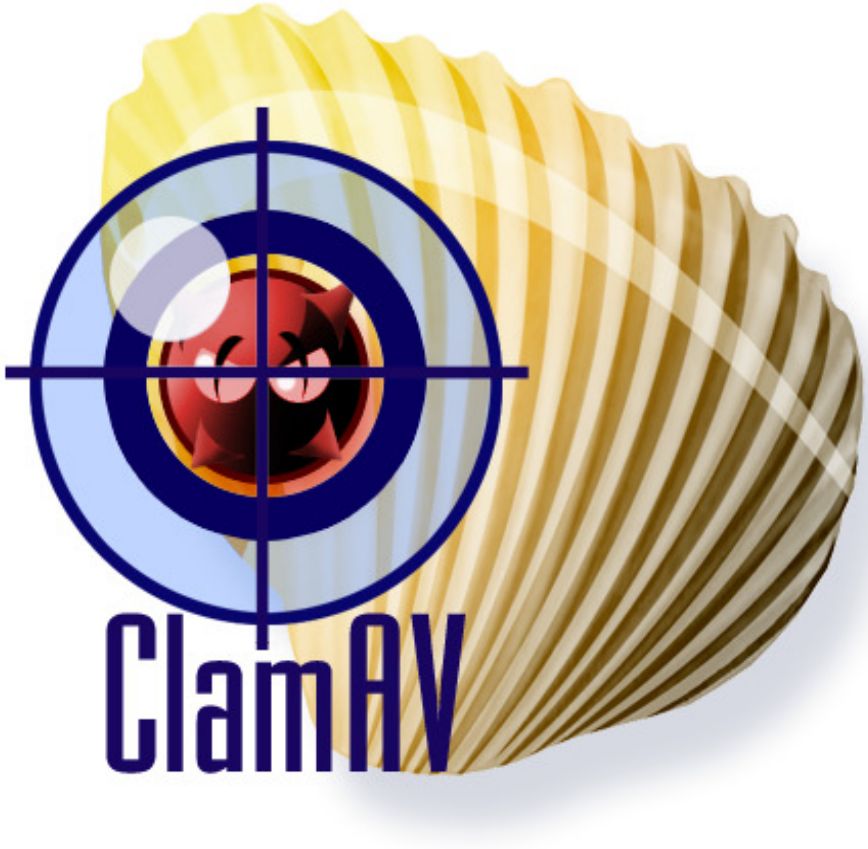

Clam AntiVirus 0.94.1

User Manual

Contents

1 Introduction

Clam AntiVirus is an open source (GPL) anti-virus toolkit for UNIX, designed especially for e-mail scanning on mail gateways. It provides a number of utilities including a flexible and scalable multi-threaded daemon, a command line scanner and advanced tool for automatic database updates. The core of the package is an anti-virus engine available in a form of shared library.

1.1 Features

- Licensed under the GNU General Public License, Version 2
- POSIX compliant, portable
- Fast scanning
- Supports on-access scanning (Linux and FreeBSD only)
- Detects over 450.000 viruses, worms and trojans, including Microsoft Office macro viruses, mobile malware, and other threats
- Scans within archives and compressed files (also protects against archive bombs), built-in support includes:
  - Zip (including SFX)
  - RAR (including SFX)
  - ARJ (including SFX)
  - Tar
  - Gzip
  - Bzip2
  - MS OLE2
  - MS Cabinet Files (including SFX)
  - MS CHM (Compiled HTML)
  - MS SZDD compression format
  - BinHex
  - SIS (SymbianOS packages)
  - AutoIt
- Supports Portable Executable (32/64-bit) files compressed or obfuscated with:

  - AsPack
  - UPX
  - FSG
  - Petite
  - PeSpin
  - NsPack
  - wwpack32
  - MEW
  - Upack
  - Y0da Cryptor
- Supports almost all mail file formats
- Support for other special files/formats includes:
  - HTML
  - RTF
  - PDF
  - Files encrypted with CryptFF and ScrEnc
  - uuencode
  - TNEF (winmail.dat)
- Advanced database updater with support for scripted updates, digital signatures and DNS based database version queries

1.2 Mailing lists and IRC channel

If you have a trouble installing or using ClamAV try asking on our mailing lists. There are four lists available:

