

Sommaire

XINETD sous toutes ses coutures.....	1
Avant propos.....	1
Inetd ou Xinetd ?.....	1
Définition.....	1
Ce qu'apporte xinetd.....	1
Passer de inetd à xinetd.....	1
configuration générale de xinetd.....	2
L'arborescence de xinetd.....	2
Syntaxe générale d'un fichier de configuration.....	2
Écriture des fichiers : attributs obligatoires.....	3
Affiner les logs avec xinetd.....	4
Localisation des logs.....	4
Contenu des logs.....	4
Xinetd pour contrôler les accès à votre machine.....	5
Contrôler l'origine des accès.....	5
Contrôler le moment des accès.....	6
Contrôler l'exposition du système pendant l'accès : chroot.....	6
Autoriser / interdire un service.....	7
Xinetd pour limiter les attaques de type Deny of Service.....	7
Autres fonctionnalités de xinetd.....	8
Redirection de ports.....	8
Attribution d'un service à une adresse IP.....	8
xinetd parle à vos visiteurs.....	9
Exemples de configuration.....	9
Exemple 1.....	9
Exemple 2.....	10
Exemple 3.....	10
Le mot de la fin.....	11
Copyright.....	12

XINETD sous toutes ses coutures

XINETD sous toutes ses coutures

par Anne

Où comment utiliser xinetd pour sécuriser encore plus votre pingouin préféré

Avant propos

Cet article a pour objectif d'expliquer le fonctionnement et la configuration du super démon `xinetd`. Pourquoi particulièrement celui-ci ? Parce qu'il est utilisé fréquemment pour l'accès à nombre de services réseau et devient un outil non négligeable de sécurisation de celui-ci (en plus d'un bon firewall bien sûr). Cet article n'a pas pour objectif d'être complètement exhaustif mais de présenter les configurations les plus courantes de `xinetd`. Pour plus d'infos, consulter le man (on s'en serait douté ;))

Inetd ou Xinetd ?

Selon la distribution vous trouverez soit un démon `inetd`, soit un démon `xinetd`. La tendance est tout de même d'utiliser de plus en plus ce dernier.

Définition

Dans l'absolu, `inetd` et `xinetd` ont le même rôle, à savoir de piloter l'accès à un ou plusieurs services réseaux. Ils agissent comme une standardiste. Ils reçoivent des requêtes de clients, extérieurs pour la plupart, qui demandent un accès à un service réseau déterminé (ex : ftp, telnet, ssh...). Le super démon va, en fonction des instructions qu'on lui aura données (fichiers de configuration) transmettre ou rejeter l'appel.

Ce qu'apporte xinetd

`inetd`, jusque là utilisé, permettait, grâce au fichier `/etc/inetd.conf` et au wrapper `tcpd`, de paramétrer l'accès aux services en l'autorisant/interdisant totalement ou partiellement. (cf. les fichiers `/etc/hosts.allow` et `/etc/hosts.deny`).

`xinetd` apporte des fonctionnalités bien plus importantes et permet d'affiner les paramétrages d'accès aux services. On citera dans le désordre :

- possibilité d'affiner les logs des services gérés
- paramétrage d'accès par service et non global
- paramétrage des plages horaires de disponibilité des services
- possibilité de chrooter les services (ex : ftp)
- possibilité de limiter les attaques de type deny of service (contrôle de la priorité d'un serveur, contrôle de la charge CPU, contrôle du nombre de connexions par service, ...)
- redirection de ports

Passer de inetd à xinetd

Si votre distribution préférée utilise `inetd` et qu'après avoir eu le courage de lire cet article, vous souhaitez utiliser `xinetd`, il existe un script qui vous permettra de transformer le fichier `inetd.conf` en fichier `xinetd.conf`, utilisable par `xinetd`

Ce script, `xconv.pl`, est un script perl fourni avec `xinetd`. Attention si ce script peut vous donner la structure générale du fichier de configuration, il ne vous permettra pas de profiter des apports de `xinetd`. Rien de tel qu'un bon éditeur de texte et ce qui suit ci-dessous.

configuration générale de xinetd

On entre dans le vif du sujet :). Pour configurer `xinetd`, vous aurez à connaître la syntaxe, commune de `/etc/xinetd.conf` et, selon les cas de figure, les fichiers situés dans le répertoire `/etc/xinetd.d`.

