.

Clam AntiVirus 0.65
Manuel de l'utilisateur

par Tomasz Kojm

Traduction française : Stéphane JEANNENOT
<stephane.jeannenot*wanadoo.fr>

Table des matières

1 Introduction

Clam AntiVirus est un ensemble d'outils antivirus pour UNIX. Le but principal de ce logiciel est l'intégration aux serveurs de courriel (vérification automatique). L'ensemble logiciel fournit un démon (processus en arrière-plan) flexible, adaptable et "multi-thread", un analyseur en ligne de commande, et un outils de mise à jour automatique via Internet.Les programes sont basés sur une librairie partagée et distribuée avec l'ensemble des logiciels Clam AntiVirus, que vous pouvez utiliser dans vos propres logiciels.

1.1 Caractéristiques

- Sous license GNU General Public License, Version 2
- Conforme à la norme POSIX, portable
- Analyse très rapide
- Analyse lors de l'accès (Linux seulement)
- Détecte plus de 10000 virus, vers et troyens
- Supporte les archives et les fichiers compressés
- Support natif pour RAR (2.0), Zip, Gzip, Bzip2
- Protection native contre les bombes archives
- Support natif pour Mbox, Maildir et les fichiers bruts de courriel
- Inclus un logiciel de mise à jour de la base de données avec la possibilité de signatures digitales

1.2 Liste de diffusion

Il y a quatre listes de diffusion disponibles :

- **clamav-announce*lists.sf.net** - informations à propos des nouvelles versions (y compris les sorties des paquets Debian), modéré [1]
- **clamav-users*lists.sf.net** - questions des utilisateurs
- **clamav-devel*lists.sf.net** - développement
- **clamav-virusdb*lists.sf.net** - annonces concernant la mise à jour de la base de données

Vous pouvez souscrire et consulter les archives des listes de diffusion à

`http ://www.clamav.net/ml`

1.3 Signaler un virus

Si vous avez un virus qui n'a pas été détecté par votre ClamAV avec les dernières bases de données, veuillez le vérifier avec le *ClamAV Online Specimen Scanner* :

`http ://www.gietl.com/test-clamav`

et signalez le ensuite sur notre site web :

`http ://www.clamav.net/cgi-bin/sendvirus.cgi`

Sinon vous pouvez l'envoyer à cette adresse :

`virus*clamav.net`

Si votre système ne vous permet pas d'envoyer des fichiers infectés, veuillez créer une archive zip protégée par le mot de passe : `virus`

[1] Cela signifie que les souscripteurs ne sont pas autorisés à écrire dans la liste de diffusion

2 Installation

2.1 Minimum requis

Vous aurez besoin des paquets *zlib* et *zlib-devel* ainsi que du compilateur *gcc* (les versions 2.9x et 3.x sont toutes deux supportées). Vous pouvez installer la librairie *bzip2* (et ses fichiers de développement) pour obtenir le support de bzip2, mais ce n'est pas obligatoire. **Il est fortement recommandé d'installer la librairie GNU MP 3 de manière à pouvoir utiliser les signatures numériques de la base de données.**
ASTUCE SOLARIS : *Vous devriez affecter la variable système ABI à 32 (ex.* ***setenv ABI 32****) avant de lancer la configuration de GMP* [2]

2.2 Plateformes supportées

Clam AntiVirus est préparé à l'installation sur les systèmes d'exploitation suivants / architectures (les plateformes testées sont entre parenthèses) :

- GNU/Linux 2.2/2.4 (Toutes saveurs, Intel/SPARC/Alpha/zSeries/S/390)
- Solaris 2.6/7/8/9 (Intel/SPARC)
- FreeBSD 4.5/6/7 5.0 (Intel/Alpha)
- OpenBSD 3.0/1/2 (Intel)
- AIX 4.1/4.2/4.3/5.1 (RISC 6000)

2.3 Paquets binaires

Des paquets binaires *deb* et *rpm* de grande qualité sont disponibles pour Linux. Le paquet Debian est maintenu par Magnus Ekdahl et vous le trouverez sur les miroirs debian, `http ://www.debian.org`. Le paquet RPM est maintenu par Arkadiusz Miskiewicz et est distribué avec la distribution Linux polonaise (`ftp ://ftp.pld.org.pl`). Il y a aussi un paquet RPM officiel pour Mandrake (maintenu par Oden Eriksson) et des paquets binaires pour AIX dans AIX PDSLIB, UCLA `http ://aixpdslib.seas.ucla.edu/packages/clamav.html`. Les portages BSD sont disponibles pour Free, Net et OpenBSD. Le portage officiel pour FreeBSD est maintenu par Masahiro Teramoto. Le portage officiel pour NetBSD est maintenu par INCONNU. Le portage non-officiel pour OpenBSD (maintenu par Flinn Mueler) est disponible à l'adresse
`http ://www.activeintra.net/openbsd/article.php ?id=5`.

2.4 Installation

Veuillez lire le fichier README de la version actuelle, car il peut contenir d'importantes informations. Si vous installez ClamAV pour la première fois, vous devez ajouter un nouvel utilisateur *clamav* et le groupe correspondant sur votre système : [3]

```
# groupadd clamav
# useradd -g clamav -s /bin/false -c "Clam AntiVirus" clamav
\end{verbatom}
La méthode ci-dessus fonctionne pour Linux et Solaris, dans le cas où vous n'avez pas
\begin{verbatim}
$ ./configure --disable-clamav
```

[2] Merci à Ed Phillips

[3] Note pour Cygwin : si vous n'avez pas /etc/passwd, vous n'avez pas besoin d'effectuer ces changements

Ceci évite le test du groupe et de l'utilisateur *clamav*. **clamscan** nécessite toujours que *clamav* fonctionne en mode superutilisateur. Le mot de passe du compte *clamav* devrait être verrouillé dans */etc/passwd* ou */etc/shadow*.
Un fois que vous avez créé le groupe et l'utilisateur clamav, veuillez extraire l'archive :

```
$ zcat clamav-x.yz.tar.gz | tar xvf -
$ cd clamav-x.yz
```

En supposant que vous voulez que le fichier de configuration soit installé dans */etc*, configurez le paquet comme ceci :

```
$ ./configure --sysconfdir=/etc
```

Actuellement, *gcc* est nécessaire à la compilation. Le support d'autres compilateurs sera ajouté dans un futur proche.

```
$ make
$ su -c "make install"
```

Lors de la dernière étape le logiciel est installé dans le répertoire /usr/local et le fichier de configuration dans /etc.
ATTENTION : Ne jamais affecter les bits SUID ou SGID aux programmes Clam AntiVirus.