- **clamav-announce*lists.clamav.net** - info about new versions, moderated[1].
- **clamav-users*lists.clamav.net** - user questions
- **clamav-devel*lists.clamav.net** - technical discussions
- **clamav-virusdb*lists.clamav.net** - database update announcements, moderated

[1] Subscribers are not allowed to post to the mailing list

You can subscribe and search the mailing list archives at: http://www.clamav.net/support/ml/
Alternatively you can try asking on the #clamav IRC channel - launch your favourite irc client and type:

```
/server irc.freenode.net
/join #clamav
```

1.3 Virus submitting

If you have got a virus which is not detected by your ClamAV with the latest databases, please submit the sample at our website:

http://www.clamav.net/sendvirus

2 Base package

2.1 Supported platforms

Most popular UNIX operating systems are supported. Clam AntiVirus 0.9x was tested on:

- GNU/Linux
- Solaris
- FreeBSD
- OpenBSD [2]
- Mac OS X

Some features may not be available on your operating system. If you are successfully running Clam AntiVirus on a system not listed above please let us know.

2.2 Binary packages

You can find the up-to-date list of binary packages at our website: http://www.clamav.net/download/packages/

[2] Installation from a port is recommended.

3 Installation

3.1 Requirements

The following elements are required to compile ClamAV:

- zlib and zlib-devel packages
- gcc compiler suite (tested with 2.9x, 3.x and 4.x series)
  If you are compiling with higher optimization levels than the default one (-O2 for gcc), be aware that there have been reports of misoptimizations. The build system of ClamAV only checks for bugs affecting the default settings, it is your responsibility to check that your compiler version doesn't have any bugs.

The following packages are optional but **highly recommended**:

- bzip2 and bzip2-devel library
- GNU MP 3
  It's very important to install the GMP package because it allows `freshclam` to verify the digital signatures of the virus databases and scripted updates. If freshclam was compiled without GMP support it will display "SECURITY WARNING: NO SUPPORT FOR DIGITAL SIGNATURES" on every update. You can download GNU MP at `http://www.swox.com/gmp/`
  A note for Solaris/SPARC users: you must set the *ABI* system variable to 32 (e.g. `setenv ABI 32`) before running the configuration script of GMP.
- `check` unit testing framework [3].

3.2 Installing on shell account

To install ClamAV locally on an unprivileged shell account you need not create any additional users or groups. Assuming your home directory is `/home/gary` you should build it as follows:

```
$ ./configure --prefix=/home/gary/clamav --disable-clamav
$ make; make install
```

To test your installation execute:

[3] See section 3.6 on how to run the unit tests

```
$ ~/clamav/bin/freshclam
$ ~/clamav/bin/clamscan ~
```

The `--disable-clamav` switch disables the check for existence of the *clamav* user and group but `clamscan` would still require an unprivileged account to work in a superuser mode.

3.3 Adding new system user and group

If you are installing ClamAV for the first time, you have to add a new user and group to your system:

```
# groupadd clamav
# useradd -g clamav -s /bin/false -c "Clam AntiVirus" clamav
```

Consult a system manual if your OS has not *groupadd* and *useradd* utilities. **Don't forget to lock access to the account!**

3.4 Compilation of base package

Once you have created the clamav user and group, please extract the archive:

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

Assuming you want to install the configuration files in /etc, configure and build the software as follows:

```
$ ./configure --sysconfdir=/etc
$ make
$ su -c "make install"
```

In the last step the software is installed into the /usr/local directory and the config files into /etc. **WARNING: Never enable the SUID or SGID bits for Clam AntiVirus binaries.**

3.5 Compilation with clamav-milter enabled

libmilter and its development files are required. To enable clamav-milter, configure ClamAV with

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

3.6 Running unit tests

ClamAV includes unit tests that allow you to test that the compiled binaries work correctly on your platform.

The first step is to use your OS's package manager to install the `check` package. If your OS doesn't have that package, you can download it from `http://check.sourceforge.net/`, build it and install it.

To help clamav's configure script locate `check`, it is recommended that you install `pkg-config`, preferably using your OS's package manager, or from `http://pkg-config.freedesktop.org`.

