
2	Préambule.....	3
3	Sécuriser son poste linux.....	3
3.1	Présentation.....	3
3.2	Attaques possibles.....	4
3.2.1	Présentation.....	4
3.2.2	Compte utilisateur.....	4
3.2.3	Les outils "r" (rsh, rlogin, ...).	4
3.2.4	FTP.....	5
3.2.5	Outils de stats.....	5
3.2.6	NFS.....	5
3.2.7	Serveur web.....	5
3.2.8	Serveurs Mail.....	6
3.2.9	Autres.....	6
3.3	Améliorer la sécurité.....	6
3.3.1	Installation de la distribution.....	6
3.3.2	Eliminer les services inutiles.....	7
3.3.3	Eliminer les scripts de lancement inutiles.....	10
3.3.4	Sécuriser /etc/passwd.....	12
3.3.5	Sécuriser FTP.....	12
3.3.6	Sécuriser Telnet.....	13
3.3.7	Les TCP Wrappers.....	14
3.3.8	Root et utilisateurs privilégiés.....	15
3.3.9	Sécuriser les fichiers et systèmes de fichiers.....	16
4	Auditer la sécurité de son réseau.....	16
4.1	Présentation.....	16
4.2	AVERTISSEMENT.....	17
4.3	Sara.....	17
4.3.1	Présentation.....	17
4.3.2	Installation.....	17
4.3.3	Utilisation.....	19
4.4	Nmap.....	27
4.4.1	Présentation.....	27
4.4.2	Installation avec le tarball.....	27
4.4.3	Syntaxe.....	28
4.4.4	Quelques exemples.....	29

4.4.5Le front end de nmap.....	32
4.5Nessus.....	33
4.5.1Présentation.....	33
4.5.2Installation.....	33
4.5.3Utilisation.....	38
5Détecter les attaques en temps réel.....	41
5.1Présentation.....	41
5.2Tcplogd.....	41
5.2.1Présentation.....	41
5.2.2Installation.....	42
5.2.3Tests de fonctionnement.....	45
5.2.4Utilisation.....	46
5.3Portsentry.....	47
5.3.1Présentation.....	47
5.3.2Installation.....	47
5.4Configuration.....	49
5.4.1Les modes de fonctionnement.....	51
5.4.2Tests de fonctionnement.....	52
5.4.3Lancement automatique de portsentry.....	54
5.5Iplog.....	54
5.5.1Présentation.....	54
5.5.2Installation.....	55
5.5.3Configuration.....	55
5.5.4Utilisation.....	56
6"Sniffer" son réseau avec Ethereal et Snort.....	57
6.1Présentation.....	57
6.2Libpcap.....	57
6.2.1Présentation.....	57
6.2.2Installation.....	57
6.3Ethereal.....	58
6.3.1Installation.....	58
6.3.2Utilisation d'Ethereal.....	59
6.4snort.....	62
6.4.1Présentation.....	62
6.4.2Installation.....	62
6.4.3Syntaxe.....	63
6.4.4Utilisation.....	64

1 Historique du document

V1.3 06.01.04 Passage à sara 4.2.7, nmap 3.48, nessus 2.0.9, libpcap 0.8.1, ethereal 0.10.0a et snort 2.1.0

V1.2 16.03.03 Rajout d'un paragraphe sur Iplog
Changement de version (Sara 4.1.4b, Nessus 1.2.6, snort 1.9.1 ,
Ethereal 0.9.11)

V1.1 15.09.02 Rajout d'un paragraphe sur les outils ethereal et snort pour sniffer son réseau

Passage à Sara 4.0.1
Passage à nmap 3.00
Passage à nessus 1.2.5

V1.0 22.02.02 Création du document

2 Préambule

Ce document présente les moyens de sécuriser son poste Linux.

La dernière version de ce document est téléchargeable à l'URL <http://www.funix.org>. Ce document peut être reproduit et distribué librement dès lors qu'il n'est pas modifié et qu'il soit toujours fait mention de son origine et de son auteur, si vous avez l'intention de le modifier ou d'y apporter des rajouts, contactez l'auteur pour en faire profiter tout le monde.

Ce document ne peut pas être utilisé dans un but commercial sans le consentement de son auteur. Ce document vous est fourni "dans l'état" sans aucune garantie de toute sorte, l'auteur ne saurait être tenu responsable des quelconques misères qui pourraient vous arriver lors des manipulations décrites dans ce document.

3 Sécuriser son poste linux

3.1 Présentation

Le but de cette page est de présenter le moyen de sécuriser son poste linux, il n'a pour vocation de présenter une configuration sécurisée à 100% mais suffisamment protégée pour affronter le "chaos" d'internet.

3.2 Attaques possibles

3.2.1 Présentation

Avant de sécuriser votre poste, voici une liste non exhaustive des méthodes les plus classiques pour pénétrer un système, les détails pour exécuter ses attaques ne sont pas exposés, pour cela vous trouverez un tas de sites sur le net qui explique ça très bien.

3.2.2 Compte utilisateur

La première étape pour essayer de pénétrer sur un système est tout simplement de se loguer comme un simple utilisateur, pour cela il faut connaître un login, anciennement beaucoup de systèmes UNIX disposait de comptes génériques par défaut, du style "**guest**" pour invité, un hacker essaiera toujours ses comptes par défaut.

Pour éviter cela, vérifiez votre fichier **/etc/passwd** et supprimer tous vos comptes génériques utilisateurs (pas les comptes systèmes !!).

Veillez à ce qu'aucun utilisateur n'ait pas de password et encore moins un password identique au login (très courant !!).

3.2.3 Les outils "r" (rsh, rlogin, ...)

Attention à ses outils, ils utilisent un système d'authentification relativement rudimentaire et sont connues pour constituer une faille de sécurité. Qui plus est avec ces outils on peut mettre en place des relations de "confiance" entre les machines, c'est à dire qu'on pourra accéder à une machine en utilisant les r-outils sans avoir à entrer de mot de passe, par conséquent si un système est hacké, cela signifie que tous les autres tomberont aussi. Sachez aussi que les r-outils ne disposent pour la plupart d'aucun mécanisme de logging, donc aucun moyen d'avoir l'historique de leur utilisateur. Les fichiers permettant d'établir les relations de confiance sont **.rhosts** ou **/etc/hosts.equiv**.

Les outils r principaux sont:

- **rsh** ouvre un shell à distance (pour lancer une commande par exemple), si vous voulez un outil équivalent sécurisé, tournez vous vers **SSH**.
- **rlogin**, équivalent à **telnet**, si un utilisateur place un fichier **.rhosts** dans sa homedirectory contenant **obelix**, n'importe qui sur la machine **obelix** pourra se connecter sur son compte sans avoir à donner de mot de passe. Si vous possédez un **/etc/hosts.equiv** contenant un **+**, n'importe qui aura accès à votre machine.
- **rexecd**, c'est le serveur pour accepter des requêtes de **rexec**, permet de lancer des commandes à distance. Le serveur ne logue aucun échec de connexion, par conséquent vous pouvez essayer une tonne de mot de passe sans que l'administrateur de la machine visée sans rende compte.

3.2.4 FTP

Ne faites pas tourner un serveur **FTP** anonyme à moins que vous sachiez ce que vous faites, sachez que les daemons **FTP** sont connus pour présenter pas mal de problèmes de sécurité, un serveur **FTP** mal configuré peut très bien servir de passerelle à un hacker pour attaquer votre machine bien sûr mais aussi d'autres machines.

TFTP (Trivial File Transfer Protocol) est un service **FTP** simplifié, il est utilisé notamment pour le boot des terminaux X. Il ne demande aucune authentification, n'importe qui peut se connecter et lire ce qu'il veut.

Je vous conseille donc de désactiver tout ce qui tourne autour de **FTP**.

3.2.5 Outils de stats

Des outils comme **finger**, **systat**, **netstat**, **rusersd**, **rwhod**, ... servent à avoir des informations sur le système (sur les utilisateurs, stats de réseau, process qui tournent, ...). Pour la plupart ce sont des daemons qu'on peut interroger de l'extérieur, il devient alors très facile pour un hacker d'obtenir un max d'informations sur votre système, qui lui permettront de mieux cibler ses attaques.

Par exemple avec **systat** et **netstat** qui tournent sur votre système, un hacker peut visualiser vos process actifs ainsi que la configuration réseau. Des outils comme **rusersd** et **rstatd** permettent à un hacker de visualiser les gens qui sont logués à un moment donné. **Finger** permet de connaître les logins existants sur la machine.

3.2.6 NFS

Les versions précédentes de **NFS** comportaient des trous de sécurité, vous devez veiller à posséder la dernière version. Vous ne devez en aucun cas exporter vos systèmes de fichiers vers le monde entier, vous devez toujours restreindre l'accès à un nombre limité de machines et préféré toujours la lecture seule que la lecture/écriture. N'exportez pas plus que nécessaire, c'est à dire exporter **/usr/local/public** plutôt que / tout entier. Vous pouvez faire en sorte que le root de la machine distante qui monte votre répertoire puisse avoir les mêmes droits que celui de la machine locale, c'est à éviter (ce n'est pas le cas par défaut). Enfin **NFS** doit être utilisé strictement en interne sur un réseau local et en aucun cas entre des machines sur internet.

3.2.7 Serveur web

Faites très attention à vos scripts **cgi-bin**, un défaut dans l'un d'entre eux et c'est la porte ouverte à votre système. Si vous mettez un serveur web en place, supprimez donc tout ce qui se trouve sous le répertoire des **cgi-bin** et mettez y que vos scripts **CGI** dont vous soyez absolument sûrs. Ne faites pas confiance aux script **CGI** que vous récupérez sur internet.

En plus des scripts **CGI**, il y a bien d'autres failles dans un serveur **HTTP**, que ce soit au niveau d'une mauvaise configuration, des serveurs bogués, ... même les logs peuvent être un problème, s'ils sont visibles d'internet, notamment ceux contenant les erreurs (page 404), qui peuvent contenir les informations sensibles comme les login des utilisateurs.

3.2.8 Serveurs Mail

Attention aux serveurs **SMTP**, il y a eu par le passé pas mal d'attaques du à des failles dans **sendmail**, **smail** et d'autres, veillez à toujours faire tourner la dernière version et n'hésitez pas à appliquer les patchs. Désactivez tout serveur **SMTP** si vous ne vous en servez pas.

Pour les serveurs **POP/IMAP**, sachez qu'il y a eu sur ces serveurs un bogue (buffer overrun) qui faisait qu'on pouvait lancer des commandes en tant que root à distance. Faites une mise à jour, ou désactivez les. Certaines serveurs **POP** ne loguent pas les erreurs de logins, donc n'importe qui peut essayer une tonne de password sans que vous vous rendiez compte. Enfin **POP/IMAP** manipule les mots de passe en clair, on peut très bien sur le réseau intercepter une requête **POP** (par sniffing) et récupérer le login et le mot de passe. Préférez les serveurs **POP** sécurisés (APOP, ...).

3.2.9 Autres

Parmis les outils réputés potentiellement peu sûrs, même si les nouvelles versions tendent à corriger petit à petit les défauts, citons **DNS**, **Samba**, services **RPC**, **portmapper**, et j'en oublie.

3.3 Améliorer la sécurité

3.3.1 Installation de la distribution

La sécurisation du poste commence à l'install, je vous déconseille de choisir les configs standards proposés dans les distribs comme **RedHat** ou **Mandrake** avec **WorkStation** (Station de travail), **Server** (Serveur), le mieux est de choisir **Custom** (personnalisé) pour avoir un oeil sur les outils qui seront installés sur votre système. L'idée est d'installer le minimum d'outils, la règle la première si vous ne voyez pas à quoi peut servir un outil est de ne pas le choisir, il faut vous dire que de toute manière vous aurez toujours la possibilité de l'installer plus tard. En résumé moins vous avez d'outils, moins vous avez de trous potentiels de sécurité.

Au moment d'arriver au partitionnement du système, je vous conseille de créer plusieurs partitions pour bien séparer le système, des données, par exemple:

```
/      150Mo
/usr   1Go au moins
/var   400Mo
/home  50Mo par utilisateurs au moins
```

swap 2*taille de la RAM au moins
/usr/local au moins 500Mo (pour stocker vos applis)

Une fois l'installation terminée, suivant votre distrib allez sur le site correspondant (**RedHat** www.redhat.com, **Mandrake**, www.linux-mandrake.com, ...), dans la rubrique support vous devriez trouver les derniers packages corrigeant les trous de sécurité. Ces patchs sont indispensables pour sécuriser correctement un système, vous devez surveiller régulièrement leur parution. Sachez qu'il existe des mailing-listes à ce sujet qui vous préviendront automatiquement de la parution d'un nouveau patch.

Beaucoup de distributions dispose d'outils (**up2date** pour la RedHat, **urpmi** pour la Mandrake) qui permettent d'upgrader automatiquement le système.

Je vous conseille de tenir à jour un cahier d'administrateur où vous noterez chacune de vos manip systèmes, ça peut se révéler un brin écolier, voire fastidieux, mais ça peut se révéler extrêmement utile dans certains cas. Un simple fichier texte (droit 400 proprio root) avec un copier coller des commandes (et les résultats qui vont avec), et quelques annotations devrait faire l'affaire.

3.3.2 Eliminer les services inutiles

Maintenant que le système est installé et que vous avez patché les packages defectueux, passons maintenant à la sécurisation propre du système. Dans un premier temps on va désactiver tous les services inutiles qui tournent sur la machine.