2.5 Configuration

Si vous voulez utiliser le démon, vous devez le configurer car il ne peut fonctionner sans configuration par défaut :

```
$ clamd
ERROR: Please edit the example config file /etc/clamav.conf
```

Ceci montre l'endroit où se trouve le fichier de configuration. Le format et les options de ce fichier sont complètement décrits dans le manuel *clamav.conf(5)*. La configuration de clamd est très simple car le fichier de configuration est bien commenté. Rappelez-vous - vous êtes obligé d'enlever la commande "Example".

Une autre caractéristique de clamd est l'analyse lors de l'accès, basée sur le module Dazuko, disponible à l'adresse `http ://dazuko.org`. **Ceci n'est pas nécessaire pour lancer clamd - de plus vous ne devriez pas lancer Dazuko sur des systèmes de production.** Un thread spécial dans clamd est responsable de la communication avec Dazuko et s'appelle "Clamuko" (à cause du nom rigolo de Dazuko - je ne sais pas ce que "Clamuko" signifie). Clamuko est supporté par les noyaux Linux 2.2 et 2.4 seulement. Pour compiler dazuko, exécutez :

```
$ tar zxpvf dazuko-a.b.c.tar.gz
$ cd dazuko-a.b.c
$ make dazuko
```

ou bien

```
$ make dazuko-smp (pour les noyaux smp)
$ su
# insmod dazuko.o
# cp dazuko.o /lib/modules/'uname -r'/misc
# depmod -a
```

Selon votre distribution Linux, vous devrez ajouter une entrée "dazuko" au fichier */etc/modules* ou quelque chose comme :

```
modprobe dazuko
```

à un fichier de démarrage de manière à charger dazuko au démarrage. Vous devez aussi créer le périphérique */dev/dazuko* :

```
$ cat /proc/devices | grep dazuko
254 dazuko
$ su -c "mknod -m 600 /dev/dazuko c 254 0"
```

Maintenant, configurez juste Clamuko dans *clamav.conf*. Veuillez vous reporter à la section 3.2

2.6 Tests

OK, faisons quelques tests. Essayez d'analyser récursivement le répertoire source :

```
$ clamscan -r -l scan.txt clamav-x.yz
```

Il devrait y avoir quelques virus (de test) dans le répertoire clamav-x.yz. Le résultat de l'analyse est sauvegardé dans le fichier de log scan.txt. [4]. Pour tester clamd : lancez-le et utilisez *clamscan* (vous pouvez ainsi vous connecter directement à clamd et lancer la commande SCAN) :

```
$ clamscan -l scan.txt clamav-x.yz
```

La sortie et le fichier de log devraient être similaires à ceux obtenus avec *clamscan*.

2.7 freshclam : Configurer la mise à jour automatique

freshclam est le programme par défaut de mise à jour pour Clam AntiVirus. Il peut fonctionner selon deux modes :

- de manière interactive - en ligne de commande
- comme un démon - seul, silencieusement

Lorsqu'il est lancé avec les droits de superutilisateur, il supprime les privilèges et change par défaut d'utilisateur : l'utilisateur *clamav*. *freshclam* utilise le DNS à tourniquet `database.clamav.net` qui choisit automatiquement un miroir (cf. 2.8) de la base de données. freshclam est un outil avancé : il supporte les serveurs proxy (avec authentification), la vérification des signatures numériques, et diverses erreurs possibles. **Test rapide : lancez *freshclam* (en tant que superutilisateur) sans aucun paramètre et regardez le résultat retourné.** Si tout s'est bien passé, vous pouvez créer un fichier log dans le répertoire /var/log (accessible par *clamav* ou un autre utilisateur). freshclam fonctionne ici en mode *utilisateur* :

```
# touch /var/log/clam-update.log
# chmod 600 /var/log/clam-update.log
# clamav /var/log/clam-update.log
```

Maintenant, vous pouvez lancer *freshclam* en mode démon :

```
# freshclam -d -c 6 -l /var/log/clam-update.log
```

[4] Pour plus d'informations sur les options de clamscan : **man clamscan**

Ceci permet la mise à jour (si nécessaire) de la base de données six fois par jour (et c'est une valeur minimale suggérée). Vous pouvez ajouter cette ligne à vos scripts de démarrage. L'autre moyen est d'utiliser le démon *cron*. Vous devez alors ajouter la ligne suivante au crontab de **root** ou **clamav** :

```
0 * * * * /usr/local/bin/freshclam --quiet -1 /var/log/clam-update.log
```

pour vérifier la présence d'une nouvelle base de données chaque heure. Pour configurer le support du proxy, vous devez modifier la variable d'environnenement *$http_proxy* :

```
export http\_proxy="my.proxy.server:8080"
```

ou utilisez les options `-http_proxy` et `-proxy-user`.

2.8 freshclam : Miroirs et mirrors.txt

freshclam télécharge la base de données depuis `http ://database.clamav.net`. Il s'agit d'un tourniquet qui essaye d'équilibrer le traffic entre les différents miroirs de téléchargement :

Miroir	IP	Emplacement	Administrateur
clamav.man.olsztyn.pl	213.184.16.3	Olsztyn, Pologne	Robert d'Aystetten `dart*man.olsztyn.pl`
avmirror1.prod.rxgsys.com	64.74.124.90	USA	`graham*rxgsys.com`
avmirror2.prod.rxgsys.com	207.201.202.73	USA	`graham*rxgsys.com`
clamav.e-admin.de	212.162.12.159	Düsseldorf, Allemagne	Andreas Gietl `a.gietl*e-admin.de`
clamav.essentkabel.com	195.85.130.84	Pays-Bas	Chris van Meerendonk `mirror*essentkabel.com`
clamav.inet6.fr	62.210.153.201 62.210.153.202	France	Lionel Bouton `clamavdb*inet6.fr`
clamav.netopia.pt	193.126.14.29	Portugal	Miguel Bettencourt Dias `mbd*netopia.pt`
clamav.sonic.net	209.204.175.217	USA	Kelsey Cummings `kgc*sonic.net`
clamav.nettron.co.za	160.124.112.17	Afrique du Sud	Ryan Zwankhuizen `info*nettron.co.za`
clamav.nchost.net	203.208.228.153	Singapour	Nicholas Chua `nicholas*ncmbox.net`

Dans le répertoire local de la base de données, il y a un fichier *mirror.txt* que freshclam lit à chaque fois qu'il essaye de mettre à jour la base de données. freshclam se connecte à un premier serveur de la liste et si la connexion échoue, un autre serveur est contacté. Normalement, vous n'avez pas à modifier ce fichier, à moins que vous ne vouliez utiliser votre propre miroir local lors des mises à jour.