The recommended way to run unit-tests is the following, which ensures you will get an error if unit tests cannot be built: [4]

```
$ ./configure --enable-check
$ make
$ make check
```

When `make check` is finished, you should get a message similar to this:

```
==================
All 5 tests passed
==================
```

If a unit test fails, you get a message similar to the following. See the next section on how to report a bug when a unit test fails.

```
========================================
1 of 5 tests failed
Please report to http://bugs.clamav.net/
========================================
```

If unit tests are disabled (and you didn't use –enable-check), you will get this message:

[4] The configure script in ClamAV automatically enables the unit tests, if it finds the check framework, however it doesn't consider it a fatal error if unit tests cannot be enabled.

```
*** Unit tests disabled in this build
*** Use ./configure --enable-check to enable them

SKIP: check_clamav
PASS: check_clamd.sh
PASS: check_freshclam.sh
PASS: check_sigtool.sh
PASS: check_clamscan.sh
=====================
All 4 tests passed
(1 tests were not run)
=====================
```

Running `./configure --enable-check` should tell you why.

3.7 Reporting a unit test failure bug

If `make check` says that some tests failed we encourage you to report a bug on our bugzilla: `http://bugs.clamav.net`. The information we need is (see also `http://clamav.net/bugs`):

- The exact output from `make check`
- Output of `uname -mrsp`
- your `config.log`
- The following files from the `unit_tests/` directory:
  - `test.log`
  - `clamscan.log`
  - `clamdscan.log`
- `/tmp/clamd-test.log` if it exists
- where and how you installed the check package
- Output of `pkg-config check --cflags --libs`
- Optionally if `valgrind` is available on your platform, the output of the following:

```
$ make check
$ CK_FORK=no ./libtool --mode=execute valgrind unit_tests/check-clamav
```

4 Configuration

4.1 clamd

Before you start using the daemon you have to edit the configuration file (in other case `clamd` won't run):

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

This shows the location of the default configuration file. The format and options of this file are fully described in the *clamd.conf(5)* manual. The config file is well commented and configuration should be straightforward.

4.1.1 On-access scanning

One of the interesting features of `clamd` is on-access scanning based on the Dazuko module, available from `http://dazuko.org/`. **This module is not required to run clamd - furthermore, you shouldn't run Dazuko on production systems**. At the moment Dazuko is avaliable for Linux and FreeBSD, but the following information only covers Linux.

```
$ tar zxpvf dazuko-a.b.c.tar.gz
$ cd dazuko-a.b.c
$ make dazuko
or
$ make dazuko-smp (for smp kernels)
$ su
# insmod dazuko.o
# cp dazuko.o /lib/modules/`uname -r`/misc
# depmod -a
```

Depending on your Linux distribution you may need to add a "dazuko" entry to */etc/modules* or run the module during system's startup by adding

```
/sbin/modprobe dazuko
```

to some startup file. You must also create a new device:

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

Now configure Clamuko in `clamd.conf` and read the 5.3 section.

4.2 clamav-milter

Nigel Horne's `clamav-milter` is a very efficient email scanner designed for Sendmail. It's written entirely in C and only depends on `libclamav` or `clamd`. You can find detailed installation instructions in the `INSTALL` file that comes with the clamav-milter sources. Basically, to connect it with Sendmail add the following lines to `/etc/mail/sendmail.mc`:

```
INPUT_MAIL_FILTER('clmilter','S=local:/var/run/clamav/clmilter.sock,
F=, T=S:4m;R:4m')dnl
define('confINPUT_MAIL_FILTERS', 'clmilter')
```

If you're running it in `--external` mode, check entry in `clamd.conf` of the form:

```
LocalSocket /var/run/clamav/clamd.sock
```

Start clamav-milter

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

and restart sendmail.

4.3 Testing

Try to scan recursively the source directory:

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

It should find some test files in the clamav-x.yz/test directory. The scan result will be saved in the `scan.txt` log file [5]. To test `clamd`, start it and use `clamdscan` (or instead connect directly to its socket and run the SCAN command):

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

Please note that the scanned files must be accessible by the user running `clamd` or you will get an error.