Si vous utilisez inetd

On va d'abord travailler sur **inetd**, **inetd** est un "super daemon" c'est à dire qu'il permet à lui tout seul de lancer tout un tas d'autres daemons (ou services), il est configurable à partir du fichier **/etc/inetd.conf**. Par défaut il comprend un certain nombre de services activés dont vous n'avez pas forcément besoin, à l'inverse certains services désactivés peuvent vous être utiles. On décomment un service en lui mettant un # devant. Voici un exemple de fichier **inetd.conf** :

```
#
# inetd.conf  This file describes the services that will be available
#             through the INETD TCP/IP super server.  To re-configure
#             the running INETD process, edit this file, then send the
#             INETD process a SIGHUP signal.
#
# Version:    @(#) /etc/inetd.conf  3.10  05/27/93
```

```

#
# Authors:    Original taken from BSD UNIX 4.3/TAHOE.
#            Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org>
#
# Modified for Debian Linux by Ian A. Murdock <imurdock@shell.portal.com>
#
# Modified for RHS Linux by Marc Ewing <marc@redhat.com>
#
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
#
# Echo, discard, daytime, and chargen are used primarily for testing.
#
# To re-read this file after changes, just do a 'killall -HUP inetd'
#echo stream tcp  nowait root  internal
#echo dgram udp   wait  root  internal
#discard stream tcp  nowait root  internal
#discard dgram udp  wait  root  internal
#daytime stream tcp  nowait root  internal
#daytime dgram udp  wait  root  internal
#chargen stream tcp  nowait root  internal
#chargen dgram udp  wait  root  internal
#time stream tcp  nowait root  internal
#time dgram udp  wait  root  internal
#
# These are standard services.
# Attention à ces deux services ce sont deux gros trous potentiels de sécurité
# j'ai désactivé le serveur ftp car j'en ai pas l'utilité
# ftp stream tcp  nowait root  /usr/sbin/tcpd in.ftpd -l -a
telnet stream tcp  nowait root  /usr/sbin/tcpd in.telnetd
#
# Shell, login, exec, comsat and talk are BSD protocols.
# r-outils et autres à désactiver
#
#shell stream tcp  nowait root  /usr/sbin/tcpd in.rshd
#login stream tcp  nowait root  /usr/sbin/tcpd in.rlogind
#exec stream tcp  nowait root  /usr/sbin/tcpd in.rexecd
#comsat dgram udp  wait  root  /usr/sbin/tcpd in.comsat
#talk dgram udp  wait  root  /usr/sbin/tcpd in.talkd
#ntalk dgram udp  wait  root  /usr/sbin/tcpd in.ntalkd
#dtalk stream tcp  wait  nobody /usr/sbin/tcpd in.dtalkd
#
# Pop and imap mail services et al
# serveur pop3 pour réseau local activé, si poste isolé à désactiver
#pop-2 stream tcp  nowait root  /usr/sbin/tcpd ipop2d

```



```

pop-3 stream tcp nowait root /usr/sbin/tcpd ipop3d
#imap stream tcp nowait root /usr/sbin/tcpd imapd
#
# The Internet UUCP service.
#
#uucp stream tcp nowait uucp /usr/sbin/tcpd /usr/lib/uucp/uucico -l
#
# Tftp service is provided primarily for booting. Most sites
# run this only on machines acting as "boot servers." Do not uncomment
# this unless you *need* it.
#tftp dgram udp wait root /usr/sbin/tcpd in.tftpd
#bootps dgram udp wait root /usr/sbin/tcpd bootpd
#
# Finger, systat and netstat give out user information which may be
# valuable to potential "system crackers." Many sites choose to disable
# some or all of these services to improve security.
# outils de stats réseau à désactiver
#
#finger stream tcp nowait root /usr/sbin/tcpd in.fingerd
#cfinger stream tcp nowait root /usr/sbin/tcpd in.cfingerd
#systat stream tcp nowait guest /usr/sbin/tcpd /bin/ps -auwwx
#netstat stream tcp nowait guest /usr/sbin/tcpd /bin/netstat -f inet
#
# Authentication
#
# auth stream tcp nowait nobody /usr/sbin/in.identd in.identd -l -e -o
#
# linuxconf stream tcp wait root /bin/linuxconf linuxconf --http
#swat stream tcp nowait.400 root /usr/sbin/swat swat

```

Si vous utilisez xinetd

Si vous utilisez **xinetd**, pour les services inutiles se trouvant sous **/etc/xinetd.d** il suffira de rajouter le paramètre **disable=yes**, exemple avec **daytime**

service daytime

```

{
    type = INTERNAL
    id = daytime-stream
    socket_type= stream
    protocol = tcp

```

```
user = root
wait = no
disable = yes
}
```

Relancez **xinetd**

/etc/rc.d/init.d/xinetd restart

3.3.3 Eliminer les scripts de lancement inutiles

Maintenant on va aller jeter un coup d'œil dans les répertoires **/etc/rc.d/rcX.d** qui contiennent des liens vers les scripts de lancement de services, on va désactiver les services inutiles. Si vous démarrez à l'état de marche 5, voici ce que vous pourriez trouver dans le répertoire **/etc/rc.d/rc5.d**

S05apmd Utile uniquement pour les portables, sert à gérer l'autonomie d'une batterie, à virer si vous avez un poste de bureau

S09pcmcia pour activer les services liés au **PCMCIA**, à virer si pas de **PCMCIA**

S10network active les interfaces réseau (**eth0**, ...)

S11portmap Utile si vous utilisez des services **RPC** comme **NFS** ou **NIS**

S15netfs lance le service **NFS** client, à désactiver si vous ne montez jamais de file systems d'un serveur **NFS**

S16ypserv pour lancer le serveur **NIS**, à désactiver si non utilisé

S20random ce n'est pas un daemon, mais un truc qui permet de générer bazar aléatoire, je vois pas trop son utilité mais vous pouvez le laisser, il n'y a aucun risque niveau sécurité

S20rstatd service "r", à désactiver

S20rwhod idem, à désactiver

S20rusersd idem, à désactiver

S20bootparamd, idem **tftp**, sert pour les terminaux X ou autres clients "diskless", à désactiver

S30syslog permet de loguer l'activité des daemons lancés par (x)**inetd**, à conserver

S34yppasswd si vous êtes un serveur **NIS** pour pouvoir changer de mot de passe, à désactiver si non utilisé (très vulnérable)

S35dhcp daemon **DHCP** sert pour obtenir une adresse IP, sert pour les câblés entre autres, à désactiver si non nécessaire

S40atd utile pour le service **at** (similaire à **cron**), vous pouvez désactiver si vous vous en servez pas

S40crond pour lancer le daemon **cron** qui permet de programmer des tâches à lancer, à virer si vous ne l'utilisez pas

S50inet pour lancer le réseau,

S50snmpd pour lancer le daemon **SNMP**, permet de donner à des utilisateurs distants des

informations détaillées sur votre système, à désactiver

S55named pour lancer le serveur **DNS**, à désactiver si vous ça ne sert à rien

S55routed pour router selon le protocole **RIP**, à désactiver

S57diald permet de lancer le daemon **diald** pour lancer une connexion internet automatiquement (du serveur ou d'un client du réseau local), à désactiver si vous vous en servez pas

S60lpd système d'impression

S60nfs pour lancer le serveur **NFS**, à désactiver si vous n'exportez pas vos systèmes de fichiers

S60mars-nwe pour lancer un serveur **Netware** de **Novell**, à désactiver si non nécessaire

S72autofs pour lancer l'automontage (peut s'appeller aussi **S72amd**), à virer si vous ne l'utilisez pas

S75keytable pour activer le clavier qui va bien (clavier français azerty)

S75gated pour lancer d'autres protocoles de routage comme **OSPF**, à désactiver

S80sendmail pour lancer **sendmail**, à désactiver si vous ne l'utilisez pas

S85sound pour activer le son

S85gpm permet d'avoir la souris sur des applis textes comme **Midnight Commander**, à désactiver si inutile

S85http pour lancer le serveur Web (**Apache**), à désactiver si non utilisé

S86ypbind si vous êtes un client **NIS**, à désactiver sinon

S90squid pour lancer le proxy **squid**, pour partager la connexion internet, à désactiver si poste isolé

S90xfs pour activer le serveur de fonts X, nécessaire

S91smb pour lancer le serveur **samba**, si vous partager des file système ou des imprimantes vers des postes **Windows**, à désactiver sinon

S94ntpd pour lancer le serveur **NTP** (network time protocol) à désactiver (ancienne version **S94xntp**)

S95innd pour lancer le serveur de news **innd**, si non utilisé à désactiver

S99linuxconf permet à quelqu'un sur internet de faire de la maintenance sur votre système à travers une interface web, à désactiver

S99local c'est un lien vers **/etc/rc.d/rc.local**, où vous pouvez rajouter vos petits trucs

C'est une liste non exhaustive, d'un système à l'autre, le numéro de lancement peut changer. Pour désactiver le service **SNMP** il suffit de taper :

```
chkconfig --level 0123456 snmpd off
```

Un autre moyen est de supprimer les liens vers **/etc/rc.d/init.d/snmpd** se trouvant sous **/etc/rc.d/rcX.d** (remplacer X par 0, 1, 2, 3, 4, 5 et 6)

3.3.4 Sécuriser /etc/passwd

Tout d'abord on va mettre en place les **shadow password**, comme ça tous les mots de passe seront contenus dans un fichier distinct de **/etc/passwd**, ce fichier **/etc/shadow** est en lecture seule pour root (droit 400 proprio root). La première chose qu'un hacker cherche à faire est de lire **/etc/passwd**, le fait de mettre les password ailleurs apportent une protection supplémentaire.

Pour activer les **shadow password**, c'est très simple en tant que root, vous devez taper:

pwconv

Dans le champ password de **/etc/passwd** (le deuxième) vous devriez trouver un **x**, un fichier **/etc/shadow** a été créé contenant les mots de passe crypté.

Maintenant on va supprimer autant que possible tous les comptes systèmes inutiles, si vous en avez pas besoin, supprimez les, car ce sont autant de portes d'entrée pour les hackers. Les comptes systèmes suivants sont nécessaires:

root, **bin**, **daemon**, **adm**, **lp** (si vous avez un système d'impression), **mail** (si serveur mail), **news** (si serveur de news), **uucp** (si vous utilisez UUCP), **nobody**

Ceux-ci sont facultatifs:

games, **gopher**, **halt**, **sync**, **shutdown**, **operator**, **ftp** (si serveur FTP anonyme), **lists**, **xfs**.

3.3.5 Sécuriser FTP

Si vous voulez activer **FTP**, vous pouvez le faire avec dé commentant la ligne le concernant dans **/etc/inetd.conf**, on va faire cependant en sorte que le système logue tout ce qui concerne **FTP**. La ligne dans **/etc/inetd.conf** est:

```
ftp  stream tcp  nowait root  /usr/sbin/tcpd in.ftpd -l -L -i -o
```

avec:

- l** chaque session **FTP** est loguée
- L** toutes les commandes utilisateurs sont loguées

- i chaque fichier récupéré est logué
- o chaque fichier envoyé est logué

En cas d'utilisation de **xinetd** voici un exemple de fichier **ftp** à placer sous **/etc/xinetd.d** définissant une plage horaire et une limitation à 4 personnes connectées simultanément

```
service ftp
{
    socket_type    = stream
    wait          = no
    protocol      = tcp
    user          = root
    server        = /usr/sbin/in.ftpd
    instances     = 4
    access_times  = 7:00-12:00 14:00-17:00
    nice=15
}
```

Maintenant vous allez créer un fichier **/etc/ftpusers** qui va contenir la liste des utilisateurs qui n'ont pas le droit d'ouvrir une session **FTP** sur votre système, dans ce fichier vous devrez y mettre tous les utilisateurs systèmes, voici un exemple de **/etc/ftpusers**:

```
root
bin
daemon
adm
lp
mail
news
nobody
```

3.3.6 Sécuriser Telnet

On doit empêcher que root puisse accéder au système via **telnet**, ça force les utilisateurs à se loguer sur leur compte habituel puis de faire un **su** en local s'ils veulent devenir root. Pour cela le fichier **/etc/securetty** doit contenir uniquement des terminaux locaux (du style **ttyX**) et aucunement des pseudos terminaux (du style **ttypX**) qui permettent à un utilisateur distant de se loguer à distance en tant que root. Voici un exemple de **/etc/securetty** :

```
tty1
tty2
tty3
```

tty4
tty5
tty6
tty7
tty8

Maintenant quand vous vous connectez avec **telnet**, vous avez toujours un petit commentaire qui s'affiche, du style "bienvenu sur la machine untel", vous pouvez modifier cela en mettant le commentaire souhaité dans le fichier **/etc/issue**.

Attention sous une **Mandrake**, le fichier **/etc/issue** est régénéré à chaque boot, pour éviter cela, modifiez la ligne qui va bien dans **/etc/rc.d/rc.local**.

3.3.7 Les TCP Wrappers

Les **TCP Wrappers** rajoutent une protection supplémentaire aux services lancés par **inetd**, c'est en fait un contrôle d'accès, par exemple pour se connecter avec **telnet**, il faut d'abord passer le contrôle des **TCP Wrappers**, une fois passer ce contrôle on peut alors se connecter à **telnet**, si on ne passe pas le contrôle des **TCP Wrappers**, la session **telnet** n'est même pas ouverte. Les **TCP Wrappers** permettent évidemment de loguer chaque connexion (réussie ou pas). Les fichiers de configuration des **TCP Wrappers** pour fixer les accès sont **/etc/hosts.allow** et **/etc/hosts.deny**. Ces fichiers déterminent qui peut ou ne peut pas accéder aux systèmes (ou du moins aux services lancés par **inetd**).

La règle est d'interdire tout à tout le monde, puis d'autoriser uniquement certains postes très limités à utiliser vos services. La syntaxe est la suivante:

service: source(adresse IP, réseau, ou nom): optionnel: ALLOW ou DENY

Par exemple **/etc/hosts.deny**:

ALL: ALL DENY

Pour **/etc/hosts.allow**

in.telnetd:192.168.13.10::ALLOW

in.ftpd:192.168.13.0/255.255.255.0::ALLOW

NOTES: - Pour le nom du service **in.telnetd** est le dernier champ de la ligne **telnet** du fichier **/etc/inetd.conf**

- Préférez les adresses IP, plutôt que le nom
- **192.168.13.10** est l'adresse IP d'un hôte autorisé
- **192.168.13.0/255.255.255.0** est un sous réseau complet

Les **TCP_WRAPPERS** concernent encore **xinetd** mais devraient disparaître dans le futur car **xinetd** offre des contrôles d'accès supérieurs à ce que peut offrir les **TCP_WRAPPERS**.

3.3.8 Root et utilisateurs privilégiés

Maintenant vous pouvez faire en sorte que seuls certains utilisateurs aient le droit d'utiliser certaines commande "puissantes" comme **su**. En limitant le nombre de personnes pouvant utiliser ces commandes vous améliorez la sécurité de votre site. Pour cela vous allez créer un groupe d'utilisateur privilégié, généralement il est appelé **wheel**, mais libre à vous de l'appeler comme vous voulez, choisissez quand même un nom discret, passe partout, pour ne pas éveiller les soupçons.

Maintenant pour que **su** soit lancé uniquement par les membres du groupe **wheel**, vous devrez taper:

```
chgrp wheel /bin/su
chmod 4750 /bin/su
```

Faites de même pour les autres commandes.

NOTES:

- N'oubliez pas qu'un utilisateur peut très bien appartenir à deux groupes
- N'oubliez pas [sudo](#) pour donner des droits privilégiés à certains utilisateurs.

Pour ce qui concerne root, vous devez prendre quelques précautions:

- vous ne devez en aucun rajouter le **.** (répertoire courant) dans le **PATH** de root, car si par malheur quelqu'un crée un script avec droits exécutable dans **/tmp** qui s'appelle **rm** contenant:

```
#!/bin/bash
rm -Rf /
```

Si root a le malheur de taper **rm** alors qu'il se trouve dans **/tmp**, c'est le script qui peut être appelé (suivant l'ordre des chemins dans le **PATH**) et pouf ! plus de système.

3.3.9 Sécuriser les fichiers et systèmes de fichiers

3.3.9.1 SUID et GUID

Evitez d'avoir recours aux SUID et SGID, ils comportent certains risques au niveau de la sécurité, ils permettent à n'importe qui de lancer un programme avec les droits du propriétaire du programme. Les fichiers SUID sont une des principales cibles des hackers, à éviter donc, préférez amplement [sudo](#).

Pour trouver les fichiers avec SUID ou SGID, tapez en tant que root:

```
find / -type f \( -perm 04000 -o -perm 02000 \)
```

3.3.9.2 Fichiers .rhosts et hosts.equiv

On a vu plus haut qu'il fallait éviter d'avoir des fichiers **.rhosts** ou **hosts.equiv**, on va donc bloquer ces deux fichiers pour que personne ne puisse les recréer. Pour bloquer les fichiers, il suffit de taper les commandes:

```
touch /.rhosts /etc/hosts.equiv  
chmod 0 /.rhosts /etc/hosts.equiv
```

Pour trouver les **.rhosts** dans les homedirectories des utilisateurs, taper:

```
find /home -name .rhosts -print
```

3.3.9.3 Umask

On peut définir les droits de création par défaut d'un fichier ou de répertoire avec la commande **umask**, on va faire en sorte d'éviter de créer des répertoires avec les droits 777.

Pour définir l'**umask** pour tous les utilisateurs du système, vous devez éditer le fichier **/etc/profile** et modifiez la ligne concernant **umask**, vous pouvez fixer 022, 027 ou même 077 qui est le masque le plus restrictif (voir mon cours unix pour voir comment marche les **umask**).