3 Utilisation

3.1 Le démon clam

clamd est un démon multi-thread basé sur *libclamav*. Il fonctionne selon l'un des deux modes suivants, à savoir :

- Unix (local) socket
- TCP socket

Le démon est totalement configurable via le fichier *clamav.conf*. Vous trouverez une description de chaque commande das le manuel **clamav.conf(5)**. *clamd* reconnaît les commandes suivantes :

- **PING**
  Vérifie l'état du démon (il devrait répondre avec "PONG").
- **VERSION**
  Affiche la version.
- **RELOAD**
  Recharge les bases de données.
- **QUIT**
  Quitte "proprement".
- **SCAN file/directory**
  Analyse un fichier ou un répertoire (récursivement) avec le support des archives activé. Un chemin complet est nécessaire.
- **RAWSCAN file/directory**
  Analyse un fichier ou un répertoire (récursivement) avec le support des archives désactivé. Un chemin complet est nécessaire.
- **CONTSCAN file/directory**
  Analyse un fichier ou un répertoire avec le support des archives activé et ne s'arrête pas, même si un virus est détecté.
- **STREAM**
  Analyse un flux de données. Clamd retournera un nouveau numéro de port auquel vous devrez vous connecter et envoyer les données à analyser. *Le protocole est obsolète et il y aura une nouvelle version bientôt (cependant celui-ci sera toujours supporté).*

Les threads internes, à l'exception de clamuko, ignorent tous les signaux externes. Le thread principal s'occupe des signaux *SIGTERM* et *SIGINT* et effectue une sortie "propre".

3.2 Clamuko

Clamuko est un thread spécial dans *clamd* qui réalise l'analyse à la demande sous Linux. Il a été implémenté comme un thread dans clamd à cause de l'implémentation de Dazuko. Le modèle client (clamuko) serveur (clamd) n'est pas actuellement supporté par Dazuko. Cependant, l'implémentation actuelle a quelques avantages : clamuko partage la base interne de virus avec clamd et il est mis à jour avec la commande RELOAD. **Vous devez suivre scrupuleusement les principes suivants lorsque vous utilisez clamuko :**

- Toujours arrêter le démon correctement, en utilisant la commande QUIT ou le signal SIGTERM. Sinon vous pouvez perdre votre accès aux fichiers protégés jusqu'au redémarrage du système.
- Ne jamais protéger un répertoire que le logiciel de vérification des mails utilise pour décompresser les fichiers attachés. L'accès à tous les fichiers infectés sera automatiquement bloqué et le scanner (même clamd) ne sera pas capable de détecter un virus. **Le mail infecté sera envoyé.**

Vous avez besoin d'activer clamuko dans *clamav.conf*. Pour protéger le répertoire /home, tapez la commande :

```
ClamukoIncludePath /home
```

Pour protéger entièrement le système :

```
ClamukoIncludePath /
ClamukoExcludePath /proc
ClamukoExcludePath /temporary/dir/of/your/mail/scanning/software
```

Vous pouvez utiliser clamuko pour protéger l'accès aux fichiers avec Samba/Netatalk (mais une meilleure idée, plus sûre, est d'utiliser le logiciel **samba-vscan** cf 5.17. NFS n'est pas supporté (Dazuko ne peut intercepter les appels d'accès à NFS). Maintenant, une autre idée : vous pourriez construire une base de données contenant les signatures des exploits populaires et configurer clamd pour protéger votre serveur des "script-kiddies".

3.3 Archives et fichiers compressés

Tous les analyseurs dépendent de LibClamAV. Un support natif des formats suivant est intégré :

- Zip
- Gzip
- Bzip2
- RAR (2.0 uniquement)

Le type d'archive es déterminé par des tests de nombres magiques [5]. Vous devez avoir la librairie zlib d'installée pour obtenir le support de Zip/Gzip. Les archives Zip sont accessibles avec la librairie zziplib de Guido Draheim et Tomi Ollila. Le support de RAR est basé sur la librarie UniquE RAR de Christian Scheurer et de Johannes Winkelmann. Les deux sont inclus et légèrement modifiés dans les sources de clamav. Unrarlib supporte uniquement les archives RAR 2.0 et d'après Christian le nouveau format (introduit par WinRAR 3.0) ne sera jamais supporté (cependant clamscan peut analyser les archives WinRAR 3.0, cf. ci-dessous). Pour des raisons de sécurité clamd analyse uniquement les archives supportées par libclamav. Clamscan est plus intelligent et il peut aussi utiliser des décompresseurs externes ; ceci est particulièrement utile lorsque le décompresseur interne échoue :

```
$ clamscan --unrar rarfail.rar
/home/zolw/Clam/test/rarfail.rar: RAR module failure
UNRAR 3.00 freeware  Copyright (c) 1993-2002 Eugene Roshal
Extracting from /home/zolw/Clam/test/rarfail.rar
Extracting test1                              OK
All OK
/tmp/44694f5b2665d2f4/test1: ClamAV-Test-Signature FOUND
/home/zolw/Clam/test/rarfail.rar: Infected Archive FOUND
```

Astuce : *vous pouvez forcer clamscan à lister tous les fichiers infectés dans une archive en utilisant* `-disable-archive` *(désactive le décompresseur interne) et* `-unzip -unrar` *...*

clamscan supporte la plupart des archiveurs et il utilise des programmes externes pour

[5] Cela fonctionne comme la commande classique `file(1)`

chaque format. **Si le scanner fonctionne avec les privilèges du superutilisateur, les décompresseurs sont exécutés avec les privilèges *clamav* ce qui rend le processus plus sûr.** clamscan vérifie aussi que l'utilisateur a le droit de lecture de tous les fichiers. **Vous devez activer l'analyse récursive avec l'option `-r` (`-recursive`), si vous voulez analyser l'ensemble du contenu d'une archive (sous-répertoires inclus)** ; cette option est aussi (généralement) nécessaire pour analyser les archives "? ? nested ? ?" Les décompresseurs externes supportés sont :

- **-unzip** : en général, vous n'avez pas besoin de cette option parce que le format Zip est supporté par libclamav. Cependant, il peut être utile si libclamav n'arrive pas à dézipper certains fichiers. clamscan a été testé avec *UnZip 5.41 du 16 avril 2000, par Info-Zip.*
- **-unrar** : testé avec le freeware *UNRAR 3.00*
- **-unace** : utilise une option supportée par *UNACE v1.2 public version*, non testé, mais devrait fonctionner.
- **-arj** : testé avec *arj 3.10b.*
- **-zoo** : testé avec *zoo 2.1.*
- **-lha** : testé avec *LHA for Unix V 1.14e.*
- **-jar** : clamscan utilise *unzip* pour les fichier .jar . Testé avec *UnZip 5.41 du 16 avril 2000, par Info-ZIP.*
- **-tar** : cette option active le support des archives non-compressées. Testé avec *GNU tar 1.13.17.*
- **-deb** : cette option active le support des paquets binaires debian. Testé avec *GNU ar 2.12.90.0.14.* Implique -tgz, mais pas de confilt avec -tgz=FULLPATH.
- **-tgz** : cette option active le support des fichiers .tar.gz et .tgz . Vous avez besoin de *GNU tar*, sur les système non-Linux, vous l'avez probablement installé sous *gtar* et s'il ne peut être trouvé dans le *$PATH*, veuillez utiliser *-tgz=gtar* pour indiquer à clamscan d'utiliser *gtar* à la place de *tar*. Sinon, donnez un chemin complet avec *-tgz.*