L'arborescence de xinetd

L'arborescence de la configuration de `xinetd` est relativement simple. On en rencontre 2 types :

- **un seul fichier de configuration** : `/etc/xinetd.conf` qui comprendra la configuration générale de `xinetd` **et** la configuration des services gérés par `xinetd`. (exemple plus loin dans l'article)
- un fichier réservé à la **configuration générale** de `xinetd`, nommé aussi `/etc/xinetd.conf`. La **configuration des services** est déportée dans des fichiers situés dans le répertoire `/etc/xinetd.d`. Ce répertoire comprend un fichier de configuration par service géré par `xinetd`. Le fichier porte le nom du service.

Pour utiliser ce deuxième cas de figure, le fichier `/etc/xinetd.conf` doit contenir la ligne suivante :

```
includedir /etc/xinetd.d
```

C'est ce deuxième cas de figure qui est le plus couramment utilisé dans les distributions.

Syntaxe générale d'un fichier de configuration

Le fichier de configuration de `xinetd` est un ensemble d'une ou plusieurs directives dont la syntaxe est la suivante :

```
service nom_du_service
{
    ...
}
```

Le nom de la directive est soit `defaults` et la configuration porte alors sur l'ensemble des services gérés par `xinetd`, soit le nom d'un des services géré par `xinetd`. A l'intérieur de chacune de ces directives on trouvera des attributs, un par ligne, écrits de la manière suivante :

```
<attribut> <assignement> <valeur> <valeur> ...
```

Les attributs seront détaillés plus loin. Les assignements peuvent prendre différentes valeurs : `=`, `-=`, `+=`.

Exemple :

```
root@pingu# cat /etc/xinetd.d/telnet
service telnet
{
    ...
    server = /usr/sbin/in.telnetd
    ...
}
```

avec :

attribut : server

assignement : "="

valeur : /usr/sbin/in.telnetd

Pour reprendre l'arborescence de xinetd et les 2 cas de figure exposés ci-dessus, on obtiendra les fichiers suivants :

un seul fichier de configuration	un fichier de configuration par service
<pre>/etc/xinetd.conf defaults { ... } service service1 { ... } service service2 { ... }</pre>	<pre>/etc/xinetd.conf defaults { ... } /etc/xinetd.d/service1 service service1 { ... } /etc/xinetd.d/service2 service service2 { ... }</pre>

Écriture des fichiers : attributs obligatoires

Nous allons lister les différents attributs utilisables pour configurer xinetd. Certains sont facultatifs et vous permettent d'affiner son rôle. Toutefois d'autres sont obligatoires et s'ils ne sont pas présents, empêcheront tout ou partie des services de fonctionner.

Avant de les lister, petit point de vocabulaire : on distingue les services dits internes et externes. A quoi ? A xinetd bien sur :). Les services internes, comme `servers`, `services`, `xadmin` sont des services propres à xinetd qui fournissent des informations sur le fonctionnement du super-démon. Il vaut mieux ne pas utiliser ces services car ils exposent la machine inutilement et les fichiers de log de xinetd vous fourniront les mêmes informations.

Voilà donc les attributs que vous devrez utiliser :

Attribut	Définition
socket-type	type de socket utilisé pour le service : <code>dgram</code> s'il utilise le protocole UDP, <code>stream</code> s'il utilise le protocole TCP – consulter le fichier <code>/etc/services</code> pour avoir l'information.
user	identité sous laquelle le service sera lancé
server	chemin et nom du serveur
wait	Définit le comportement du service dans le traitement des threads : <code>yes</code> pour un service mono-thread (une connexion simultanée par service et une seule), <code>no</code> pour un service

Xinetd

	multithread (possibilité d'avoir plusieurs connexions simultanées au service)
protocol	Protocole utilisé par le service. Si rien n'est précisé, c'est le protocole spécifié dans le fichier <code>/etc/services</code> qui sera utilisé.
rpc_version rpc_number	Ne concerne que les services basés sur les RPC (exemple : NFS)
port	Port associé au service. Là encore, s'il n'est pas précisé, c'est le port spécifié pour le service dans le fichier <code>/etc/services</code> .