4 Auditer la sécurité de son réseau

4.1 Présentation

Le but de cette page est de vous présenter trois outils qui vous permettront de tester la sécurisation des machines de votre réseau, ils vous révéleront vos trous de sécurité et vous

avertiront des problèmes potentiels, à partir de là libre à vous de "boucher" les trous en question et d'upgrader certains programmes présentant quelques déficiences de sécurité. Ces outils procèdent en scannant votre machine, en testant tous les ports ouverts notamment et en testant un grand nombre de trous de sécurité connus.

Les outils présentés sont **SARA** qui est dérivé du célèbre **SATAN** qui n'a pas été maintenu depuis un certain temps, **nmap** qui est un puissant "scanneur" et **nessus** basé entre autres sur **nmap** mais avec en plus une interface particulièrement conviviale. Des trois outils c'est le dernier que je juge le plus puissant et de surcroît facile d'utilisation.

4.2 AVERTISSEMENT

Je vous déconseille évidemment fortement de tenter de scanner une machine ne vous appartenant pas se trouvant sur le net, le scan sera considéré comme une attaque, vous vous exposez à des problèmes en proportion avec la machine visitée, mais bon je vous aurais prévenu.

4.3 Sara

4.3.1 Présentation

En 1995 est apparu un outil d'administration pour tester la sécurité d'un réseau, avec le nom évocateur de **SATAN** (Security Administrator Tool for Analyzing Networks). Il avait la particularité d'être Open Source, il est très vite devenu un standard et fut à la base d'une pléthore d'outils équivalents. Le problème est que maintenant **SATAN** commence fortement à dater, ça fait un paquet de temps qu'il n'a pas été remis à niveau, en d'autres termes, il est actuellement complètement dépassé. En conséquence une boîte (Advanced Research Corporation) commença le développement d'un outil similaire au goût du jour, ainsi naquit **SARA** (Security Auditor Research Assistant), comme **SATAN**, **SARA** est Open Source.

A noter qu'il existe une version commerciale de **SARA**, appelée **SARA-PRO**. A noter encore qu'un des auteurs de **SARA** est aussi à l'origine de **SAINT**, un autre SATAN-like.

4.3.2 Installation

On peut récupérer l'archive de **SARA** à savoir **sara-4.2.7.tar.gz** à l'URL www-arc.com/sara, qu'on décompressera en tapant :

```
tar xvfz sara-4.1.4b.tar.gz
```

Après décompression, vous allez récupérer un répertoire **sara-4.2.7**. Vous devez avoir préalablement installé le package **tcsh**. Puis dans le répertoire ainsi obtenu, il suffit de taper successivement :

./configure

Puis en tant que root

make

Lors de la compilation, il peut se plaindre de l'absence de certaines commandes, mais ce n'est pas bien grave.

Looking for all the commands now...

AEEEEIII...!!! can't find mail

AEEEEIII...!!! can't find tftp

AEEEEIII...!!! can't find ypcat

AEEEEIII...!!! can't find finger

AEEEEIII...!!! can't find rusers

AEEEEIII...!!! can't find ypwhich

AEEEEIII...!!! can't find rlogin

AEEEEIII...!!! can't find rsh

Il va créer des exécutables sous **./sara-4.2.7/bin** . L'exécutable de lancement de **SARA** se trouve directement sous **./sara-4.2.7**.

4.3.3 Utilisation

En tant que root il suffit de taper

./sara-4.2.7/sara

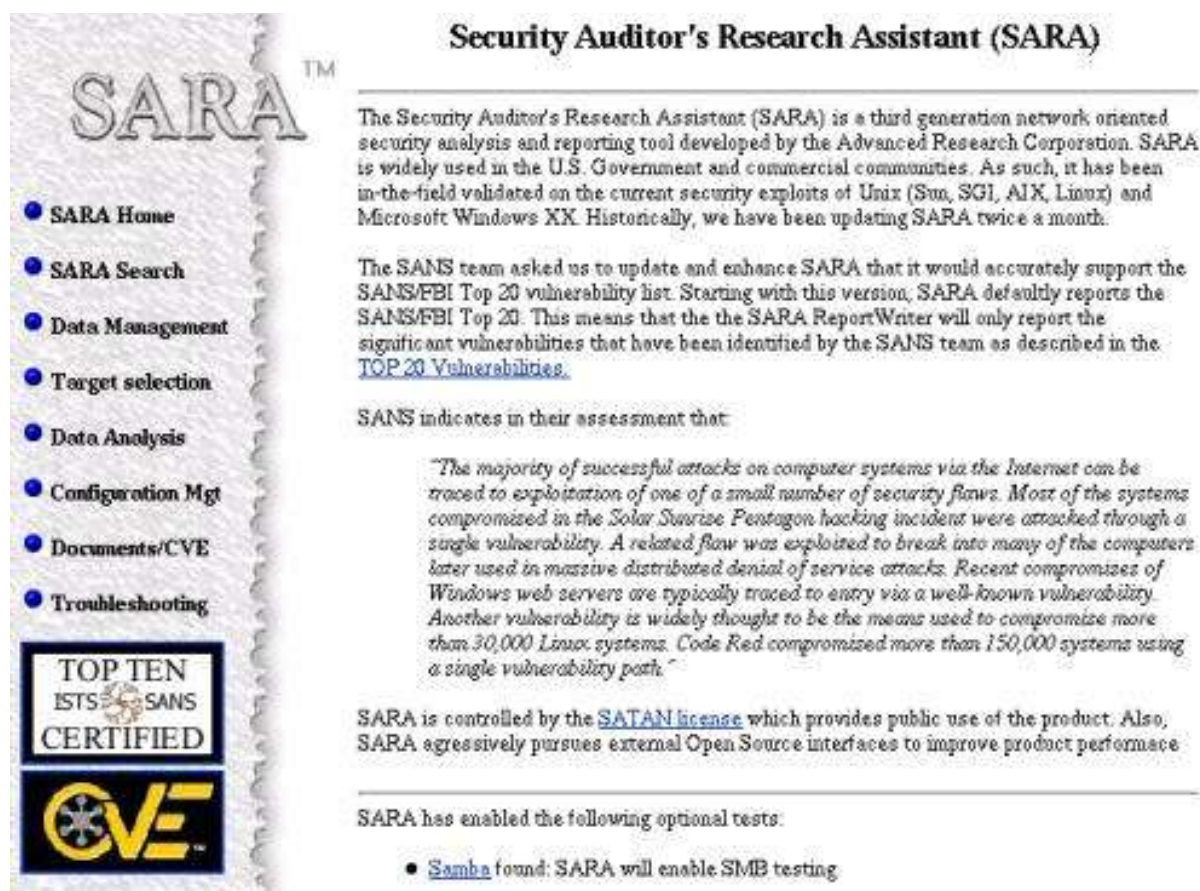
NOTE: Si vous disposez de **nmap** (voir le chapitre suivant), vous pouvez rajouter l'option **-n** pour lui indiquer, le scan sera d'autant meilleur.

Et là surprise, c'est le browser par défaut qui se lance. Pour **mozilla**, il y a une chose à modifier dans les **Préférences** pour que ça puisse marche, aller dans le menu:

Edition -> Préférences -> Navigateur-> Applications -> dans la liste choisir **Perl Program** (au besoin créer un **Nouveau Type**)-> puis en final **Edition**, modifiez les champs tels qu'indiqués sur le screenshot à droite.



Pour **Konqueror**, il n'y a rien à faire. Voilà maintenant dans le browser vous avez la page d'accueil de **Sara**



A noter que la partie **Documents/CVE** est très bien fournie ainsi que la section **Troubleshooting** se présentant sous forme de FAQ.

La première étape est de sélectionner la cible à analyser. on clique donc sur **Target Selection**, vous devez alors saisir:

Primary target selection: choix de la cible

Vous pouvez choisir une machine particulière (**target host**), un sous réseau complet (**network**), ou une plage d'adresse (**range**), la syntaxe est la suivante :

- un hôte simple hote.local.com
- plusieurs hôtes: hote1.local.com hote2.local.com
- un sous réseau: 192.168.13.0/24
- une plage d'adresse: 192.168.0.55-192.168.0.98

Scanning level selection choix du niveau de scan

Les choix sont :

- Light : léger (pour que la machine cible ne se rende pas compte)
- Normal : peut être détecté et inscrit dans les fichiers de log de la cible
- Heavy : lourd, des messages d'erreurs peuvent apparaître sur les consoles système de la cible
- Extreme : extrême, ça peut planter
- Custom : personnalisé (seulement scan SMB (samba), Web, Mail, Telnet, FTP)
- Custom : personnalisé (seulement scan TCP)
- Custom : personnalisé (seulement un scan de test)

Firewall Support utilisation d'un firewall

Est ce que la cible se trouve derrière un firewall ?

On a le choix entre oui et non

Pour l'exercice, je choisis une machine sous Mandrake avec un scanning **heavy** qui n'est pas derrière un firewall mais sur mon réseau local.

Voilà maintenant on a un bouton **start the scan**, et c'est parti, là ça peut prendre un certain temps, voilà le résultat final sur la machine **obelix.armoric.bz** dans une page du navigateur.

Data collection in progress...

Adding a primary target

Add-primary: obelix.breizland.bz

Add-target: obelix.breizland.bz prox 0

policy: obelix.breizland.bz prox 0 level 2

Check-pulse: obelix.breizland.bz

==> running bin/timeout 180 bin/fping obelix.breizland.bz

process_targets: probe obelix.breizland.bz...

Prox: 0

AL : 2

Add-todo: obelix.breizland.bz|dns.sara|

Add-todo: obelix.breizland.bz|rpc.sara|

Add-todo: obelix.breizland.bz|finger.sara|

Add-todo: obelix.breizland.bz|ddosscan.sara|

Add-todo: obelix.breizland.bz|hosttype.sara|
Add-todo: obelix.breizland.bz|tcpscan.sara
1-1525,1527-5404,5406-5899,5901-7099,7101-8887,8889-
9999,12345,16600,20034,27374,27665,31337,31785,65000|
Add-todo: obelix.breizland.bz|udpscan.sara
1-1760,1763-2050,31335,31337,27444,32767-33500|
==> running bin/timeout 20 bin/ddosscan.sara obelix.breizland.bz
==> running bin/timeout 180 bin/udpscan.sara
1-1760,1763-2050,31335,31337,27444,32767-33500 obelix.breizland.bz
Add-fact: obelix.breizland.bz|#|a|x|_|_|offers #
Add-fact: obelix.breizland.bz|echo|a|x|_|_|offers echo
Add-fact: obelix.breizland.bz|bootps|a|x|_|_|offers bootps
Add-fact: obelix.breizland.bz|sunrpc|a|x|_|_|offers sunrpc
Add-fact: obelix.breizland.bz|netbios-ns|a|x|_|_|offers netbios-ns
Add-fact: obelix.breizland.bz|netbios-dgm|a|x|_|_|offers netbios-dgm
==> running bin/timeout 20 bin/finger.sara obelix.breizland.bz
Add-fact: obelix.breizland.bz|finger.sara|u|_|_|program timed out
==> running bin/timeout 20 bin/rpc.sara obelix.breizland.bz
Add-fact: obelix.breizland.bz|nfs|a|x|_|_|runs NFS
Add-fact: obelix.breizland.bz|mouted|a|x|_|_|runs NFS
Add-fact: obelix.breizland.bz|statd|a|x|_|_|runs statd
==> running bin/timeout 20 bin/dns.sara obelix.breizland.bz
==> running bin/timeout 180 bin/hosttype.sara obelix.breizland.bz
==> running bin/timeout 180 bin/tcpscan.sara
1-1525,1527-5404,5406-5899,5901-7099,7101-8887,8889-
9999,12345,16600,20034,27374,27665,31337,31785,65000
obelix.breizland.bz
Add-fact: obelix.breizland.bz|ipp|a|_|_|HTTP/1.0 400 Bad Request\r\nDate: Sat, 10 Mar
2001 09:01:04 GMT\r\nServer: CUPS/1.1\r\nContent-Type: text/html\r\nContent-
Length: 101\r\n\r\n
Bad Request
Bad Request\r\n|offers ipp
Add-fact: obelix.breizland.bz|masqdiabler|a|_|_|READY\r\n|offers masqdiabler
Add-fact: obelix.breizland.bz|tproxy|a|_|_|offers tproxy
Add-fact: obelix.breizland.bz|nntp|a|_|_|200 Leafnode NNTP Daemon, version 1.9.18
running at obelix.breizland.bz\r\n|offers nntp
Add-fact: obelix.breizland.bz|blackjack|a|_|_|offers blackjack
Add-fact: obelix.breizland.bz|smtp|a|_|_|220 rennes-1-a7-21-114.dial.proxad.net
ESMTP Sendmail 8.11.0/8.11.0/Olivier Hoarau-992911; Sat, 10 Mar 2001 10:01:11
+0100\r\n221 2.0.0 rennes-1-a7-21-114.dial.proxad.net closing connection\r\n|offers
smtp
Add-fact: obelix.breizland.bz|submission|a|_|_|220 rennes-1-a7-21-114.dial.proxad.net
ESMTP Sendmail 8.11.0/8.11.0/Olivier
Hoarau-992911; Sat, 10 Mar 2001 10:01:11 +0100\r\n221 2.0.0

[illegible]

Add-todo: obelix.breizland.bz|sample.sara.ext|http
 Add-todo: obelix.breizland.bz|http.sara|http-alt
 Add-todo: obelix.breizland.bz|sample.sara.ext|http-alt
 Add-fact: obelix.breizland.bz|http-alt|a|g|offers http:http-alt
 Add-todo: obelix.breizland.bz|depends.sara|nfs
 Add-fact: obelix.breizland.bz|nfs|a|g|runs NFS
 Add-todo: obelix.breizland.bz|sendmail.sara|
 Add-todo: obelix.breizland.bz|relay.sara|
 Add-todo: obelix.breizland.bz|login.sara|-u root
 Add-todo: obelix.breizland.bz|login.sara|-u guest
 Add-todo: obelix.breizland.bz|depends.sara|telnet
 Add-todo: obelix.breizland.bz|showmount.sara|
 Add-todo: obelix.breizland.bz|nfs-chk.sara|-t 10
 Add-fact: obelix.breizland.bz|mountd|a|g|runs NFS
 Add-todo: obelix.breizland.bz|ssh.sara|
 Add-todo: obelix.breizland.bz|ftp.sara|
 Add-todo: obelix.breizland.bz|xhost.sara|-d obelix.breizland.bz:0
 Add-fact: obelix.breizland.bz|ipp|a|g|offers http:ipp
 Add-todo: obelix.breizland.bz|http.sara|ipp
 Add-todo: obelix.breizland.bz|sample.sara.ext|ipp
 Add-fact: obelix.breizland.bz|http-alt|a|g|offers http
 ==> running bin/timeout 45 bin/login.sara -u root obelix.breizland.bz
 Add-fact: obelix.breizland.bz|telnet|a|g|
 ==> running bin/timeout 45 bin/login.sara -u guest obelix.breizland.bz
 ==> running bin/timeout 180 bin/http.sara ipp obelix.breizland.bz
 Add-fact: obelix.breizland.bz|ipp|a|g|offers http
 ==> running bin/timeout 700 bin/smb.sara obelix.breizland.bz
 Add-fact:
 obelix.breizland.bz|netbios-ssn|a|zwoi|ANY@obelix.breizland.bz|
 ANY@obelix.breizland.bz|netbios
 over the internet|Is your Netbios secure
 ==> running bin/timeout 20 bin/xhost.sara -d obelix.breizland.bz:0
 obelix.breizland.bz
 ==> running bin/timeout 20 bin/ftp.sara obelix.breizland.bz
 Add-fact: obelix.breizland.bz|ftp|a|g|offers ftp
 ==> running bin/timeout 20 bin/showmount.sara obelix.breizland.bz
 Add-fact: obelix.breizland.bz|showmount|a|g|Not running showmount or other
 error
 ==> running bin/timeout 20 bin/depends.sara telnet obelix.breizland.bz
 ==> running bin/timeout 180 bin/http.sara http-alt obelix.breizland.bz
 Add-fact: obelix.breizland.bz|http-alt|a|g|offers http
 ==> running bin/timeout 20 bin/sample.sara.ext http-alt obelix.breizland.bz
 ==> running bin/timeout 20 bin/depends.sara statd obelix.breizland.bz
 Add-fact: obelix.breizland.bz|statd|a|rcio|ANY@ANY|ANY@ANY|rpc statd access|

rpc.statd on Linux is vulnerable if not patched

==> running bin/timeout 20 bin/sendmail.sara obelix.breizland.bz

**Add-fact: obelix.breizland.bz|smtp|a|zcio|ANY@ANY|ANY@ANY|sendmail version|
sendmail VRFY command may provide hacker information**