3.4 Format de sortie

clamd utilise un format de sortie compatible avec clamscan :

```
zolw@Wierszokleta:~$ telnet localhost 3310
Trying 127.0.0.1...
Connected to localhost.
Escape character is '\^]'.
SCAN /home/zolw/infected
/home/zolw/infected/sobre.com: W32/Magistr.B FOUND
Connection closed by foreign host.
```

En mode **SCAN**, la connexion est fermée lorsque le premier virus est trouvé. Dans le cas d'archives, la sortie est exactement la même qu'avec des fichiers normaux :

```
SCAN /home/zolw/Clam/test/test2.zip
/home/zolw/Clam/test/test2.zip: ClamAV-Test-Signature FOUND
```

CONTSCAN affiche tous les fichiers infectés trouvés.
Les messages d'erreur sont affichés sous ce format :

```
SCAN /no/such/file
/no/such/file: Can't stat() the file ERROR
```

et peuvent être facilement analysés.

clamscan écrit tous les messages vers **stderr** (seule l'aide est envoyée vers **stdout** par défaut). Si vous voulez les rediriger vers **stdout**, utilisez *-stdout*. Exemple d'une sortie de clamscan :

```
/tmp/test/removal-tool.exe: Worm.Sober FOUND
/tmp/test/md5.o: OK
/tmp/test/blob.c: OK
/tmp/test/message.c: OK
/tmp/test/error.hta: VBS.Inor.D FOUND
```

Lorsqu'un virus est découvert, son nom est affiché entre les chaînes *filename :* et *FOUND*. Si un virus est découvert dans une archive qui a été décompressée par un programme externe, elle est marquée comme *Archive Infectée*. "Les Archives Infectées" ne sont pas comptées comme des fichiers infectées ; seuls les fichiers qu'elles contiennent le sont. Remarquez la différence avec le décompresseur interne qui marque juste que l'ensemble de l'archive est infecté ; en effet, le processus d'extraction effectué par libclamav est transparent et clamscan ne sait pas quel fichier est infecté.

3.5 Outil de signature

sigtool automatise la création de signature. Si vous avez un fichier infecté qui n'a pas été reconnu par ClamAV et qu'un autre antivirus fonctionnant en mode console l'a découvert, vous pouvez essayer de créer une signature automatiquement. *Sigtool est en partie utile car in ne détecte que la dernière partie d'une signature réelle. Il échouera avec les signatures en plusieurs parties (souvent utilisées pour rechercher les virus polymorphes).* Exemple d'utilisation : créez un fichier quelconque (avec n'importe quel contenu) et insérez le contenu du fichier **test1** à l'intérieur. Nous allons utiliser *clamscan* pour générer la signature. Ceci n'est qu'un exemple ; en réalité, vous n'avez pas besoin d'une telle astuce. Analysez maintenant le fichier avec *clamscan -stdout testfile* et la sortie devrait être la suivante :

```
testfile: ClamAV-Test-Signature FOUND
----------- SCAN SUMMARY -----------
Known viruses: 10213
Scanned directories: 0
Scanned files: 1
Data scanned: 0.95 MB
Infected files: 1
I/O buffer size: 131072 bytes
Time: 0.245 sec (0 m 0 s)
```

La seule chaîne de caractère dans cette sortie est "ClamAV-Test-Signature". Lancez alors *sigtool* avec les arguments suivants :

```
$ sigtool -c "clamscan --stdout" -f testfile -s "ClamAV-Test"
```

Le programme va concaténer les arguments pour *-c (-command)* et *-f (-file)*, c'est pourquoi les arguments doivent être donnés dans le bon ordre. A la fin, un fichier *testfile.sig* sera généré et devrait contenir 100 bytes dans notre exemple. Ce fichier contient la bonne signature.

```
Detected, decreasing end 20051 -> 16040
Detected, decreasing end 16040 -> 12029
Detected, decreasing end 12029 -> 8018
Not detected at 8018, moving forward.
Detected, decreasing end 10024 -> 8018
Not detected at 8018, moving forward.
Detected, decreasing end 9021 -> 8018
Not detected at 8018, moving forward.
Not detected at 8520, moving forward.
Detected, decreasing end 8771 -> 8520
Not detected at 8520, moving forward.
Not detected at 8646, moving forward.
Not detected at 8709, moving forward.
Detected, decreasing end 8741 -> 8709
Not detected at 8709, moving forward.
Not detected at 8725, moving forward.
Detected, decreasing end 8733 -> 8725
Not detected at 8725, moving forward.
Not detected at 8729, moving forward.
Detected, decreasing end 8731 -> 8729
Not detected at 8729, moving forward.
Detected, decreasing end 8730 -> 8729
Not detected at 8729, moving forward.
Increasing end 8729 -> 8730
*** Signature end found at 8730
Detected at 8680, moving forward.
Detected at 8680, moving forward.
Not detected, moving backward 8693 -> 8680
Detected at 8680, moving forward.
Not detected, moving backward 8687 -> 8680
Detected at 8680, moving forward.
Not detected, moving backward 8684 -> 8680
Detected at 8680, moving forward.
Not detected, moving backward 8682 -> 8680
Detected at 8680, moving forward.
Not detected, moving backward 8681 -> 8680
Detected at 8680, moving forward.
Not detected, moving backward 8681 -> 8680
Detected at 8680, moving forward.
Moving forward 8680 -> 8681
*** Signature start found at 8681
The scanner was executed 33 times.
The signature length is 49 (98 hex)
Saving signature in testfile.sig file.
Saving binary signature in testfile.bsig file.
```

Pour générer une signature complète, vous avez seulement besoin de rajouter la chaîne *VirusName=* au début de la signature hexadécimale dans testfile.sig .

Astuce : *les scanners ClamAV lisent tous les fichiers .db dans le répertoire des bases de données. Vous pouvez créer vos propres fichiers de données (ex : local.db) et ils ne*

seront pas modifiés par freshclam !