Affiner les logs avec xinetd

Une des fonctionnalités de `xinetd` est de permettre d'affiner ce que vous voulez loguer et dans quel(s) fichier(s) vous voulez le loguer.

Localisation des logs

C'est l'attribut `log_type` qui va donner cette localisation. Il peut prendre 2 valeurs :

- **SYSLOG** `syslog_facility [syslog level]` : les logs seront alors gérés par le démon `syslogd`. Vous pourrez préciser le *niveau* de log (voir man `syslog`) : avec les valeurs `emerg`, `alert`, `crit`, `err`, `warning`, `notice`, `info`, `debug`. L'ordre de ces valeurs indique une quantité croissante d'informations récupérées par `syslogd`. Par défaut, c'est le niveau `info` qui est utilisé. Vous retrouverez ensuite les logs généralement dans le fichier `/var/log/messages` (cf. `/etc/syslog.conf`).
Inconvénient de cette méthode : toutes les informations sont stockées dans un seul fichier et difficiles à lire étant donné la quantité d'éléments recueillis pour le noyau.
- **FILE** `file [soft limit [hard limit]]` : vous allez pouvoir complètement configurer la destination de vos logs et dans l'absolu, prévoir un fichier de log par service géré par `xinetd`. Le mot-clé `FILE` sera suivi du nom du fichier. Si celui-ci n'existe pas, il sera créé.
De manière facultative, vous pouvez également préciser 2 types de limites à la taille du fichier de log : une *limite soft* qui lorsqu'elle est franchie provoque l'envoi d'un message d'alerte (mais les logs continuent d'alimenter le fichier) et une *limite hard*, qui lorsqu'elle est franchie bloque l'envoi de logs supplémentaires dans le fichier concerné. L'utilisation de ces deux limites est recommandée pour éviter de saturer le système de fichiers.
Les limites sont à donner en octets (1000b), en kilo-octets (1000k) ou en mega-octets (1000m).

Contenu des logs

Outre la localisation des logs, il est possible également de paramétrer le contenu des logs. On utilisera les attributs `log_on_success` et `log_on_failure` qui, comme l'intitulé l'indique, listent ce qui sera logué en cas de succès et d'échec de l'accès au service. Les valeurs sont :

- `PID` : numéro de process du serveur lancé
- `HOST` : adresse distante cliente du serveur
- `USERID` : user id de l'utilisateur distant
- `DURATION` : durée de la session

Xinetd

Les 4 valeurs sont utilisables avec l'attribut `log_on_success`. Seuls `HOST` et `USERID` sont utilisables avec `log_on_failure`.

Exemple :

```
service trucmuche
{
    ...
    log_type = FILE /var/log/xinetd/trucmuche.log
    log_on_success = PID USERID HOST DURATION
    log_on_failures = HOST USERID
    ...
}
```

Dans ce cas de figure, le fichier de log du service `trucmuche` sera `/var/log/xinetd/trucmuche.log`. Les informations recueillies dans tous les cas seront l'adresse du client et son identité et en cas de succès, on aura également le PID du serveur et la durée de la session

Xinetd pour contrôler les accès à votre machine

Xinetd dispose de nombreux attributs complémentaires, facultatifs mais qui vont permettre d'en faire un outil de sécurisation des services réseaux et du système dans son ensemble. Ci-dessous, les principales fonctionnalités proposées.

Contrôler l'origine des accès

Avec des attributs supplémentaires, vous allez pouvoir filtrer les clients qui vont pouvoir ou non se connecter à vos serveurs.

Filtrer des adresses : `only_from = valeur [valeurs...]`.