**Add-fact: obelix.breizland.bz|smtp|a|zcio|ANY@ANY|ANY@ANY|sendmail version|
sendmail EXPN command may provide hacker information**

==> running bin/timeout 120 bin/nfs-chk.sara -t 10 obelix.breizland.bz

==> running bin/timeout 20 bin/sample.sara.ext http obelix.breizland.bz

==> running bin/timeout 700 bin/relay.sara obelix.breizland.bz

Add-fact:

**obelix.breizland.bz|smtp|a|ycio|ANY@obelix.breizland.bz|ANY@obelix.breizland.bz|
SMTP may be a mail relay|Probable smtp relay (spam)**

==> running bin/timeout 180 bin/http.sara http obelix.breizland.bz

Add-fact: obelix.breizland.bz|http|a|g|_|_|_|offers http

==> running bin/timeout 20 bin/depends.sara nfs obelix.breizland.bz

**Add-fact: obelix.breizland.bz|nfsd|a|zcio|ANY@ANY|ANY@ANY|mountd
vulnerabilities|nfsd version may be vulnerable to buffer overflow**

==> running bin/timeout 20 bin/ssh.sara obelix.breizland.bz

Add-fact: obelix.breizland.bz|ssh|a|g|_|_|_|offers ssh

==> running bin/timeout 20 bin/sample.sara.ext ipp obelix.breizland.bz

Waiting for all processes to complete

Data collection completed (1 host(s) visited).

Back to the SARA start page | Continue with report and analysis | View primary target results

ATTENTION Le scan peut provoquer le blocage de certains services de la machine cible.

Si vous choisissez **View primary target results**, vous obtiendrez un truc du style:

General host information:

Host type: unknown type

NFS server

NIS server

NNTP (Usenet news) server

SSH server

Telnet server

WWW server
Subnet 192.168.13
Scanning level: heavy
Last scan: Fri Mar 10 19:01:23 2001

Vulnerability information:

Is your Netbios secure
sendmail EXPN command may provide hacker information
sendmail VRFY command may provide hacker information
rpc.statd on Linux is vulnerable if not patched
Probable smtp relay (spam)
SMTP may be a mail relay
nfsd version may be vulnerable to buffer overflow

Vous constatez qu'il y a des liens un peu partout qui vous donne plus d'info.

Si par contre vous choisissez "**Continue with report and analysis**", vous aurez alors le menu suivant:

Table of contents

Vulnerabilities

By Approximate Danger Level
By Type of Vulnerability
By Vulnerability Count

Host Information

By Class of Service
By System Type
By Internet Domain
By Subnet
By Host Name

Trust

Trusted Hosts

Trusting Hosts

Reporting

SARA Pro Reporter

Chaque catégorie étant un lien vers un autre page avec davantage de détails. Je ne vous les présenterai pas de manière exhaustive, car ça part vraiment dans tous les sens.

4.4 Nmap

4.4.1 Présentation

Nmap est un puissant outil qui permet de scanner les ports, il a pour but de tester la sécurisation des postes de vos réseau, vous devez vous en servir uniquement pour votre réseau, si vous tentez de scanner quelqu'un d'autres sur le réseau internet, sachez que ce sera considéré comme une attaque, c'est donc à vos risques et périls...

Pour info **nmap** se trouve maintenant intégré dans la mandrake, je vous présenterai l'installation avec le tarball avec la dernière version stable 3.48, l'installation avec les rpm de la Mandrake se réduit à sa plus simple expression. L'utilisation de **nmap** que vous l'avez installé avec le rpm ou avec le tarball reste la même

4.4.2 Installation avec le tarball

Vous pouvez trouver **nmap** sur le site www.insecure.org/nmap La dernière version stable est la 3.48 qu'on peut récupérer sous forme de tarball.

Attention de ne pas installer **nmap** si vous disposez déjà de la version Mandrake, pour le savoir:

```
rpm -qa | grep -i nmap
```

Si vous obtenez

```
nmap-3.00-1mdk
```

```
nmap-frontend-3.00-1mdk
```

On va supprimer ces deux packages.

```
rpm -e nmap-3.00-1mdk  
rpm -e nmap-frontend-3.00-1mdk
```

Pour décompresser le tarball rien de plus simple.

```
tar xvfz nmap-3.48.tar.gz
```

Cela va créer dans le répertoire courant un répertoire **nmap-3.48**. Au préalable vous veillerez à installer si ce n'est déjà fait sur votre Mandrake les packages suivants: **byacc**, **glib-devel**, **gtk+-devel**, **flex** et **libpcap**

Puis la commande classique:

```
./configure
```

Tapez maintenant

```
make
```

Puis en tant que **root**

```
make install
```

Cela va installer les exécutables **nmap** et **nmapfe** sous **/usr/local/bin**, et d'autres fichiers sous **/usr/local/share/nmap**, si ça ne vous convient pas, vous pouvez changer les chemins en tapant **configure** avec les arguments qui vont bien(**./configure --help** pour la syntaxe).

4.4.3 Syntaxe

La syntaxe est classique:

```
nmap [options éventuelles] cible
```

Les options principales sont:

- sT** scanning des ports TCP ouverts, on ouvre une connexion sur tous les ports ouverts, toutes les connexions sont visibles sur la machine cible (dans les fichiers de log notamment),
- sS** scanning des ports TCP, on envoie un message SYN pour dire qu'on va ouvrir une connexion TCP puis on attend la réponse, après réponse on sait que le port est ouvert, l'avantage de cette option est que l'action n'est pas loguée par la cible,
- sF**,**-sX**,**-sN** Stealth FIN, Xmas, ou Null scan (marche que sous UNIX) scan des ports plus discrets (voir **man nmap** pour plus de détails)
- sP** équivalent à **ping**, pour voir si la cible est "alive", ne fait pas de scan,
- sU** scanning des ports UDP ouverts (plutôt lent),
- b** **<ftp_relay_host>** ftp "bounce attack" port scan, en gros ça exploite un trou de sécurité sur certains proxy, je vous en dirais pas plus (voir le **man**).

D'autres options intéressantes (liste non exhaustive):

- O** permet de connaître sur quel OS tourne la cible (fingerprinting en anglais),
- p** **<range>** syntaxe pour avoir un ensemble de ports à scanner, exemple :
 - p 23** on va regarder que le port 23
 - p 20-30,63000-** va scanner les ports de 20 à 30 et supérieur à 63000 (jusqu'à 65535 en fait)par défaut il scrute uniquement de 1 à 1024 plus les ports se trouvant dans **/etc/services**,
- F** scan rapide, scanne uniquement les ports contenus dans **/etc/services**,
- I** permet d'avoir plus d'info sur les ports TCP ouverts, notamment le proprio,
- o** **<logfile>** log le résultat dans le fichier **<logfile>**,
- v** Verbose, mode verbeux (recommandé),
- h** help, pour avoir la liste complète des options,
- V** pour avoir uniquement le numéro de version, pas la peine de rentrer de cible,
- e** **<devicename>**, avec ça on peut spécifier une interface particulière pour envoyer les paquets (**eth0**, **ppp0**, etc.).

Maintenant la cible peut être identifiée par son nom sur internet ou son adresse IP. On peut désigner un ensemble d'adresse aussi, ou des sous-masques de réseau particulier. Par exemple **domaine.org/24** ou **192.88.209.5/24**, **192.88.209.0-255** ou bien encore **'128.88.209.*'**.

4.4.4 Quelques exemples

L'utilisation la plus simple est celle-ci:

nmap -v cible

Starting nmap V. 3.48 (www.insecure.org/nmap/)

No tcp,udp, or ICMP scantype specified, assuming SYN Stealth scan. Use -sP if you really don't want to portscan (and just want to see what hosts are up).

WARNING! The following files exist and are readable: /usr/local/share/nmap/nmap-services and ./nmap-services. I am choosing /usr/local/share/nmap/nmap-services for security reasons. set NMAPDIR=. to give priority to files in your local directory

Host cible (192.168.13.11) appears to be up ... good.

Initiating SYN Stealth Scan against cible (192.168.13.11)

Adding open port 25/tcp

Adding open port 6000/tcp

Adding open port 80/tcp

Adding open port 587/tcp

Adding open port 139/tcp

Adding open port 901/tcp

Adding open port 111/tcp

Adding open port 110/tcp

Adding open port 119/tcp

The SYN Stealth Scan took 4 seconds to scan 1601 ports.

Interesting ports on cible (192.168.13.11):

(The 1592 ports scanned but not shown below are in state: closed)

Port	State	Service
25/tcp	open	smtp
80/tcp	open	http
110/tcp	open	pop-3
111/tcp	open	sunrpc
119/tcp	open	nntp
139/tcp	open	netbios-ssn
587/tcp	open	submission
901/tcp	open	samba-swat
6000/tcp	open	X11

Nmap run completed -- 1 IP address (1 host up) scanned in 4 seconds

La commande suivante va permettre de scanner tous les ports TCP réservés :

nmap -sS -O cible

WARNING! The following files exist and are readable: /usr/local/share/nmap/nmap-os-fingerprints and ./nmap-os-fingerprints. I am choosing /usr/local/share/nmap/nmap-os-fingerprints for security reasons. set NMAPDIR=. to give priority to files in your local directory

Starting nmap V. 3.48 (www.insecure.org/nmap/)

WARNING! The following files exist and are readable: /usr/local/share/nmap/nmap-services and ./nmap-services. I am choosing /usr/local/share/nmap/nmap-services for security reasons. set NMAPDIR=. to give priority to files in your local directory

Interesting ports on cible (192.168.13.11):

(The 1592 ports scanned but not shown below are in state: closed)

Port	State	Service
25/tcp	open	smtp
80/tcp	open	http
110/tcp	open	pop-3
111/tcp	open	sunrpc
119/tcp	open	nntp
139/tcp	open	netbios-ssn
587/tcp	open	submission
901/tcp	open	samba-swat
6000/tcp	open	X11

Remote operating system guess: Linux Kernel 2.4.0 - 2.5.20

Uptime 0.033 days (since Sun Sep 1 04:45:16 2002)

Nmap run completed -- 1 IP address (1 host up) scanned in 7 seconds

Maintenant on va tester si **sshd**, **DNS**, **pop3d**, **imapd** (ports respectifs 22, 53, 110 et 143) tournent sur le système et si le port 4564 est utilisé:

nmap -sX -p 22,53,110,143,4564 cible

Starting nmap V. 3.48 (www.insecure.org/nmap/)

Interesting ports on cible (192.168.13.11):

(The 4 ports scanned but not shown below are in state: closed)

Port	State	Service
------	-------	---------

WARNING! The following files exist and are readable: /usr/local/share/nmap/nmap-services and ./nmap-services. I am choosing /usr/local/share/nmap/nmap-services for security reasons. set NMAPDIR=. to give priority to files in your local directory

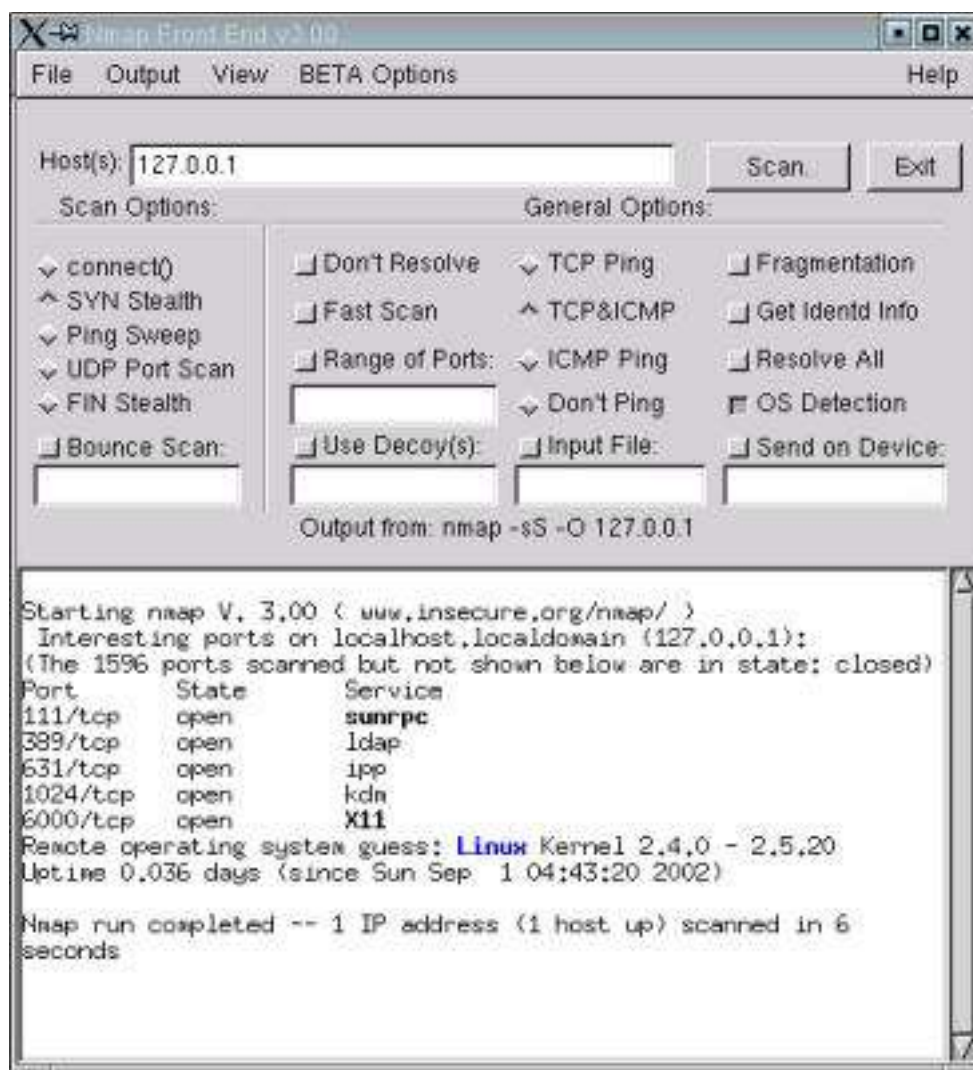
110/tcp	open	pop-3
---------	------	-------

Nmap run completed -- 1 IP address (1 host up) scanned in 2 seconds

On constate qu'il n'y a que le serveur POP3 qui marche sur la machine.

4.4.5 Le front end de nmap

En tant que root, il suffit de lancer **nmapfe**, voici ce que ça donne:



Vous voyez qu'il devient très simple d'utiliser **nmap**.