4 Résoudre les problèmes

4.1 Codes de retour

Les codes de retour sont très pratiques, surtout pour les scripts système. Vous pouvez vérifier le code de retour de *clamscan* en lançant la commande suivante :

```
$ clamscan; echo Return code: $?
```

Voici une liste des codes de retour fournis par *clamscan* :

0 : aucun virus trouvé.
1 : virus détecté(s).
40 : option inconnue passée à *clamscan*. Utilisez *clamscan –help* ou la page du manuel pour obtenir les options disponibles.
50 : erreur d'initialisation de la base de données des virus. Il n'existe probablement pas de base par défaut ou elle est corrompue (ex : signature digitale corrompue).
52 : type de fichier non supporté ; clamscan supporte uniquement les fichiers réguliers, les répertoires et les liens symboliques.
53 : ne peut ouvrir le répertoire.
54 : ne peut ouvrir le fichier [6]
55 : erreur I/O lors de la lecture [7]
56 : ne peut statuer sur le fichier d'entrée ou le répertoire. Le fichier (ou le répertoire) que vous voulez analyser n'existe pas.
57 : ne peut obtenir le chemin absolu du répertoire de travail courant. Votre chemin courant est plus long que 200 caractères. Vous devriez recompiler ClamAV pour corriger ceci.
58 : erreur I/O. Veuillez vérifier le système de fichiers.
59 : impossible d'obtenir les informations de l'utilisateur courant (faisant fonctionner clamscan).
60 : impossible d'obtenir les informations de l'utilisateur *clamscan*. L'utilisateur *clamscan* (utilisateur par défaut sans privilèges) n'existe pas dans /etc/passwd.
61 : impossible de forker. Impossible de créer un nouveau processus, veuillez vérifier les limites de votre système.
63 : impossible de créer un fichier ou un répertoire temporaire. Veuillez vérifier les permissions de /tmp ou utilisez -tempdir.
64 : impossible d'écrire dans le répertoire temporaire. Veuillez en spécifier un autre.
70 : impossible d'allouer ou de vider la mémoire. C'est une erreur critique, veuillez vérifier votre système.
71 : impossible d'allouer de la mémoire. Voir ci-dessus.

[6] Seulement en mode fichier par fichier ; cette erreur est ignorée en mode récursif
[7] Seulement en mode fichier par fichier ; cette erreur est ignorée en mode récursif

5 Logiciels certifiés

Il y a de nombreux projets qui supportent ClamAV. Voici une liste des logiciels qui ont été testés et qui sont connus pour bien fonctionner.

5.1 clamav-milter

Page d'accueil : inclus dans le package de clamav
Supporte : clamd
clamav-milter de Nigel Horne est un scanner d'email très rapide destiné à sendmail. Il est écrit entièrement en C et utilise le scanner de mail interne de ClamAV (aussi écrit par Nigel)
Installation :
Vous avez besoin des fichiers de développement libmilter. Configurez ClamAV avec :

```
$ ./configure --enable-milter
```

et recompilez. Le programme sera installé dans /usr/local/sbin/clamav-milter. Les instructions suivantes ont été adaptées à partir du fichier INSTALL de Nigel. Ajoutez au fichier /etc/mail/sendmail.mc :

```
INPUT\_MAIL\_FILTER('clmilter','S=local:/var/run/clmilter.sock,
F=,T=S:4m;R:4m')dnl
define('confINPUT\_MAIL\_FILTERS', 'clmilter')
```

Vérifiez les entrées dans clamav.conf qui ont la forme suivante :

```
LocalSocket /var/run/clamd.sock
ScanMail
StreamSaveToDisk
```

Lancez clamav-milter :

```
/usr/local/sbin/clamav-milter -blo /var/run/clmilter.sock
```

et relancez sendmail

5.2 IVS Milter

Page d'accueil : `http ://ivs-milter.lsbd.net`
Supporte : clamd
IVS Milter permet à la fois de scanner les virus et les pourriels. Le nom est issu de Industrial Virus + Spam milter. Il s'applique aussi bien aux simples utilisateurs qu'aux grands fournisseurs d'accès internet.

5.3 smtp-vilter

Page d'accueil : `http ://www.etc.msys.ch/software/smtp-vilter`
Supporte : clamd
smtp-vilter est un filtre de contenu très performant pour sendmail utilisant l'API milter. Le logiciel scanne les emails à la recherche de virus et rejette ou marque les messages infectés. ClamAV est le scanner par défaut.

5.4 mod_clamav

Page d'accueil : `http ://software.othello.ch/mod_clamav`
Supporte : libclamav, clamd
mod_clamav est un filtre antivirus pour Apache. Il a été écrit et est actuellement maintenu par Andreas Muller. Le projet est très bien documenté et l'installation est très facile.

5.5 TrashScan

Page d'accueil : `clamav-sources/support/trashscan`
Supporte : clamscan
C'est un scanner basé sur procmail développé par Trashware et est extrèmement simple à configurer bien que ce ne soit que pour des mono-utilisateurs uniquement et moins efficace que les scanners basés sur MTA.

5.6 AMaViS - "Next Generation"

Page d'accueil : `http ://www.sourceforge.net/projects/amavis`
Supporte : clamscan
AMaViS-ng est une version réécrite et plus modulaire de amavis-perl/amavisd développée par Hilko Bengen.
Installation :
Veuillez télécharger la nouvelle version (au moins 0.1.4). Après installation (ce qui est plutôt simple), veuillez décommenter les lignes suivantes dans le fichier amavis.conf :

```
virus-scanner = CLAM
```

et éventuellement changer le chemin de clamscan dans la section *[CLAM]* :

```
[CLAM]
clamscan = /usr/local/bin/clamscan
```

5.7 amavisd-new

Page d'accueil : `http ://www.ijs.si/software/amavisd`
Supporte : clamd, clamscan
amavisd-new est une version réécrite d'amavis et maintenue par Mark Martinec
Installation :
clamscan est automatiquement activé si le fichier bianire clamscan est trouvé au démarrage d'amavisd-new. clamd est activé en décommentant son entrée dans la liste @av_scanners dans le fichier /etc/amavisd.conf.

5.8 Qmail-Scanner

Page d'accueil : `http ://qmail-scanner.sf.net`
Supporte : clamscan
Vous devez augmenter la valeur softlimit ou attendre un support du démon.

5.9 Sagator

Page d'accueil : `http ://www.salstar.sk/sagator`
Supporte : clamscan, clamd, libclamav
Ce programme est une passerelle antivirus/antispam pour emails. C'est une interface à

postfix (ou à un quelconque smtpd) qui dispose d'un antivirus et/ou d'un antispam. Son architecture modulaire peut utiliser n'importe quelle combinaison d'antivirus/antispam selon la configuration.

5.10 ClamdMail

Page d'accueil : `http ://clamdmail.sf.net`
Supporte : clamd
Un client mail pour ClamAV. Rapide, petit et facile à installer.