La connexion au service ne sera possible qu'à partir de la liste fournie à cet attribut. Elle peut contenir :

- des adresses IP : facile à comprendre je ne m'étends pas.
- des adresses réseau : 192.168.0.0 par exemple. Seules les adresses IP de ce réseau pourront accéder au service
- des hostnames : Ce sont des noms de machine. A utiliser si et seulement si le fichier `/etc/hosts` est correctement renseigné. La résolution de nom se fait au moment de l'accès.
- des noms de domaine : `lea-linux.org` par exemple. Seul le domaine de Léa pourra alors accéder à votre service.

Vous pouvez bien sûr panacher les valeurs.

Exemple : configuration des accès ftp sur ma machine :

```
service proftpd
{
    ...
    only_from = citrouille 192.168.1.0 lea-linux.org
    ...
}
```

Xinetd

Les seuls autorisés à utiliser mon serveur ftp seront : la machine "citrouille", les machines du réseau 192.168.1.0 et celles du domaine lea-linux.org.

Un attribut qui a également pour effet de filtrer les accès est `no_access = valeur [valeurs...]`. Il fonctionne exactement de la même façon que `only_from` sauf qu'il détermine les machines, adresses IP, hostnames, adresses réseaux et/ou noms de domaine pour lesquels vous voulez interdire l'accès à votre service.

Attention, encore une fois, `xinetd` utilisé seul ne vous garantira pas la sécurité de votre système. Il est essentiel d'y adjoindre un bon firewall.

Contrôler le moment des accès

Vous pouvez choisir le moment auquel vous autoriserez les accès à tout ou partie de vos services réseaux. On utilisera l'attribut `access_times`. Il vous permet de définir une ou plusieurs plages horaires pendant lesquelles la connexion sera possible.

Syntaxe : `access_times = interval [interval...]`
L'intervalle de temps s'écrit : heures:minutes-heures:minutes

Exemple : Je veux limiter l'accès de mon ftp de 9h à 12h et de 14h à 16h.

```
service proftpd
{
    ...
    access_time = 9:00-12:00 14:00-16:00
    ...
}
```

Contrôler l'exposition du système pendant l'accès : chroot

Véritable couteau suisse de la configuration de services réseau, `xinetd` vous permet de "chrooter" un service. *Rappel* : la commande `chroot` permet de lancer un programme en restreignant ses accès disques à une sous arborescence. En fait pour le processus, la racine du disque est la racine de l'arborescence dans laquelle il a été restreint.

L'attribut `server_args` va nous permettre d'automatiser le chroot : la commande `chroot` est considérée comme le serveur et le service est passé en argument.

Exemple : Je veux chrooter mon serveur ftp.

```
service proftpd
{
    ...
    server = /usr/sbin/chroot
    server_args = /opt/proftpd/proftpd
    ...
}
```

Lorsqu'un client tente un accès ftp, `chroot` est exécuté en tant que serveur et `proftpd` en tant qu'argument, ce qui revient à la commande connue : `/usr/bin/chroot opt/proftpd/proftpd`.

Autoriser / interdire un service

Dans la plupart des distributions, les services installés et gérés par `xinetd` sont désactivés d'office, pour des raisons de sécurité. D'autre part vous pouvez, pour un temps, choisir de désactiver complètement un service. Tout ceci est lié à l'utilisation de l'attribut `disable`.

Syntaxe : `disable = yes|no`

Exemple : Je veux désactiver telnet.

```
service telnet
{
    disable = yes
    ...
}
```