4.5 Nessus

4.5.1 Présentation

Nessus est un outil qui permet de tester la sécurisation des postes de votre réseau, il se base en autres sur **nmap**, mais en plus de tester vos ports, il va scruter les composants logiciels de votre machine, pour déterminer les trous de sécurité et vous avertir sur certaines faiblesses. **Nessus** est constitué d'une partie serveur et cliente, les deux peuvent très bien fonctionner sur la même machine.

4.5.2 Installation

Tout d'abord vous devez vous procurer les packages de **nessus** que vous trouverez à l'URL suivante www.nessus.org. Dézippez l'un après l'autre les packages dans un même répertoire

```
tar xvfz nessus-core-2.0.9.tar.gz
tar xvfz nessus-libraries-2.0.9.tar.gz
tar xvfz nessus-plugins-2.0.9.tar.gz
tar xvfz libnasl-2.0.9.tar.gz
```

Cela va donner les répertoires **nessus-core**, **nessus-libraries** , **nessus-plugins** et **libnasl**. Il faut d'abord installer le package **byacc** et **bison**, puis dans le répertoire **nessus-libraries** on tape

```
./configure
```

Un message vous indique de taper **uninstall-nessus** si vous voulez effacer une ancienne installation de **nessus**, on tape maintenant :

```
make
```

on passe en root puis on tape

```
make install
```

Les librairies sont installées sous **/usr/local/lib**, maintenant vous devez rajouter ce chemin dans **/etc/ld.so.conf** et tapez

ldconfig

Pour la suite des opérations les répertoires **/usr/local/bin** et **/usr/local/sbin** doivent être dans votre **PATH** (**env | grep PATH** pour vérifier) si ce n'est pas le cas, dans le fichier **.bashrc** de votre homedirectory rajoutez :

```
PATH=$PATH:/usr/local/bin:/usr/local/sbin  
export PATH
```

Et relancez un shell pour prendre en compte la modification (ou alors **source ~/.bashrc**)

Dans le répertoire **libnasl**, on tape :

```
./configure
```

Puis

```
make
```

Et enfin en tant que **root**

```
make install
```

suivi de

ldconfig

On tape les mêmes commandes (**./configure**, **make** et en tant que **root make install**) dans l'ordre dans le répertoire **nessus-core** puis **nessus-plugins**

Le daemon **nessusd** est installé sous **/usr/local/sbin**, les fichiers nécessaires sous **/usr/local/lib/nessus**

Maintenant on va créer un utilisateur **lambda** pour pouvoir utiliser **nessus**. En tant que root on tape :

nessus-adduser

Voilà les commentaires

Add a new nessusd user

Login : lambda

Méthode d'authentification (mot de passe ou certificat), je choisis la méthode par défaut password (**pass**),

Authentication (pass/cert) [pass] : pass

Mot de passe (en clair), attention on ne le rentre qu'une fois

Login password :

A ce niveau on peut créer des critères pour limiter **lambda** dans ses actions de scan, faire un **man nessus-adduser** pour plus de détail

User rules

nessusd has a rules system which allows you to restrict the hosts that olivier has the right to test. For instance, you may want him to be able to scan his own host only.

Please see the nessus-adduser(8) man page for the rules syntax

**Enter the rules for this user, and hit ctrl-D once you are done :
(the user can have an empty rules set)**

On tape CTRL-D pour stopper la saisie, par défaut les règles peuvent être vides s'il y en a pas.

On récapitule

Login : **lamba**
Password : **mot-de-passe-en-clair**
DN :
Rules :

Is that ok ? (y/n) [y]
user added.

Pour lancer le serveur on doit préalablement créer les certificats d'authentification, tapez en tant que root

nessus-mkcert

Vous pouvez taper **enter** à chaque ligne, si vous voulez aller vite

Creation of the Nessus SSL Certificate

This script will now ask you the relevant information to create the SSL certificate of Nessus. Note that this information will *NOT* be sent to anybody (everything stays local), but anyone with the ability to connect to your Nessus daemon will be able to retrieve this information.

CA certificate life time in days [1460]: 1460
Server certificate life time in days [365]: 365
Your country (two letter code) [FR]: FR
Your state or province name [none]: none
Your location (e.g. town) [Paris]: Papeete
Your organization [Nessus Users United]: Tahiti connection
Congratulations. Your server certificate was properly created.

/usr/local/etc/nessus/nessusd.conf updated

The following files were created :

. Certification authority :

Certificate = /usr/local/com/nessus/CA/cacert.pem

Private key = /usr/local/var/nessus/CA/cakey.pem

. Nessus Server :

Certificate = /usr/local/com/nessus/CA/servercert.pem

Private key = /usr/local/var/nessus/CA/serverkey.pem

Press [ENTER] to exit

Ce ne sera plus nécessaire de taper cette commande par la suite. Maintenant on tape

nessusd &

A noter que j'ai eu anciennement l'erreur suivante

Access rights problem with /usr/local/var/nessus/nessusd.messages (File has path segment with wrong owner)-- aborting[

Cela venait du fait que le répertoire **/usr/local/var** appartenait à **mysql** Il faut qu'il appartienne à **root**

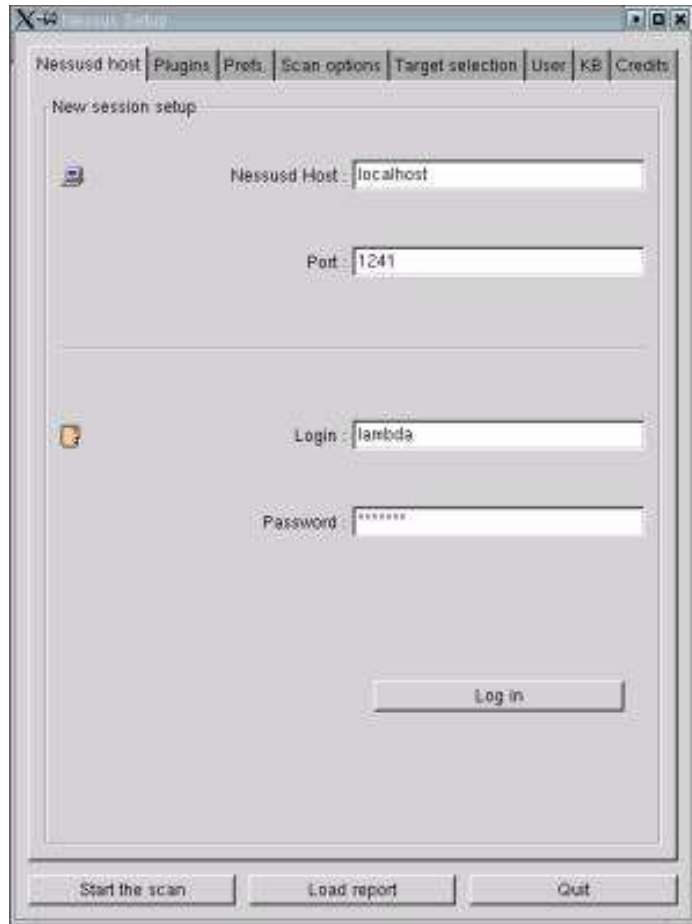
4.5.3 Utilisation

En supposant que le daemon **nessusd** est lancé, il suffit en tant qu'utilisateur quelconque de taper **nessus** pour lancer le client.

La fenêtre suivante (à droite) apparaît, vous devez saisir le nom de la machine où tourne le serveur, si la machine et le client tourne sur la même machine, vous pouvez laisser **localhost**. Laissez le port par défaut, c'est celui utilisé par **nessusd**.

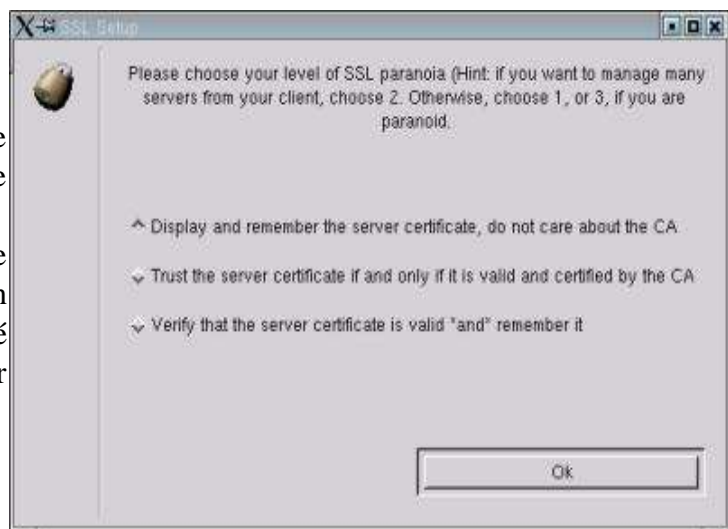
Vous devez au niveau du champ **Login** saisir l'utilisateur **nessus** que vous avez créé auparavant (**lambda** dans notre cas). Saisissez le mot de passe dans le champ **Password**.

Maintenant vous pouvez vous connecter au serveur en appuyant sur le bouton **Log in**. Vous voyez alors que le champ du bouton **Log in** se change en **Log out**, c'est bon vous êtes connecté.



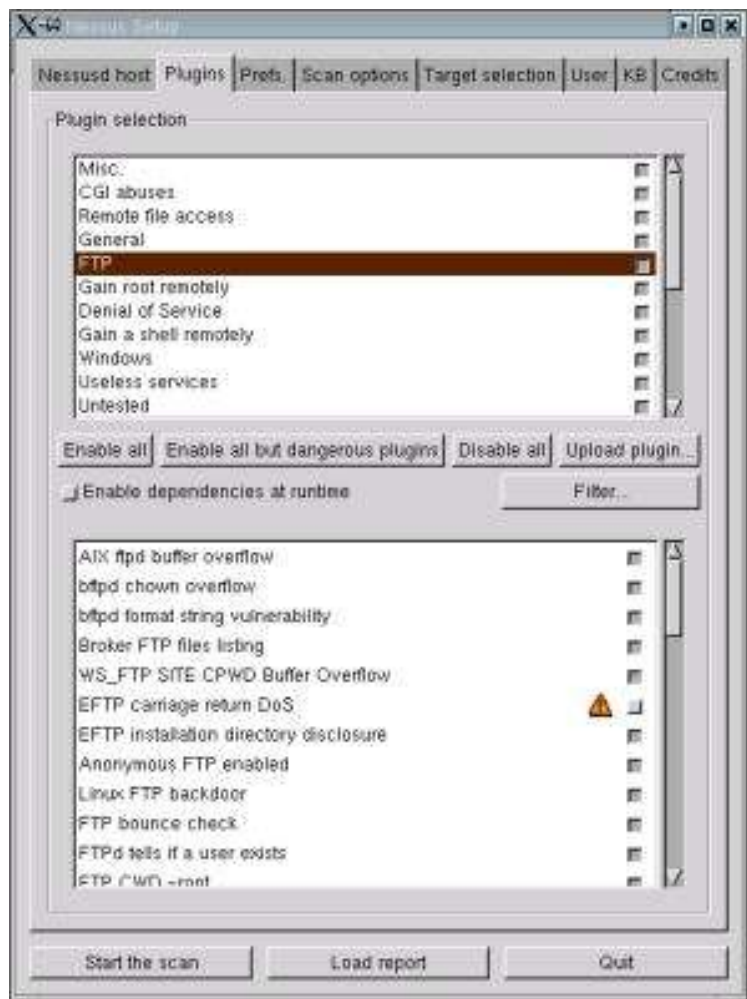
La fenêtre à droite apparaît, on clique OK. Une autre fenêtre affiche le certificat, on doit l'accepter ou non.

Une autre fenêtre nous indique que les plugins qui peuvent planter un service ou un hôte distant ont été désactivés. Il faut les réactiver pour avoir un scan exhaustif



Une fois connecté dans le champ **plugins**, vous verrez la liste des points à vérifier (liste du haut), la liste du bas contenant les détails du point sélectionné dans celle du haut, vous pouvez éventuellement invalider certains points à vérifier. A noter que vous avez des infos bulles fort riches quand vous passez la souris sur un champ. Par défaut tout est sélectionné. Sur le site de **nessus**, vous pouvez trouver d'autres plugins au gré de la découverte de nouveaux trous de sécurité, on vous explique aussi comment les installer, ou éventuellement carrément en créer d'autres tout seul dans son coin.

NOTE Par défaut les plugins qui peuvent faire planter la machine sont désactivés (comme le Denial of Service).



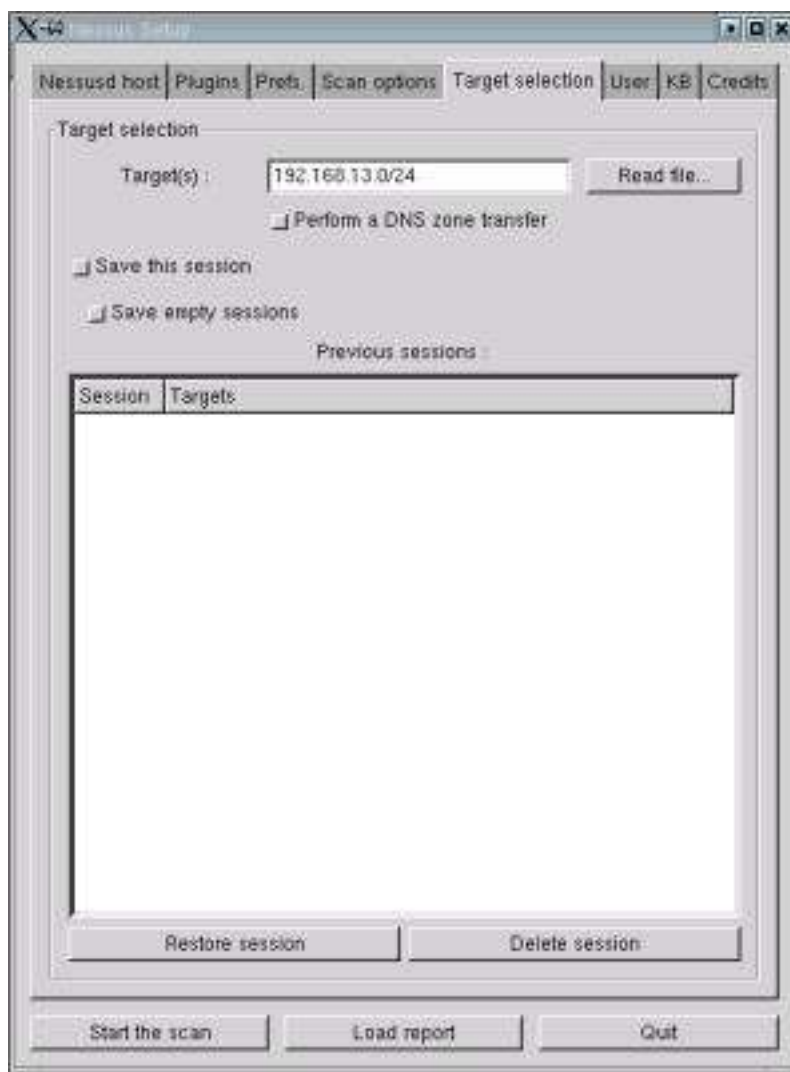
Dans l'onglet **Prefs**, vous pouvez saisir d'autres informations, notamment un nom d'utilisateur valide et son mot de passe si votre machine est aussi serveur POP ou SMB (samba).

Dans l'onglet **Scan options**, vous avez encore d'autres options pour la vérification, je n'y ai pas touché.

Dans l'onglet **User**, vous devez indiquer l'adresse email de la personne destinataire du rapport (**root@localhost** par défaut).

L'onglet **KB** permet de tout sauvegarder ou non dans une base de données

Dans l'onglet **Target Selection**, vous devez indiquer la machine cible dont on va auditer la sécurité en rentrant son adresse IP, éventuellement si vous voulez auditer toutes les machines du réseau 192.168.13.X, saisissez comme dans le screenshot 192.168.13.0/24.