5.11 BlackHole

Page d'accueil : `http ://www.groovy.org/blackhole.shtml`
Supporte : clamscan, clamd
BlackHole est un filtre spam/virus pour Qmail, Postfix, Sendmail, Exim et Courier écrit par Chris Kennedy. Cet outil est destiné aux administrateurs avancés (l'installation est difficile).

5.12 MailScanner

Page d'accueil : `http ://www.mailscanner.info`
Supporte : clamscan
Mailscanner scanne tous les emails à la recherche de virus, de spam et s'attaque aux vulnérabilités de sécurité. Il n'est lié à aucun antivirus particulier mais il peut être utilisé en associations avec 14 antivirus différents permettant aux sites de choisir leur "meilleur" antivirus.

5.13 MIMEDefang

Page d'accueil : `http ://www.roaringpenguin.com/mimedefang`
Supporte : clamscan, clamd
C'est un scanner de mails efficace pour Sendmail/milter.

5.14 exiscan

Page d'accueil : `http ://duncanthrax.net/exiscan`
Supporte : clamscan, clamd
exiscan est un patch pour exim version 4, permettant de scanner les emails reçus par exim. Quatre méthodes différentes sont disponibles : antivirus, antispam, expressions régulières et extension des fichiers.

5.15 scanexi

Page d'accueil : `http ://w1.231.telia.com/~u23107873/scanexi.html`
Supporte : clamscan, clamd
scanexi est un plugin pour exim version4.14 avec un patch dlopen, permettant de scanner le contenu des emails reçus par exim.

5.16 Mail : :ClamAV

Page d'accueil : `http ://cpan.gossamer-threads.com/modules/by-authors/id/S/SA/SABECK`
Supporte : libclamav
Extension Perl pour la librairie libclamav.

5.17 OpenAntiVirus samba-vscan

Page d'accueil : `http ://www.openantivirus.org/projects`
Supporte : clamd
samba-vscan permet de tester les virus lors de l'accès aux partages de Samba. Il supporte Samba 2.2.x/3.0 ainsi que les système de fichiers virtuels (VFS).

5.18 Sylpheed Claws

Page d'accueil : `http ://claws.sylpheed.org`
Supporte : libclamav
Sylpheed Claws est une branche de Sylpheed, un agent mail léger pour UNIX. Il scanne les pièces attachées aux emails reçus d'un compte POP et éventuellement détruit le mail ou le sauvegarde dans un répertoire spécifié.

5.19 nclamd

Page d'accueil : `http ://www.kyzo.com/nclamd`
Supporte : libclamav
nclamd, nclamav-milter et nclamdscan sont des versions réécrites des outils originaux et utilise des processus au lieu des threads ainsi que ripMIME au lieu du décodeur MIME interne de clamav.

5.20 cgpav

Page d'accueil : `http ://program.farit.ru`
Supporte : clamd
C'est un plugin antivirus rapide (écrit en C) de CommuniGate Pro qui supporte clamd.

6 LibClamAV

libclamav peut être utilisé pour ajouter une protection contre les virus à vos logiciels. La librairie est "thread-safe", reconnaît automatiquement et scanne les archives. L'analyse est très rapide et dans la plupart des cas est imperceptible.

6.1 API Générale

Chaque programme basé sur libclamav doit inclure le fichier d'en-tête *clamav.h* :

```
#include <clamav.h>
```

La première étape consiste à initialiser le moteur d'analyse. Il y a trois fonctions disponibles :

```
int cl_loaddb(const char *filename, struct cl_node **root,int *virnum);
int cl_loaddbdir(const char *dirname, struct cl_node **root,int *virnum);
char *cl\_retdbdir(void);
```

cl_loaddb() charge une base de données particulière, *cl_loaddbdir()* charge toutes les bases de données *.cvd* (et les anciennes *.db*, *.db2*) à partir d'un répertoire *dirname*. *cl_retdbdir()* renvoie le chemin d'un répertoire d'une base de donnée. La base de données initiale interne (arbre Aho-Corasick, voir 6.3) sera sauvegardée sous le compte *root* et un certain nombre de signature seront **ajoutées** [8] à *virnum*. Le pointeur vers trie doit être initialisé à NULL. Si vous ne voulez pas vous préoccuper du nombre de signatures, passez NULL comme troisième argument. Les fonctions de *cl_loaddb* retournent 0 si succès et une autre valeur si échec.

```
struct cl_node *root = NULL;
int ret;
ret = cl_loaddbdir(cl_retdbdir(), &root, NULL);
```

Il y a une manière élégante de faire afficher les codes d'erreurs de libclamav :

```
char *cl_strerror(int clerror);
```

cl_strerror retourne une chaîne (allouée de façon statique) décrivant un code *clerror* :

```
if(ret) {
    printf("cl_loaddbdir() error: %s\n", cl_strerror(ret));
    exit(1);
}
```

Lorsque la base de données est chargée, vous devez le arbre final avec :

```
void cl_buildtrie(struct cl_node *root);
```

Dans notre exemple :

```
cl_buildtrie(root);
```

Maintenant, vous pouvez scanner un buffer, un descripteur ou un fichier avec :

[8] Pensez à initialiser la variable du compteur de virus à 0

```
int cl_scanbuff(const char *buffer, unsigned int length,
char **virname, const struct cl_node *root);

int cl_scandesc(int desc, char **virname, unsigned long int
*scanned, const struct cl_node *root, const struct cl_limits
*limits, int options);

int cl_scanfile(const char *filename, char **virname,
unsigned long int *scanned, const struct cl_node *root,
const struct cl_limits *limits, int options);
```

Toutes les fonctions sauvegardent une adresse du nom du virus via le pointeur *virname*. *virname* pointe vers un nom dans la sructure arbre et ainsi on ne peut y accéder directement. *cl_scandesc()* et *cl_scanfile()* peuvent augmenter la valeur *scanned* dans les unités CL_COUNT_PRECISION, et supportent aussi une limite de taille d'archives :

```
struct cl_limits {
    int maxreclevel;
    int maxfiles;
    long int maxfilesize;
};
```

Le dernier argument configure le moteur d'analyse. Il supporte actuellement **CL_ARCHIVE** (permet de scanner des archives), **CL_RAW** (désactive l'analyse des archives), **CL_MAIL** (permet de scanner mbox et Maildir) et **CL_DISABLERAR** (désactive le décompresseur interne des archives RAR). Ces fonctions retournent le code 0 (**CL_CLEAN**) lorsqu'aucun virus n'est trouvé, **CL_VIRUS** lorsqu'un virus est découvert et une autre valeur en cas d'échec.

```
struct cl_limits limits;
char *virname;

/* maximal number of files in archive */;
limits.maxfiles = 1000
/* maximal archived file size == 10 MB */
limits.maxfilesize = 10 * 1048576;
/* maximal recursion level */
limits.maxreclevel = 5;

if((ret = cl_scanfile("/home/zolw/test", &virname, NULL, root,
&limits, CL_ARCHIVE)) == CL_VIRUS) {
    printf("Detected %s virus.\n", virname);
} else {
    printf("No virus detected.\n");
    if(ret != CL_CLEAN)
        printf("Error: %s\n", cl_strerror(ret));
}
```

Libérez le arbre si vous n'en avez plus besoin :

```
void cl_freetrie(struct cl_node *root);
```

Vous trouverez un exemple de scanner dans les sources de clamav (/example). Les programmes basés sur libclamav doivent être liées comme ceci :

```
gcc -Wall ex1.c -o ex1 -lclamav
```

Profitez-en !