Xinetd pour limiter les attaques de type Deny of Service

- **Contrôle de la charge CPU :** `rlimit_cpu = seconds`. Cet attribut vous permet de limiter le temps CPU utilisé par un ou plusieurs services. Un des effets induits par une attaque de type Deny Of Service.
- **Priorité accordée au processus serveur :** `nice = level`. Fonctionnant comme avec la commande `nice`, l'attribut permet de fixer une priorité d'ordonnancement pour le serveur. Le `level` peut prendre les valeurs de `-20` (le plus prioritaire) à `19` (le moins prioritaire). Cela vous permet par exemple de passer en process moins prioritaire un serveur ftp par rapport aux process de vos applications courantes. Après tout c'est votre machine ;).
- **Limite du nombre de connexions par service :** `instances = value`. L'attribut détermine le nombre d'instances simultanées du serveur qui seront autorisées. Préciser un nombre. Sans précision, le nombre d'instances pourra être illimité.
- **Limite du nombre de connexions ayant la même origine :** `per_source = value`. Non seulement vous pouvez filtrer les adresses IP clientes, le nombre d'instances du serveur mais vous pouvez aussi limiter le nombre de connexions à un serveur donné provenant d'une même adresse IP. Ceci est une limite non négligeable aux attaques dues à des connexions multiples sur vos serveurs accessibles.
- **Blacklister des adresses IP :** Il vous est possible blacklister des adresses IP qui tenteraient des connexions sur des services que vous avez désactivés mais qui constituent la cible préférée des hackers (exemple : telnet). On utilisera 2 attributs de manière combinée :

```
flags = SENSOR
deny_time = minutes
```

`SENSOR` empêche toute connexion au service concerné et stocke l'adresse IP pendant un temps déterminé par l'attribut `deny_time`. Si cette même adresse tente de se connecter à **n'importe quel service géré par `xinetd`**, il sera automatiquement bloqué. Le temps est déterminé en minutes, mais vous pouvez utiliser également la valeur `FOREVER` : l'IP restera blacklistée jusqu'au prochain redémarrage de `xinetd`.

Autres fonctionnalités de xinetd

Je ne verrai pas ici toutes les possibilités offertes par xinetd mais 2 en particulier : la redirection de port et l'attribution d'un service à une interface réseau.

Redirection de ports

Même si ce n'est pas sa fonction principale, xinetd peut vous permettre de faire de la redirection de port, si vous ne souhaitez pas approfondir iptables. On utilisera pour cela l'attribut **redirect**.

Syntaxe : `redirect = adresseIP port`

Exemple : Je dispose d'une passerelle vers Internet et une machine sur le même réseau local dont l'adresse IP est 192.168.0.3. Je souhaite installer un serveur telnet sur cette dernière machine. Je vais donc faire en sorte que les requêtes qui arrivent de l'extérieur pour mon serveur telnet soient redirigées vers cette machine.

```
root@pingu# cat /etc/xinetd.d/telnet
service telnet
{
    ...
    server = /usr/sbin/in.telnetd
    redirect = 192.168.0.4 23
}
```

On teste maintenant la connexion :

```
root@pingu# telnet 217.11.12.13
Trying 217.11.12.13...
Connected to 217.11.12.13.
Escape character is '^]'.
Red Hat Linux release 8.0 (Psyche)
Kernel 2.4.18-14 on an i586
login: anne
Password:
Last login: Fri Nov 8 12:26:44 on :0
[anne@citrouille anne]$ hostname
citrouille
```

La connexion telnet est établie et la commande `hostname` nous confirme que je ne suis pas sur la passerelle mais sur la machine citrouille dont l'adresse IP est 192.168.0.4

Attribution d'un service à une adresse IP

xinetd va vous permettre de lier un service à une adresse IP grâce à l'attribut `bind`.

Pour éclaircir la description de cet attribut, nous allons nous appuyer sur un exemple. Je vais reprendre mes 2 machines de l'exemple précédent. Mon objectif : je veux construire 2 serveurs ftp bien différenciés. L'un sera réservé à mes machines en local (serveur de fichiers interne) l'autre mettra à disposition d'autres fichiers sur un serveur ftp réservé aux connexions externes. (Après tout je ne partage pas tout :)).