Voilà, on peut lancer maintenant le scan en appuyant sur le bouton "**Start the scan**", une fenêtre apparaît avec une barre d'avancement. Le scan peut être relativement long, ça dépend évidemment de la puissance de votre machine, du coup soyez patient et ne l'interrompez pas même s'il paraît bloqué.



Une fois le scan terminé, une fenêtre **"Report"** apparaît avec le détail des trous de sécurité et des warnings, vous pouvez éventuellement sauvegarder le rapport au format désiré (html, txt, latex,) (bouton du milieu en bas). On a même une option de sauvegarde en html avec des graphes.

On voit ici que j'ai un trou de sécurité pour PHP, je dois passer à la version 4.2.2.



5 Détecter les attaques en temps réel

5.1 Présentation

Le but de cette page est de vous présenter des outils permettant de détecter en temps réel les attaques lorsque vous êtes connectés à internet, dans cette catégorie vous trouverez des outils de détection passifs se contentant de vous avertir d'attaques, libre à vous de mener les actions en conséquence, vous trouverez ensuite des outils de détection actifs, c'est à dire qu'ils vont eux mêmes automatiquement faire des manips systèmes pour bloquer les attaques le plus rapidement possibles.

Pour l'instant je vous présente deux outils **tcplogd** dans la catégorie détection passive et le très puissant **portsentry** pour une détection active.

5.2 Tcplogd

5.2.1 Présentation

Tcplogd est un outil qui permet une détection des attaques du style scan de port TCP par nmap, toute attaque est inscrite dans le fichier `/var/log/messages` via **syslogd**.

5.2.2 Installation

Vous trouverez l'archive [tcplogd-0_1_5pre1.tar.gz](http://www.kalug.lug.net/tcplogd-0_1_5pre1.tar.gz) à l'URL <http://www.kalug.lug.net/tcplogd> qui n'a plus l'air d'être actif. Vous trouverez une copie sur le site <http://www.funix.org/> vous la décompressez en tapant dans un répertoire de travail:

```
tar xvfz tcplogd-0_1_5pre1.tar.gz
```

Cela va vous créer un répertoire **tcplogd-0.1.5**. Préalablement vous devez installer le package **flex**

Vous devez créer un **Makefile** adapté à votre système en tapant:

```
./configure
```

ATTENTION Les exécutables vont se trouver sous **/usr/local/sbin** et **/usr/local/bin**, le fichier de conf dans **/usr/local/etc**. Si ça ne vous convient pas, tapez préalablement

```
configure --help
```

Vous constaterez alors que vous avez le moyen de définir d'autres répertoires d'accueil

Taper maintenant :

```
make
```

Et enfin en tant que root

```
make install
```

Voici les messages qu'on obtient pour cette dernière commande:

```
/usr/bin/install -c -g bin -m 755 -o root -s tcplogd /usr/local/sbin
/usr/bin/install -c -g bin -m 755 -o root -s confcheck /usr/local/bin
Do make cf-install or/and rh-install to install config files or/and redhat startup scripts
separately.
```

A présent on va mettre en place le fichier de config en place en tapant:

make cf-install

Le résultat de la commande est:

```
/usr/bin/install -c -g bin -m 400 -o root tcplogd.cf /usr/local/etc
```

Dans ce fichier vous pouvez spécifier des hôtes amis au niveau de la section **trusted**.

```
trusted {
www.yahoo.com
foo.bar.com
202.202.202.2
}
```

Vous y supprimerez les valeurs par défaut et mettez les adresses IP des hôtes amis.

Maintenant on va faire en sorte que **tcplogd** soit lancé à chaque démarrage en tapant:

make rh-install

Le résultat de la commande est:

```
make install
make[1]: Entering directory &pi0;/alphonse/linux/securite/tcplogd-0.1.5'
/usr/bin/install -c -g bin -m 755 -o root -s tcplogd /usr/local/sbin
/usr/bin/install -c -g bin -m 755 -o root -s confcheck /usr/local/bin
Do make cf-install or/and rh-install to install config files
or/and redhat startup scripts separately.
make[1]: Leaving directory &pi0;/alphonse/linux/securite/tcplogd-0.1.5'
```

./rh_rcd.install

Making symlinks in startup runlevels: 2 3 4 5

symlinking /etc/rc.d/init.d/tcplogd.init /etc/rc.d/rc2.d/S17tcplogd

symlinking /etc/rc.d/init.d/tcplogd.init /etc/rc.d/rc3.d/S17tcplogd

symlinking /etc/rc.d/init.d/tcplogd.init /etc/rc.d/rc4.d/S17tcplogd

symlinking /etc/rc.d/init.d/tcplogd.init /etc/rc.d/rc5.d/S17tcplogd

Making symlinks in stop runlevels: 0 1 6

symlinking /etc/rc.d/init.d/tcplogd.init /etc/rc.d/rc0.d/K58tcplogd

symlinking /etc/rc.d/init.d/tcplogd.init /etc/rc.d/rc1.d/K58tcplogd

symlinking /etc/rc.d/init.d/tcplogd.init /etc/rc.d/rc6.d/K58tcplogd

done.Do not forget to add:

local3.* /var/log/tcp-log

to /etc/syslog.conf

Le fichier de démarrage de **tcplogd**, **tcplogd.init** a été installé sous **/etc/rc.d/init.d** avec les liens suivant pointant vers ce fichier:

/etc/rc.d/rc2.d/S17tcplogd

/etc/rc.d/rc3.d/S17tcplogd

/etc/rc.d/rc4.d/S17tcplogd

/etc/rc.d/rc4.d/S17tcplogd

/etc/rc.d/rc0.d/K58tcplogd

/etc/rc.d/rc1.d/K58tcplogd

/etc/rc.d/rc6.d/K58tcplogd

NOTES - Si vous ne voulez plus que **tcplogd** soit lancé au démarrage il suffira de supprimer liens sous les répertoires **/etc/rc.d/rcX.d**

- Si vous ne voulez pas que **tcplogd** soit lancé au démarrage, pour le lancer ponctuellement il suffira de taper en tant que root

./tcplogd-0.1.5/tcplogd.init start

A présent éditer le fichier **/etc/syslog.conf** et rajouter:

local3.* /var/log/tcp-log

5.2.3 Tests de fonctionnement

Soit le poste **asterix** (IP 192.168.13.10) de votre réseau local qui va faire office d'attaquant et **obelix** (IP 192.168.13.11) votre poste à protéger, **armoric.bz** votre domaine. D'**obelix** on va d'abord lancer **tcplogd**, en tapant:

```
/etc/rc.d/init.d/tcplogd.init start
Starting tcplogd: [ OK ]
```

Checking tcplogger config file [/usr/local/etc/tcplogd.cf]

encounted 0 errors. Scanned configuration is:

ignoring ports: 25 113

logging packets:

type: "Xmas Three" mask: 29 facility: INFO

type: "Syn probe" mask: 2 facility: INFO

type: "StealthFin" mask: 1 facility: INFO

type: "Null probe" mask: 0 facility: INFO

flood limit: 100 flood timeout: 12

trusted hosts

192.168.13.11 [A92.168.13.11]

Maintenant d'**asterix** on va faire un scan des ports TCP d'**obelix** en tapant:

nmap -v obelix

Voici ce qui va s'afficher en temps réel dans le fichier **/var/log/tcp-log**

```
Apr 3 18:57:51 obelix tcplogd: "Syn probe" asterix.armoric.bz[192.168.13.10]:[1220]-
>obelix.armoric.bz[192.168.13.11]:[1421]
```

```
Apr 3 18:57:51 obelix tcplogd: "Syn probe" asterix.armoric.bz[192.168.13.10]:[1221]-
>obelix.armoric.bz[192.168.13.11]:time
```

```
Apr 3 18:57:51 obelix tcplogd: "Syn probe" asterix.armoric.bz[192.168.13.10]:[1222]-
>obelix.armoric.bz[192.168.13.11]:[928]
```

```
Apr 3 18:57:51 obelix tcplogd: "Syn probe" asterix.armoric.bz[192.168.13.10]:[1223]->obelix.armoric.bz[192.168.13.11]:[4133]
Apr 3 18:57:51 obelix tcplogd: "Syn probe" asterix.armoric.bz[192.168.13.10]:[1224]->obelix.armoric.bz[192.168.13.11]:[147]
Apr 3 18:57:51 obelix tcplogd: "Syn probe" asterix.armoric.bz[192.168.13.10]:[1225]->obelix.armoric.bz[192.168.13.11]:[304]
Apr 3 18:57:51 obelix tcplogd: Flood detected
```

A présent d'**asterix** on va voir si y a **SSH** (port 22), **DNS** (port 53), **POP3** (port 110), **IMAP** (port 143) qui tourne et le port 4564 ouvert sur **obelix**.

```
nmap -sX -p 22,53,110,143,4564 obelix
```

Résultat dans `/var/log/tcp-log`

```
Apr 3 19:00:56 obelix tcplogd: "Xmas Three" asterix.armoric.bz[192.168.13.10]:[49686]->obelix.armoric.bz[192.168.13.11]:pop3
Apr 3 19:00:56 obelix tcplogd: "Xmas Three" asterix.armoric.bz[192.168.13.10]:[49686]->obelix.armoric.bz[192.168.13.11]:domain
Apr 3 19:00:56 obelix tcplogd: "Xmas Three" asterix.armoric.bz[192.168.13.10]:[49686]->obelix.armoric.bz[192.168.13.11]:ssh
Apr 3 19:00:56 obelix tcplogd: "Xmas Three" asterix.armoric.bz[192.168.13.10]:[49686]->obelix.armoric.bz[192.168.13.11]:[4564]
Apr 3 19:00:56 obelix tcplogd: "Xmas Three" asterix.armoric.bz[192.168.13.10]:[49686]->obelix.armoric.bz[192.168.13.11]:imap2
Apr 3 19:00:56 obelix tcplogd: "Xmas Three" asterix.armoric.bz[192.168.13.10]:[49687]->obelix.armoric.bz[192.168.13.11]:pop3
Apr 3 19:00:57 obelix tcplogd: "Xmas Three" asterix.armoric.bz[192.168.13.10]:[49686]->obelix.armoric.bz[192.168.13.11]:pop3
Apr 3 19:00:57 obelix tcplogd: "Xmas Three" asterix.armoric.bz[192.168.13.10]:[49687]->obelix.armoric.bz[192.168.13.11]:pop3
```

5.2.4 Utilisation

Etant connecté pour voir si vous êtes attaqués vous devez avoir le fichier **tcp-log** constamment ouvert, pour cela dans un shell indépendant taper:

```
tail -f /var/log/tcp-log
```

Automatiquement dès qu'une nouvelle ligne sera rajoutée au fichier vous la verrez dans le shell.

5.3 Portsentry

5.3.1 Présentation

Portsentry est un outil permettant de détecter les attaques de type scan, en plus de se contenter de détecter les attaques, il peut mener les actions suivantes:

- inscription de l'incident dans le fichier **/var/log/messages** (via **syslogd**)
- l'hôte "attaqueur" est automatiquement rajouté dans **/etc/hosts.deny** (il ne pourra donc plus faire appel aux services validés dans **/etc/inetd.conf**). Pour une Mandrake n'oubliez pas que les **TCP Wrappers**, dont dépendent les fichiers **hosts.deny** et **hosts.allow**, ne sont pas forcément installés sur votre système, veuillez installer le package **tcp_wrappers**.
- on reconfigure **iptables** pour que tout ce qui vient de l'hôte "attaqueur" soit immédiatement rejeté

D'autres part on peut aller plus loin, en coupant automatiquement la connexion par exemple à la moindre attaque.

5.3.2 Installation

On peut récupérer **Portsentry** à l'URL www.psionic.com/abacus/portsentry, visiblement tout a disparu et je n'ai pas trouvé une nouvelle URL. Vous trouverez une copie locale de **portsentry-1.1.tar.gz** à l'URL <http://www.funix.org/>. Une fois l'archive placée dans un répertoire de travail de votre choix, tapez pour la décompresser:

```
tar xvfz portsentry-1.1.tar.gz
```

Un répertoire **portsentry-1.1** va se créer, dans ce répertoire vous trouverez un fichier **Makefile** vous pouvez lui définir le répertoire d'installation de **portsentry**, par défaut il est fixé ainsi

```
INSTALLDIR = /usr/local/psionic
```

Portsentry sera donc installé sous **/usr/local/psionic/portsentry**, si vous voulez que **portsentry** soit installé sous **/usr/local/portsentry**, il suffit de modifier la variable **INSTALLDIR** ainsi:

```
INSTALLDIR=/usr/local
```

C'est pas tout, maintenant éditer **portsentry_config.h** par défaut le fichier de configuration de **portsentry** se trouve sous **/usr/local/psionic/portsentry/portsentry.conf**, si vous voulez qu'il soit sous **/usr/local/portsentry**, vous devez modifier

```
#define CONFIG_FILE "/usr/local/psionic/portsentry/portsentry.conf"
```

Pour lire

```
#define CONFIG_FILE "/usr/local/portsentry/portsentry.conf"
```

On peut à présent lancer la compilation en tapant

```
make linux
```

Voici les messages que vous allez obtenir

```
SYSTYPE=linux
```

```
Making
```

```
cc -O -Wall -DLINUX -DSUPPORT_STEALTH -o ./portsentry ./portsentry.c \  
./portsentry_io.c ./portsentry_util.c
```

Voilà en tant que root vous pouvez passer à l'installation au bon endroit des exécutable, en tapant:

```
make install
```


Creating portsentry directory /usr/local/portsentry

Setting directory permissions

chmod 700 /usr/local/portsentry

Copying files

cp ./portsentry.conf /usr/local/portsentry

cp ./portsentry.ignore /usr/local/portsentry

cp ./portsentry /usr/local/portsentry

Setting permissions

chmod 600 /usr/local/portsentry/portsentry.ignore

chmod 600 /usr/local/portsentry/portsentry.conf

chmod 700 /usr/local/portsentry/portsentry

Edit /usr/local/portsentry/portsentry.conf and change your settings if you haven't already. (route, etc)

WARNING: This version and above now use a new directory structure for storing the program and config files (/usr/local/portsentry). Please make sure you delete the old files when the testing of this install is complete.

C'est bon **portsentry** est installé, passons à la configuration.