6.2 Recharger la base de données

Il est essentiel de conserver la représentation de la base de données à jour. Vous pouvez observer les changements de la base de données grâce à la famille de fonctions *cl_stat* :

```
int cl_statinidir(const char *dirname, struct cl_stat *dbstat);
int cl_statchkdir(const struct cl_stat *dbstat);
int cl_statfree(struct cl_stat *dbstat);
```

Initialisation :

```
struct cl_stat dbstat;
memset(&dbstat, 0, sizeof(struct cl_stat));
cl_statinidir(dbdir, &dbstat);
```

Pour chercher un changement, vous n'avez qu'à appeler la fonction suivante :

```
if(cl_statchkdir(&dbstat) == 1) {
    reload_database...;
    cl_statfree(&dbstat);
    cl_statinidir(cl_retdbdir(), &dbstat);
}
```

Pensez à réinitialiser la structure après un rechargement.

6.3 Moteur d'analyse

Les nouvelles version de Clam AntiVirus utilisent une modification de l'algorithme de pattern matching Aho-Corasick. L'algorithme est basé sur un automate de pattern matching à états finis [1] et est une généralisation du fameux algorithme de Knuth-Morris-Pratt. Veuillez jetter un oeil aux définitions des types de données dans *matcher.h*. L'automate est représenté par un arbre. C'est un arbre racinaire avec quelques propriétés particulières [2]. Chaque noeud de l'arbre représente un état de l'automate. Dans notre implémentation, le noeud est défini de la manière suivante :

```
    struct cl_node {
    short int islast;
    struct cli_patt *list;
    int maxpatlen;
    struct node *next[NUM_CHILDS], *trans[NUM_CHILDS], *fail;
};
```

[A terminer ...]

6.4 Le format CVD

CVD (ClamAV Virus Database) est un fichier tar signé numériquement qui contient une ou plusieurs bases de données. Vous pouvez trouver un certain nombre d'informations utiles dans l'en-tête ASCII du fichier. Il s'agit un chaîne longue de 512 bytes contenant des colonnes avec les champs suivants :

```
ClamAV-VDB:build time:version:number of signatures:functionality
level required:MD5 checksum:digital signature:builder name
```

Elle peut être facilement parcourue par des scripts ou par *sigtool –info*. Il y a deux bases CVD dans ClamAV : *main.cvd* et *daily.cvd* pour les mises à jour quotidiennes. Vous pouvez utiliser *sigtool* pour décompresser un fichier CVD.

7 Credits

Dans l'odre alphabétique :

- AIX PDSLIB, University of California at Los Angeles http ://aixpdslib.seas.ucla.edu - binary packages for AIX
- Kamil Andrusz <wizz*mniam.net> - OpenBSD support patch
- Marc Baudoin <babafou*babafou.eu.org> - NetBSD testing
- Hilko Bengen <bengen*vdst-ka.inka.de> - support for Clam AntiVirus in his AMaViS - "Next Generation"
- Patrick Bihan-Faou <patrick*mindstep.com> - support for -with-user/group in the configure script.
- Eric I. Lopez Carreon <elopezc*technitrade.com> - Spanish "Sendmail + AMaViS + ClamAV Installation" how-to
- Nicholas Chua <nicholas*ncmbox.net> - big virus submissions and signatures
- Christoph Cordes <ib*precompiled.de> - big virus submissions.
- Damien Curtain <damien*pagefault.org> - fix for the -remove option in clamscan (it didn't work with internal archivers) ; implementation of the -move option in clamscan, mirroring support in freshclam.
- Krisztian Czako <slapic*linux.co.hu> - virus signatures.
- Diego d'Ambra <da*softcom.dk> - Database developer.
- Michael Dankov <misha*btrc.ru> - clamd fixes
- Alejandro Dubrovsky <s328940*student.uq.edu.au> - patch for including and excluding multiple patterns.
- Magnus Ekdahl <magnus*debian.org> - Debian (http ://www.debian.org) package maintainer ; fixes and improvements.
- Jason Englander <jason*englanders.cc> - bug report : problem with clamd recursive scanning of directories on non standard file systems ; configure script support for id checking. Database developer.
- Oden Eriksson <oden.eriksson*kvikkjokk.net> - Mandrake package maintainer.
- Edison Figueira Junior <edison*brc.com.br> - money donation.
- David Ford <david+cert*blue-labs.org> - gcc 3.x support fix.
- Free Oscar <freeoscar*wp.pl> - hex2str() enhancement
- Piotr Gackiewicz <gacek*intertele.pl> - bug report : clamd THREXIT bug
- Jeremy Garcia <jeremy*linuxquestions.org> - help with Affero.net account.
- Nick Gazaloff <nick*sbin.org> - socket descriptors leak fix in clamd.
- Luca 'NERvOus' Gibelli <nervous*nervous.it> - ElektraPro.com administrator.
- Wieslaw Glod <wkg*x2.pl> - bug report : FreeBSD compile problem in 0.22.
- Matthew A. Grant <grantma*anathoth.gen.nz> - OpenAntiVirus Update script (oav-update)
- Hrvoje Habjanic <hrvoje.habjanic*zg.hinet.hr> - syslog support patch for clamd ; virus provider.
- Michal Hajduczenia <michalis*mat.uni.torun.pl> - old clam title logo.
- Paul Hoadley <paulh*logixsquad.net> - "Installing qmail-scanner, Clam AntiVirus and SpamAssassin under FreeBSD" how-to.
- Thomas W. Holt Jr. <twh*cohesive.net> - information about ClamAV compiling on Solaris 2.6 and Cobalt MIPS boxes.
- Douglas J Hunley <doug*hunley.homeip.net> - clamav.linux-sxs.org mirror (no longer active).
- Kurt Huwig <kurt*iku-netz.de> - smart suggestions, ScannerDaemon (OpenAntiVirus) author.