Xinetd

Ma machine pingu (ma passerelle) a 2 interfaces réseau avec les adresses IP respectives : 192.168.0.3 et 217.11.12.13. Je vais donc effectuer les attributions grâce au fichier suivant :

```
root@pingu# cat /etc/xinetd.d/proftpd
service proftpd
{
    id = ftp_public
    ...
    server = /opt/proftpd/proftpd
    bind = 217.11.12.13
}
service proftpd
{
    id = ftp_privé
    ...
    server = /opt/proftpd/proftpd
    bind = 192.168.0.3
}
```

L'attribut `id` sert uniquement à différencier les 2 configurations du service. Tout ceci bien sûr doit s'accompagner d'une configuration de votre serveur ftp.

xinetd parle à vos visiteurs

Pour finir, `xinetd` vous permet également d'afficher des messages lors de la connexion à un service grâce aux attributs suivants :

- **banner = fichier** – affiche le contenu de `fichier` si l'accès est autorisé, et avant même authentification.
- **banner_succes = fichier** – affiche le contenu de `fichier` en cas de réussite de l'authentification
- **banner_fail = fichier** – affiche le contenu de `fichier` en cas d'échec de l'authentification

Exemples de configuration

Ci-dessous quelques exemples de fichiers de configuration de services, issus entre autres du man de `xinetd.conf`

Exemple 1

Le 1er exemple est un fichier de configuration du service `ftp`.

```
root@pingu# cat /etc/xinetd.d/ftp
service ftp
{
    socket_type = stream
    wait = no
    nice = 10
    user = root
    server = /usr/etc/in.ftpd
}
```

Xinetd

```
server_args = -l
instances = 4
log_type = FILE /var/log/ftp.log
log_on_success = DURATION HOST USERID
access_times = 2:00-9:00 12:00-24:00
}
```

Le serveur ftp, hormis les paramètres génériques de configuration, autorise des connexions entre 2h et 9h et entre midi et minuit. Son fichier de log est `/var/log/ftp.log` et contient, en cas de connexions réussies, la durée de la connexion, le nom de la machine cliente et l'UID de l'utilisateur connecté. Seules 4 instances simultanées sont autorisées

Exemple 2

Le 2e exemple est un fichier de configuration du service telnet.

```
root@pingu# cat /etc/xinetd.d/telnet
service telnet
{
    socket_type = stream
    wait = no
    nice = 10
    user = root
    server = /usr/etc/in.telnetd
    rlimit_as = 8M
    rlimit_cpu = 20
}
```

Exemple 3

Le dernier exemple se compose de 2 fichiers de configuration pour la gestion de samba. par xinetd. En effet, le service samba lance 2 démons, nmbd et smbd

```
root@pingu# cat /etc/xinetd.d/netbios-ssn
service netbios-ssn
{
    socket_type = stream
    protocol = tcp
    wait = no
    user = root
    server = /usr/sbin/smbd
    only_from = 192.168.0.0
    disable = no
}

root@pingu# cat /etc/xinetd.d/netbios-ns
service netbios-ns
{
```

Xinetd

```
socket_type = dgram
protocol = udp
wait = no
user = root
server = /usr/sbin/nmbd
only_from = 192.168.0.0
disable = no
}
```

Hormis les paramètres génériques de configuration de `smbd` et `nmbd`, on a configuré `samba` de manière à ce qu'il ne soit utilisable que sur le réseau interne 192.168.0.0.

Le mot de la fin

Je n'ai pas exploré ici l'intégralité des possibilités de `xinetd`. La meilleure des choses à faire pour cela est de se reporter au man. Si vous disposez encore une distribution utilisant `inetd`, et au vu de toutes les fonctionnalités de `xinetd`, il semble opportun d'abandonner le premier pour le deuxième. Bien exploité, il devient véritablement un outil de sécurisation de votre système, dès lors que vous mettez en place un certain nombre de serveurs qui, ouvrant votre machine sur l'extérieur, ouvre également la voie aux vilains hackers ;).

N'hésitez pas à me faire parvenir vos remarques et ajouts divers sur le contenu de cet article.

Cette page est issue de la documentation 'pré-wiki' de Léa a été convertie avec `HTML::WikiConverter`. Elle fut créée par Anne le 15/11/2002.

Copyright

Copyright © 15/11/2002, Anne



*Ce document est publié sous licence Creative Commons
Attribution, Partage à l'identique, Contexte non commercial 2.0 :
<http://creativecommons.org/licenses/by-nc-sa/2.0/fr/>*