5.4 Configuration

Pour cela éditer le fichier de configuration qui se trouve (du moins dans mon exemple) sous **usr/local/portsentry** et qui se nomme [portsentry.conf](#). Les lignes intéressantes ou à modifier sont:

```
#####  
# Configuration Files#  
#####  
#  
# Hosts to ignore  
IGNORE_FILE="/usr/local/portsentry/portsentry.ignore"  
# Hosts that have been denied (running history)  
HISTORY_FILE="/usr/local/portsentry/portsentry.history"  
# Hosts that have been denied this session only (temporary until next restart)  
BLOCKED_FILE="/usr/local/portsentry/portsentry.blocked"
```

A ces endroits là, vous devez veiller à avoir les chemins correspondants.

iptables support for Linux

KILL_ROUTE="/sbin/iptables -I INPUT -s \$TARGET\$ -j DROP"

Tous les paquets venants de l'hôte "attaqueur" (-s source **TARGET**) sont rejetés (**DENY**) et l'ensemble est logué (option **-l**) dans **/var/log/messages**.

```
#####  
# TCP Wrappers#  
#####  
# This text will be dropped into the hosts.deny file for wrappers  
# to use. There are two formats for TCP wrappers:  
#  
# Format One: Old Style - The default when extended host processing  
# options are not enabled.  
#  
KILL_HOSTS_DENY="ALL: $TARGET$"
```

L'hôte attaquateur va être automatiquement rajouté à **/etc/hosts.deny**

```
#####  
# External Command#  
#####  
# This is a command that is run when a host connects, it can be whatever  
# you want it to be (pager, etc.). This command is executed before the  
# route is dropped. I NEVER RECOMMEND YOU PUT IN RETALIATORY  
# ACTIONS  
# AGAINST THE HOST SCANNING YOU. TCP/IP is an *unauthenticated protocol*  
# and people can make scans appear out of thin air. The only time it  
# is reasonably safe (and I *never* think it is reasonable) to run  
# reverse probe scripts is when using the "classic" -tcp mode. This  
# mode requires a full connect and is very hard to spoof.  
#  
#KILL_RUN_CMD="/some/path/here/script $TARGET$ $PORT$"  
KILL_RUN_CMD="/etc/ppp/ppp-off"
```

Ca c'est si vous êtes vraiment parano, à la moindre attaque détectée, on coupe la connexion, l'exemple est donné ici avec une connexion par PPP, pour une connexion type câble via l'interface **eth0**, la ligne suivante devrait faire l'affaire:

KILL_RUN_CMD="/sbin/ifconfig eth0 down"

A présent éditer le fichier **/usr/local/portsentry/portsentry.ignore**, il contient la liste des hôtes "amis" et qui ne seront pas considérés comme des attaquants, il contient par défaut les lignes suivantes qu'il ne faut pas modifier:

127.0.0.1/32
0.0.0.0

Pour rajouter un hôte "ami", il suffit de rajouter son adresse IP à la suite des deux par défaut. Et pour le sous réseau 192.168.13.X

192.168.13.0/24

5.4.1 Les modes de fonctionnement

Plusieurs modes de lancement existe:

-tcp - Basic port-bound TCP mode

Portsentry va vérifier les scans sur les ports TCP et les inscrire dans **/var/log/messages** via **syslog**

-udp - Basic port-bound UDP mode

Idem mais pour les ports UDP

Mais les modes les plus intéressants sont

-audp - Advanced UDP "stealth" scan detection mode

-atcp - Advanced TCP stealth scan detection mode

-sudp "stealth" UDP scan detection

Qui vont permettre une détection active des ports TCP et/ou UDP avec blocage des ports. Pour plus de détails sur ces modes, se reporter aux fichiers **README** que vous trouverez dans le répertoire **./portsentry_1.0**.

5.4.2 Tests de fonctionnement

Un poste de votre réseau va jouer l'attaquant on va utiliser un outil comme **nmap** pour scruter les ports, il faut évidemment que votre hôte attaquateur ne soit pas dans le fichier **portsentry.ignore**. Pour les besoins de l'exemple, l'hôte "attaqueur" est appelé **asterix** (IP 192.168.13.10) et l'hôte à protéger est nommé **obelix** (IP 192.168.13.11) votre domaine est **armoric.bz**.

On va tout d'abord lancer **portsentry** sur **obelix** en tapant:

```
/usr/local/portsentry -atcp
```

Voici ce que vous pouvez voir dans **/var/log/messages**:

```
Feb 22 18:52:39 asterix portsentry[5673]: adminalert: Psionic PortSentry 1.1 is
starting.
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced mode will monitor
first 1024 ports
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced mode will manually
exclude port: 113
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced mode will manually
exclude port: 139
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection
mode activated. Ignored TCP port: 25
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection
mode activated. Ignored TCP port: 53
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection
mode activated. Ignored TCP port: 80
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection
mode activated. Ignored TCP port: 110
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection
mode activated. Ignored TCP port: 111
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection
mode activated. Ignored TCP port: 119
```

(...)

Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection mode activated. Ignored TCP port: 932
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection mode activated. Ignored TCP port: 953
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection mode activated. Ignored TCP port: 113
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: Advanced Stealth scan detection mode activated. Ignored TCP port: 139
Feb 22 18:52:39 asterix portsentry[5674]: adminalert: PortSentry is now active and listening.

Du poste "attaqueur", tapez maintenant:

nmap -v obelix

Voici le résultat au niveau d'obelix dans **/var/log/messages**:

Détection du scan

Feb 22 18:53:03 asterix portsentry[5674]: attackalert: TCP SYN/Normal scan from host: asterix.kervao.fr/192.168.13.10 to TCP port: 794

Rajout de l'attaquant dans le fichier **/etc/hosts.deny**

Feb 22 18:53:03 asterix portsentry[5674]: attackalert: Host 192.168.13.10 has been blocked via wrappers with string: "ALL: 192.168.13.10"

Rejet de tous les paquets de l'attaquant

Feb 22 18:53:03 asterix portsentry[5674]: attackalert: Host 192.168.13.10 has been blocked via dropped route using command: "/sbin/iptables -I INPUT -s 192.168.13.10 -j DROP"

Pas de problème pour les autres scans venant de l'attaquant ils seront tous bloqués

Feb 22 18:53:03 asterix portsentry[5674]: attackalert: TCP SYN/Normal scan from host: asterix.kervao.fr/192.168.13.10 to TCP port: 550

Feb 22 18:53:03 asterix portsentry[5674]: attackalert: Host: asterix.kervao.fr/192.168.13.10 is already blocked Ignoring

(...)

Feb 22 18:53:03 asterix portsentry[5674]: attackalert: Host: asterix.kervao.fr/192.168.13.10 is already blocked Ignoring

Feb 22 18:53:03 asterix portsentry[5674]: attackalert: TCP SYN/Normal scan from host: asterix.kervao.fr/192.168.13.10 to TCP port: 444

Vous constaterez maintenant qu'asterix a été rajouté dans **/etc/hosts.deny** et ne peut donc plus accéder aux services du super serveur **inetd**.

ATTENTION N'oubliez pas de modifier **/etc/hosts.deny** pour que votre poste asterix puisse à nouveau bénéficier des services d'obelix.

5.4.3 Lancement automatique de portsentry

Pour que **Portsentry** soit lancé automatiquement, on rajoutera tout à la fin du fichier **/etc/rc.d/rc.local** les lignes suivantes:

```
/usr/local/portsentry/portsentry -atcp  
/usr/local/portsentry/portsentry -audp
```

Vous pouvez aussi bien placer ces lignes à la fin du fichier **/etc/rc.d/rc.firewall**

5.5 Iplog

5.5.1 Présentation

iplog est un outil qui va archiver dans un fichier log le trafic TCP/IP. Concrètement il détecte les protocoles TCP, UDP et ICMP, mais il est très facile de rajouter d'autres protocoles. Il va vous permettre de détecter les attaques en tout genre, du style scan. Il est basé sur **libpcap** qu'on installera à partir des packages de la Mandrake ou avec le tarball (chapitre 6.2).

5.5.2 Installation

Le site officiel d'**iplog** est <http://ojnk.sourceforge.net> on y récupérera l'archive qu'on décompressera en tapant

```
tar xvfz iplog-2.2.3.tar.gz
```

Cela donne le répertoire **iplog-2.2.3** dans lequel on tape successivement

```
./configure  
make
```

Puis en tant que root

```
make install
```

5.5.3 Configuration

Il existe un fichier de conf **/etc/iplog.conf** qui est non existant par défaut. Sa syntaxe très simple est donnée par la commande

```
man iplog.conf
```

La syntaxe est du style

```
set mot-clé true| false
```

Exemple

```
set tcp false
```

Pour ne pas archiver le trafic tcp. La liste des mots clé et le reste de la syntaxe est dans le man.

5.5.4 Utilisation

Pour avoir la liste des options disponibles, il suffit de taper

iplog -h

Vous devez taper **man iplog** pour voir la liste des options activées par défaut, vous constaterez qu'elles sont très nombreuses et qu'on peut se contenter de lancer **iplog** sans options. Toutefois en voilà deux intéressantes

-i sélection de l'interface, par défaut il écoute toutes les interfaces "up" sauf loopback
-l nom du fichier de log, sinon tout est archivé dans **/var/log/messages**

Voilà un extrait du résultat d'un scan lancé sur asterix par obelix au moyen de **nmap**

```
Oct 31 21:56:30 asterix iplog[27350]: TCP: at-nbp connection attempt from
obelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: port 884 connection attempt from
obelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: port 108 connection attempt from
obelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: port 3001 connection attempt from
obelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: port 1384 connection attempt from
obelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: port 10000 connection attempt from
obelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: port 13714 connection attempt from
obelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: port 170 connection attempt from
obelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: port 1364 connection attempt from
aobelix.breizland.bz:56160
Oct 31 21:56:30 asterix iplog[27350]: TCP: SYN scan detected [ports
202,884,108,3001,1384,10000,13714,170,1364,9876,...] from obelix.breizland.bz[port
56160]
Oct 31 21:57:08 asterix iplog[27350]: ICMP: echo from obelix.breizland.bz (8 bytes)
Oct 31 21:57:11 asterix iplog[27350]: TCP: Bogus TCP flags set by
obelix.breizland.bz:52799 (dest port 80)
Oct 31 21:57:11 asterix iplog[27350]: UDP: dgram to tcpmux from
obelix.breizland.bz:52792 (300 data bytes)
```


6 "Sniffer" son réseau avec Ethereal et Snort

6.1 Présentation

Maintenant pour tester la « robustesse » de votre réseau vous pouvez tenter de le sniffer pour voir si les mots de passe ne circulent pas en clair par exemple. C'est l'objet de ce chapitre.

Ethereal est un analyseur de trafic réseau, ou "sniffer". Il utilise une interface graphique basée sur **GTK+**, il est basé sur la bibliothèque **libpcap**, qui fournit des outils pour capturer les paquets réseau. La page d'accueil d'**Ethereal** est www.ethereal.com

6.2 Libpcap

6.2.1 Présentation

libpcap est une bibliothèque d'outils permettant de faire la capture des paquets qui circulent sur le réseau, on peut ainsi faire des stats, de la surveillance de réseau, du débogage et bien d'autres choses.

6.2.2 Installation

Vous avez le choix entre installer le package fourni avec la Mandrake ou d'installer le tarball qu'on trouvera sur le site www.tcpdump.org . L'archive est le tarball **libpcap-0.8.1.tar.gz** qu'on décompressera en tapant :

```
tar xvfz libpcap-0.8.1.tar.gz
```

Cela va nous donner le répertoire **libpcap-0.8.1**, dans ce répertoire on tapera pour créer le **Makefile** :

```
./configure
```

A présent tapons :

```
make
```

NOTE : Les packages suivants sont nécessaires sur une Mandrake **byacc** et **flex**

Puis en tant que **root**

make install

Si vous avez choisi d'installer avec le package RPM, pour la suite des opérations il vous faudra aussi installer le package de développement **libpcap0-devel**

6.3 Ethereal

6.3.1 Installation

On récupère l'archive **ethereal-0.10.0a.tar.gz** qu'on trouvera à l'URL www.ethereal.com, qu'on décompressera en tapant :

tar xvfz ethereal-0.10.0a.tar.gz

Cela va nous créer un répertoire **ethereal-0.10.0a**. Avant d'aller plus loin, vous devez vous assurer que la bibliothèque **libpcap** est bien installée. On tape d'abord :

./configure --with-pcap=/usr/local/linux/libpcap-0.8.1/

On spécifie ici l'emplacement (en absolu, à changer chez vous éventuellement) de **libpcap** installé précédemment au cas où vous avez une version antérieure de **libpcap** préinstallé sur votre système. Voilà le résultat de la commande :

The Ethereal package has been configured with the following options.

Build ethereal : yes

Build tethereal : yes

Build editcap : yes

Build mergecap : yes

Build text2pcap : yes

Build idl2eth : yes

Build randpkt : no

Build dfptest : no

Install setuid : no

Use plugins : yes

Use GTK+ v2 library : no

Use pcap library : yes

Use zlib library : yes
Use pcre library : no
Use GNU ADNS library : no
Use IPv6 name resolution : yes
Use UCD SNMP/NET-SNMP library : no

On tape maintenant

make

En tant que **root**, tapez maintenant:

make install

Editer le fichier **/etc/ld.so.conf** et rajoutez la ligne

/usr/local/lib/ethereal/plugins/0.10.0a

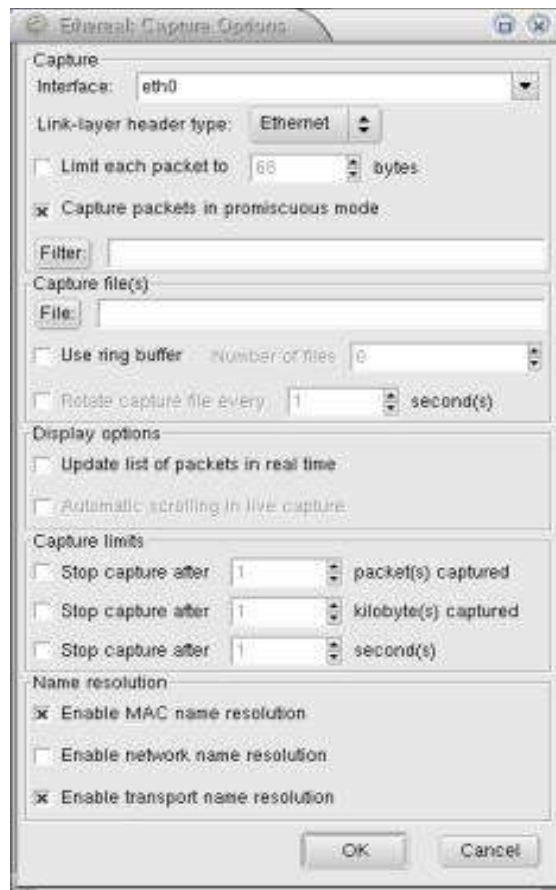
Pour que le fichier soit pris en compte tapez maintenant

ldconfig

Un répertoire contenant divers fichiers (plugins et autres) a été installé sous **/usr/local/lib/ethereal**, et les exécutable se trouvent sous **/usr/local/bin**.

6.3.2 Utilisation d'Ethereal

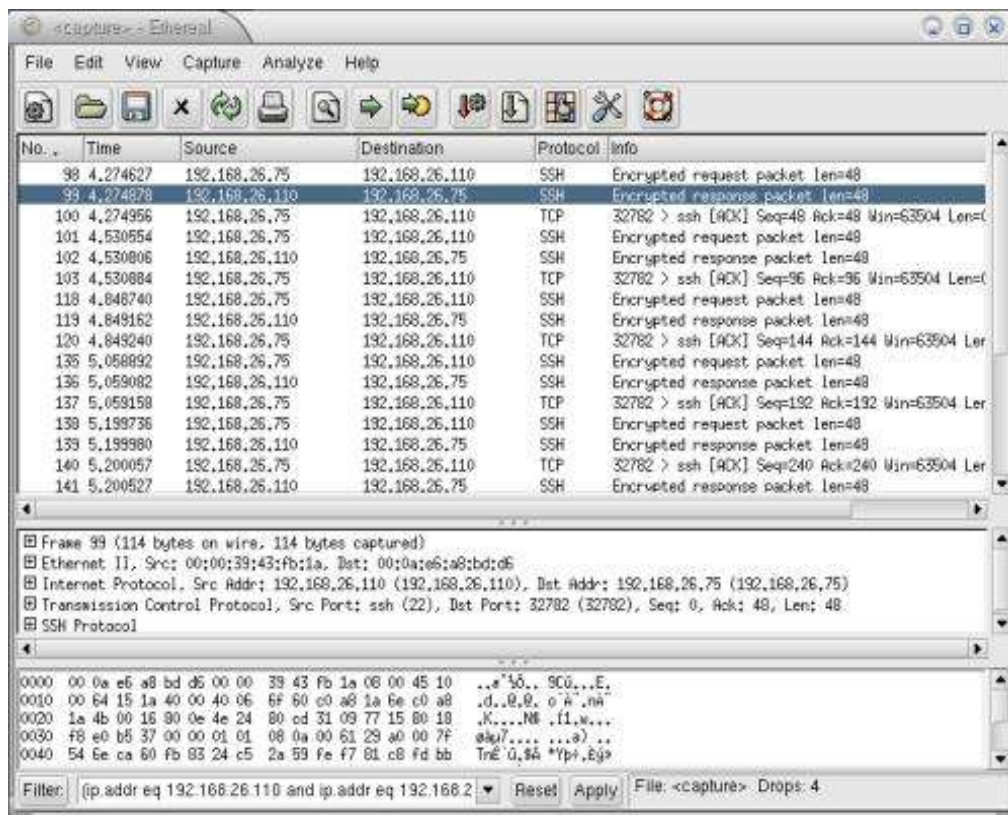
Vous devez être **root** pour pouvoir utiliser **ethereal**. Au lancement la fenêtre principale apparaît, cliquez sur **Capture** dans la barre de menu, puis **Start**, la fenêtre suivante apparaît :



eth0 correspond à l'interface réseau à observer, appuyez sur OK. Une fenêtre comptant le nombre de paquets capturés avec le protocole utilisé va apparaître :



Le trafic est maintenant capturé, par exemple ici on a 2066 paquets de type TCP. Cliquez sur **Stop** pour cesser la capture, toutes les informations sur les paquets capturés apparaissent au niveau de la fenêtre principale.