- Dave Jones <dave*kalkbay.co.za> - bug report : problem in option parser.
- Kazuhiko <kazuhiko*fdiary.net> - Qmail-Scanner 0.12 support patch.
- Robbert Kouprie <robbert*exx.nl> - patch for unrarlib buffer overflow.
- Henk Kuipers <henk*opensourcesolutions.nl> - bug report : 0.50 compile problem.
- Nigel Kukard <nkukard*lbsd.net> - virus signatures.
- Dr Andrzej Kurpiel <akurpiel*mat.uni.torun.pl> - choice of this project from my list.
- Thomas Lamy <Thomas.Lamy*in-online.net> - code enhancements.
- Dennis Leeuw <dleeuw*made-it.com> - "Debian GNU/Linux Mail Server" howto, corrections of this document.
- Martin Lesser <admin-debian*bettercom.de> - patch for the http-proxy problem in 0.51.
- Peter N Lewis <peter*stairways.com.au> - Mac OS X data type problem bug-fix.
- Mike Loewen <mloewen*sturgeon.cac.psu.edu> - bug report : clamscan 0.24 compile error on Solaris 8 ; various Solaris and AIX tips.
- Thomas Madsen <tm*softcom.dk> - submission management interface (only for developers).
- Stefan Martig <sm*officeco.ch> - bug report : /proc/cpuinfo problem analysis on Linux/Alpha, providing me with access to the Linux/Alpha system.
- Brian May <bam*debian.org> - bug report : clamd writing to an undefined file.
- Ken McKittrick <klmac*usadatanet.com> - intensive FreeBSD testing, hdd donation.
- Chris van Meerendonk <cvm*castel.nl> - virus samples, clamav.essentkabel.com mirror.
- Arkadiusz Miskiewicz <misiek*pld.org.pl> - Polish(ed) Linux Distribution (http ://www.pld.org.pl) rpm package maintainer ; fixes and ideas.
- Mark Mielke <mark*mark.mielke.cc> - code enhancements, bug reports.
- Jo Mills <Jonathan.Mills*frequentis.com> - great bug reports.
- Doug Monroe <doug*planetconnect.com> - Qmail-Scanner problem analysis.
- Flinn Mueller<flinn*activeintra.net> - OpenBSD port maintainer.
- Hendrik Muhs <Hendrik.Muhs*student.uni-magdeburg.de> - pattern matcher optimization.
- Farit Nabiullin http ://program.farit.ru - big virus submissions.
- Wojciech Noworyta <wnow*konarski.edu.pl> - bug report : buffer overflow in clamscan's help under Windows.
- Joe Oaks <joe.oaks*hp.com> - HPUX support.
- Washington Odhiambo <wash*wananchi.com> - extensive mbox code testing, bug reports.
- Masaki Ogawa <proc*mac.com> - Mac OS X support, Japanese documentation.
- Martijn van Oosterhout <kleptog*svana.org> - code analysis and suggestions.
- OpenAntiVirus.org Team - initial virus database.
- Tomasz Papszun <tomek*lodz.tpsa.pl> - various bug reports and ideas. Database developer.
- Eric Parsonage <eric*eparsonage.com> - "Installing qmail-scanner, Clam AntiVirus and SpamAssassin under FreeBSD" how-to.
- Oliver Paukstadt <pstadt*stud.fh-heilbronn.de> - bug report : crash with strange Zip archives.
- Kristof Petr <Kristof.P*fce.vutbr.cz> - bug report : socket descriptors leak in clamd ; file decriptors leak in clamd, clamscan and libclamav.
- Ed Phillips <ed*UDel.Edu> - patch for the internal logger in clamd ; ideas and suggestions.

- Andreas Piesk <Andreas.Piesk*heise.de> - clamd : old ScannerDaemonOutputFormat option.
- Ant La Porte <ant*dvere.net> - proxy support enhancement.
- Sergei Pronin <sp*finndesign.fi> - bug report : access problems in superuser mode.
- Thomas Quinot <thomas*cuivre.fr.eu.org> - patch for non-default prefix and incoherent database location specification in defaults.h of clamscan and freshclam.
- David Sanchez <dsanchez*veloxia.com> - bug report : thread deadlocking in a critical error situation.
- Martin Schitter - bug report : libclamav crash on certain zip files.
- Enrico Scholz <enrico.scholz*informatik.tu-chemnitz.de> - daemonize() enhancements.
- Dr Zbigniew Szewczak <zssz*mat.uni.torun.pl> - ideas, suggestions and time spent on discussing some aspects of ClamAV.
- Matt Sullivan <matt*sullivan.gen.nz> - clamav-milter fixes.
- Joe Talbott <josepht*cstone.net> - clamav-milter enhancements.
- Gernot Tenchio <g.tenchio*telco-tech.de> - proxy authorization support in freshclam.
- Masahiro Teramoto <markun*onohara.to> - official FreeBSD port maintainer.
- Trashware <trashware*gmx.net> - TrashScan
- Laurent Wacrenier <lwa*teaser.fr> - mbox fixes
- NikolajWicker <n.wicker*cnk-networks.de> - sponsored Solaris 9 (i386) for Nigel (for clamav-milter development purposes).
- David Woakes <david*mitredata.co.uk> - freshclam -on-error-execute fix.
- Troy Wollenslegel <troy*intranet.org> - bug report : handling inaccessible directories in archives.
- Andoni Zubimendi <andoni*lpsat.net> - fix for a segmentation fault in 0.12 (NULL pointer dereference).

8 Auteurs

8.1 Développeurs de la base des virus

La base des virus constitue le coeur de tous les logiciels d'antivirus. Les personnes suivantes prennent soin du coeur de ClamAV :

- aCaB <acab*clamav.net>
- Diego D'Ambra <diego*clamav.net>
- Jason Englander <jason*clamav.net>
- Tomasz Kojm <tkojm*clamav.net>
- Tomasz Papszun <tomek*clamav.net>

Notre base inclut la base (près de 5000 signatures) des virus d'OpenAntiVirus.org

8.2 Gestion du réseau

Grâce à Luca 'NERvOus' Gibelli <nervous*clamav.net> vous pouvez télécharger notre base à partir des miroirs listés au paragraphe 2.8. Luca est aussi responsable de notre site principal `www.clamav.net`, des mailing listes et du méchanisme de sousmission des virus.

8.3 Graphiques

Les auteurs du magnifique logo ClamAV (voir la page de couverture) sont Mia Kalenius et Sergei Pronin <sp*finndesign.fi>.

8.4 Développeurs principaux

Nigel Horne <njh*clamav.net> est un développeur actif de ClamAV responsable du code mbox et de clamav-milter. Tomasz Kojm (moi) gère le projet et garde un oeil sur tout 8-).

Tomasz Kojm <tkojm*clamav.net>

Références

[1] Cormen, Leiserson, Rivest : Introduction to Algorithms, Chapter 34, MIT Press.

[2] http ://www-sr.informatik.uni-tuebingen.de/~buehler/AC/AC.html : Aho-Corasick algorithm description