On observe ici une session SSH entre le client 192.168.26.75 et le serveur 192.168.26.110. Dans la partie centrale, on a des informations complémentaires (protocole utilisé, numéro du port...), dans la partie du bas on a le contenu du paquet qui a été sélectionné dans la partie du haut.

Dans **Analyze -> Follow TCP Stream**, on pourra reconstituer à partir des paquets capturés une session. Voilà un exemple avec une session **telnet** :



On y voit le mot de passe en clair !! (depuis j'ai changé le mot de passe ;-))

Vous voyez ici toute la puissance d'**ethereal**, je vous laisse découvrir les autres fonctionnalités du logiciel et découvrir toutes les faiblesses de votre réseau (**telnet**, **pop**, ...).

6.4 snort

6.4.1 Présentation

Snort est aussi un sniffer de réseau, à la différence d'**Ethereal**, il agit en temps réel. Le site officiel de snort est <http://www.snort.org>.

6.4.2 Installation

On doit préalablement installer **pcre** qui est une bibliothèque fourissant des outils gérant des expressions régulières compatibles avec Perl. L'URL officiel <http://www.pcre.org/> on y récupère l'archive qu'on décompresse en tapant:

```
tar xvfz pcre-4.5.tar.gz
```

Cela donne le répertoire **pcre-4.5** dans lequel on tape successivement

```
./configure  
make
```

Puis en tant que root

```
make install  
ldconfig
```

Ensuite on décompresse l'archive de **snort** en tapant. On peut récupérer sur le site de **snort** l'archive **snort-2.1.0.tar.gz** qu'on décompressera en tapant :

```
tar xvfz snort-2.1.0.tar.gz
```

Cela va nous créer un répertoire **snort-1.9.1**. Avant d'aller plus loin, vous devez vous assurer que la bibliothèque **libpcap** (chapitre 6.2) est bien installée. On tape ensuite successivement :

```
./configure --with-libpcap-includes=/usr/local/linux/libpcap-0.8.1/ --with-libpcap-libraries=/usr/local/lib
```

On spécifie ici l'emplacement (en absolu, à changer chez vous éventuellement) de **libpcap** installé précédemment au cas où vous avez une version antérieure de **libpcap** préinstallé sur votre système. On tape ensuite:

make

NOTE Les logs peuvent être archivés dans une base de données, par défaut si MySQL est installé sur votre système et que vous ne voulez pas bénéficier de cette option ajoutez l'option **--without-mysql** à **configure**

Puis en tant que **root**

make install

Les exécutables seront placés sous **/usr/local/bin**

6.4.3 Syntaxe

La syntaxe de la commande est la suivante :

snort -options expression

Les options disponibles sont les suivantes:

-b les paquets sont logués dans un fichier au format **tcpdump** appelé **snort.log**. C'est l'option qu'il faut prendre si on ne veut pas que **snort** perdre du temps à faire la conversion binaire->ASCII pour ne rater aucun paquets, l'option **-r** permet de relire ces fichiers en temps différé.

-c <cf> on utilise le fichier de règles **<cf>**, ce fichier indique ce que le système doit logger.

-h <hn> on définit ici l'adresse du réseau local, cela sert uniquement pour le formattage du texte pour mettre la flèche qui va bien en fonction du sens du trafic par rapport au réseau (entrant ou sortant)

-i <if> on utilise l'interface **<if>**, par défaut eth0

-s Les alertes sont archivées au travers de **syslog** dans **/var/log/secure**.

-l log-dir pour définir le répertoire d'archivage pour les logs. Par défaut le répertoire est **/var/log/snort**

-d Pour extraire uniquement la couche transport (layer) du paquet

-v mode verbeux, les paquets apparaissent dans la console ou le shell à partir duquel a été lancée la commande, contrairement à l'option **-b** ça ralentit considérablement le fonctionnement de **snort** du coup on peut perdre des paquets, à déconseiller donc, si vous ne voulez pas perdre une miette des échanges de paquets

L'expression fixe les critères pour les paquets qui seront logués. Si aucune expression n'est donnée, tous les paquets seront logués. Une expression consiste en un ou plusieurs primitives, une primitive consiste en une identité (nom ou adresse) précédée par un ou plusieurs qualificatifs, dont il existe trois types:

- type: pour définir le type précis de l'identité, on a le choix entre **host** pour une machine, **net** pour un réseau et **port** pour un port, par défaut on utilise le type **host**.

Exemple: **host www.breizland.bz** pour l'hôte du même nom, ou encore **net 192.168.13** pour un réseau du type 192.168.13.X

- dir: pour définir la direction à partir ou vers l'identité, les directions possibles sont **src**, pour en provenance de, et **dst**, pour à destination de. Par défaut on prend les paquets dans les deux sens (**src** et **dst**).

Exemple: **src www.breizland.bz** les paquets provenant de l'hôte **www.breizland.bz**, ou encore **dst port 20** les paquets à destination du ports 20 (**ftp**).

- proto: pour restreindre les paquets utilisant un protocole particulier, les protocoles possibles sont : **ether**, **fddi**, **ip**, **arp**, **rarp**, **decnet**, **lat**, **sca**, **moprc**, **mopdl**, **tcp** et **udp**. Si aucun protocole n'est spécifié on capture les paquets quel que soit le protocole utilisé.

Exemple: **ether src www.breizland.bz** les paquets utilisant **ethernet** provenant de **www.breizland.bz**, ou encore **tcp port 21**, les paquets allant ou venant du port 21 et utilisant **TCP**.

6.4.4 Utilisation

Commençons par la base, on va afficher à la console tous les entêtes de paquets :

snort -v

Running in packet dump mode

Log directory = /var/log/snort

Initializing Network Interface eth0

--== Initializing Snort ==--

Initializing Output Plugins!

Decoding Ethernet on interface eth0

--== Initialization Complete ==--

.*> Snort! <*-

Version 2.1.0 (Build 9)

By Martin Roesch (roesch@sourcefire.com, www.snort.org)

10/21-08:23:16.645501 192.168.13.15 -> 192.168.13.11

ICMP TTL:64 TOS:0x0 ID:12

ID:691 Seq:12 ECHO

+++++
+

10/21-08:23:16.645610 192.168.13.11 -> 192.168.13.15

ICMP TTL:255 TOS:0x0 ID:100

ID:691 Seq:12 ECHO REPLY

+++++
+±

Concrètement on voit ici un **ping** (demande d'écho, puis écho)

En faisant un CTRL-C pour stopper la commande, on obtient le bilan suivant sur **eth0**

=====
=====

Snort analyzed 61 out of 79 packets, The kernel dropped 0(0.000%) packets

Breakdown by protocol:

TCP: 0 (0.000%)
UDP: 49 (62.025%)
ICMP: 6 (7.595%)
ARP: 6 (7.595%)
IPv6: 0 (0.000%)

Action Stats:

ALERTS: 0
LOGGED: 0
PASSED: 0

IPX: 0 (0.000%)
OTHER: 0 (0.000%)
DISCARD: 0 (0.000%)

=====
=====

Fragmentation Stats:

Fragmented IP Packets: 18 (22.785%)

Fragment Trackers: 0

Rebuilt IP Packets: 0

Frag elements used: 0

Discarded(incomplete): 0

Discarded(timeout): 0

Frag2 memory faults: 0

=====
=====

TCP Stream Reassembly Stats:

TCP Packets Used: 0 (0.000%)

Stream Trackers: 0

Stream flushes: 0

Segments used: 0

Stream4 Memory Faults: 0

=====
=====

Snort received signal 2, exiting

Si vous voulez voir le contenu du paquet en plus de l'entête, vous pouvez taper:

snort -vd

Running in packet dump mode

Log directory = /var/log/snort

Initializing Network Interface eth0

--== Initializing Snort ==--

Initializing Output Plugins!

Decoding Ethernet on interface eth0

--== Initialization Complete ==--

-*> Snort! <*-

By Martin Roesch (roesch@sourcefire.com, www.snort.org)

```
F1 82 87 22 00 00 00 00 00 00 00 02 00 01 86 A3 ...".....
00 00 00 03 00 00 00 03 00 00 00 01 00 00 00 2C .....
00 00 12 34 00 00 00 11 73 61 62 6C 69 6E 65 2E ...4....obelix.
6B 65 72 76 61 6F 2E 66 72 00 00 00 00 00 13 89 breizland.bz.....
00 00 13 88 00 00 00 01 00 00 13 88 00 00 00 00 .....
00 00 00 00 00 00 00 0C 01 00 00 00 00 16 00 07 .....
02 00 00 00 00 00 00 08 68 6F 6D 65 70 61 67 65 .....homepage
```

$\begin{array}{ccc} + & + & + & + & + & + & + & + & + & + & - & - & + \\ + & + & + & + & + & + & + & + & \end{array}$

```
F1 82 87 22 00 00 00 01 00 00 00 00 00 00 00 ...".
00 00 00 00 00 00 00 00 00 00 00 00 00 00 14 .....
01 00 00 01 00 16 00 07 02 00 00 00 41 A8 04 00 .....A...
25 D3 91 A3 00 00 00 01 00 00 00 02 00 00 41 ED %.....A.
00 00 00 09 00 00 13 89 00 00 13 88 00 00 00 00 .....
00 00 10 00 00 00 00 00 00 00 10 00 00 00 00 .....
00 00 00 00 00 00 00 00 00 00 16 07 00 00 00 .....
00 04 A8 41 3D 72 C4 14 00 00 00 00 3D 30 FC 88 ...A=r.....=0..
00 00 00 00 3D 30 FC 88 00 00 00 00 00 00 00 01 ...=0.....
00 00 00 02 00 00 41 FF 00 00 00 11 00 00 00 00 .....A.....
00 00 00 00 00 00 00 00 00 00 10 00 00 00 00 .....
00 00 10 00 00 00 00 00 00 00 00 00 00 00 00 .....
00 00 16 07 00 00 00 00 00 00 02 3D 72 C5 88 .....=r..
00 00 00 00 3D 57 38 52 00 00 00 00 3D 57 38 52 ...=W8R....=W8R
00 00 00 00 .....
```

[illegible]

<http://www.funix.org>

snort -vde

Maintenant on va loguer les paquets en tapant

snort -vde -l /var/log/snort

Attention le répertoire **/var/log/snort** doit être créé préalablement. Ainsi en tapant cette commande pour la machine d'adresse 192.168.13.15 après avoir fait un **ping** et un **telnet** sur la machine 192.168.13.11, on trouvera dans le répertoire **/var/log/snort**, les répertoires suivants 192.168.13.11 et 192.168.13.15, le premier contenant le fichier **ICMP_ECHO_REPLY** et le deuxième **ICMP_ECHO**, TCP:1025-23. Ces fichiers contenant les paquets résultants du **ping** et du **telnet**.

Pour spécifier un fichier de règles vous taperez :

snort -vde -l /var/log/snort -c snort-lib

Vous trouverez un exemple de fichier de règles sous le répertoire de **snort**, et un certain nombre sur le site même de **snort**.

Maintenant si vous voulez exploiter en temps différé un fichier binaire du format **tcpdump**, vous taperez:

snort -vde -l /var/log/snort -r tcpdump_file

Pour afficher les alertes dans **/var/log/secure** au moyen de **syslog** en utilisant un fichier de règles on tapera:

snort -vde -l /var/log/snort -c snort-lib -s

Pour regarder maintenant un peu les paquets qui circulent lors d'une connexion **PPP**, on peut taper (l'adresse devant **host** correspond à l'adresse IP attribuée par le FAI) :

snort -h 192.168.13.0/24 -d -v host 213.228.15.14 -i ppp0

```

=====
+
10/16-21:31:09.048632 194.158.97.244:110 -> 213.36.44.26:1292
TCP TTL:59 TOS:0x0 ID:62404 DF
*****PA* Seq: 0xDCC4E1BD Ack: 0xF1084599 Win: 0x8218
TCP Options => NOP NOP TS: 240412927 876595
2B 4F 4B 20 50 4F 50 33 20 73 65 72 76 65 72 20 +OK POP3 server
4D 65 64 69 61 6E 65 74 2F 31 2E 31 33 20 3C 32 Medianet/1.13 <2
32 37 38 34 2E 39 37 31 37 32 34 36 36 37 40 6D 2784.971724667@m
65 64 69 61 6E 65 74 2D 31 76 2E 67 72 6F 6C 69 edianet-1v.groli
65 72 2E 66 72 3E 0D 0A er.fr>..

```

```

=====
+
10/16-21:31:09.058988 213.36.44.26:1292 -> 194.158.97.244:110
TCP TTL:64 TOS:0x0 ID:4595  DF
*****PA* Seq: 0xF1084599  Ack: 0xDCC4E205  Win: 0x7F88
TCP Options => NOP NOP TS: 876617 240412927
55 53 45 52 20 6F 6C 69 76 69 65 72 2E 68 6F 61  USER mon-login

```

**+=====+
+
10/16-21:31:09.218671 194.158.97.244:110 -> 213.36.44.26:1292
TCP TTL:59 TOS:0x0 ID:62406 DF**

```

=====
+
10/16-21:31:09.218898 213.36.44.26:1292 -> 194.158.97.244:110
TCP TTL:64 TOS:0x0 ID:4596 DF
*****PA* Seq: 0xF10845AE Ack: 0xDCC4E22F Win: 0x7F88
TCP Options => NOP NOP TS: 876633 240412944
50 41 53 53 20 73 61 78 6F 32 37 30 0D 0A    PASS mot-de-passe-en-clair..

```

```

=====
+
10/16-21:31:09.638641 194.158.97.244:110 -> 213.36.44.26:1292
TCP TTL:59 TOS:0x0 ID:62408 DF
*****PA* Seq: 0xDCC4E22F Ack: 0xF10845BC Win: 0x8218
TCP Options => NOP NOP TS: 240412987 876633
2B 4F 4B 20 6F 6C 69 76 69 65 72 2E 68 6F 61 72 +OK mon-login
61 75 20 68 61 73 20 30 20 6D 65 73 73 61 67 65 has 0 message
73 20 28 30 20 6F 63 74 65 74 73 29 0D 0A      s (0 octets)..

```

<http://www.funix.org>