[5] To get more info on clamscan options run 'man clamscan'

4.4 Setting up auto-updating

`freshclam` is the automatic database update tool for Clam AntiVirus. It can work in two modes:

- interactive - on demand from command line
- daemon - silently in the background

`freshclam` is advanced tool: it supports scripted updates (instead of transferring the whole CVD file at each update it only transfers the differences between the latest and the current database via a special script), database version checks through DNS, proxy servers (with authentication), digital signatures and various error scenarios. **Quick test: run freshclam (as superuser) with no parameters and check the output.** If everything is OK you may create the log file in /var/log (owned by *clamav* or another user `freshclam` will be running as):

```
# touch /var/log/freshclam.log
# chmod 600 /var/log/freshclam.log
# chown clamav /var/log/freshclam.log
```

Now you *should* edit the configuration file `freshclam.conf` and point the *UpdateLogFile* directive to the log file. Finally, to run `freshclam` in the daemon mode, execute:

```
# freshclam -d
```

The other way is to use the *cron* daemon. You have to add the following line to the crontab of **root** or **clamav** user:

```
N * * * * /usr/local/bin/freshclam --quiet
```

to check for a new database every hour. **N should be a number between 3 and 57 of your choice. Please don't choose any multiple of 10, because there are already too many clients using those time slots.** Proxy settings are only configurable via the configuration file and `freshclam` will require strict permission settings for the config file when `HTTPProxyPassword` is turned on.

```
HTTPProxyServer myproxyserver.com
HTTPProxyPort 1234
HTTPProxyUsername myusername
HTTPProxyPassword mypass
```

4.4.1 Closest mirrors

The `DatabaseMirror` directive in the config file specifies the database server `freshclam` will attempt (up to `MaxAttempts` times) to download the database from. The default database mirror is `database.clamav.net` but multiple directives are allowed. In order to download the database from the closest mirror you should configure `freshclam` to use `db.xx.clamav.net` where xx represents your country code. For example, if your server is in "Ascension Island" you should have the following lines included in `freshclam.conf`:

```
DNSDatabaseInfo current.cvd.clamav.net
DatabaseMirror db.ac.clamav.net
DatabaseMirror database.clamav.net
```

The second entry acts as a fallback in case the connection to the first mirror fails for some reason. The full list of two-letters country codes is available at `http://www.iana.org/cctld/cctld-whois.htm`

5 Usage

5.1 Clam daemon

`clamd` is a multi-threaded daemon that uses *libclamav* to scan files for viruses. It may work in one or both modes listening on:

- Unix (local) socket
- TCP socket

The daemon is fully configurable via the `clamd.conf` file [6]. `clamd` recognizes the following commands:

- **PING**
  Check the daemon's state (should reply with "PONG").
- **VERSION**
  Print program and database versions.
- **RELOAD**
  Reload the databases.

[6] man 5 clamd.conf

- **SHUTDOWN**
  Perform a clean exit.

- **SCAN file/directory**
  Scan file or directory (recursively) with archive support enabled (a full path is required).

- **RAWSCAN file/directory**
  Scan file or directory (recursively) with archive and special file support disabled (a full path is required).

- **CONTSCAN file/directory**
  Scan file or directory (recursively) with archive support enabled and don't stop the scanning when a virus is found.

- **MULTISCAN file/directory**
  Scan file in a standard way or scan directory (recursively) using multiple threads (to make the scanning faster on SMP machines).

- **STREAM**
  Scan stream: `clamd` will return a new port number you should connect to and send data to scan.

- **SESSION, END**
  Start/end a `clamd` session - you can do multiple commands per TCP session (WARNING: due to the `clamd` implementation the **RELOAD** command will break the session).

It's recommended to prefix clamd commands with the letter `n` (eg. `nSCAN`) to indicate that the command will be delimited by a newline character and that clamd should continue reading command data until a newline is read. The newline delimiter assures that the complete command and its entire argument will be processed as a single command.

Clamd can handle the following signals:

- **SIGTERM** - perform a clean exit

- **SIGHUP** - reopen the log file

- **SIGUSR2** - reload the database

Clamd should not be started in the background using the shell operator `&` or external tools. Instead, you should run and wait for clamd to load the database and daemonize itself. After that, clamd is instantly ready to accept connections and perform file scanning.

5.2 Clamdscan

`clamdscan` is a simple `clamd` client. In many cases you can use it as a `clamscan` replacement however you must remember that:

- it only depends on `clamd`
- although it accepts the same command line options as `clamscan` most of them are ignored because they must be enabled directly in `clamd`, i.e. `clamd.conf`
- scanned files must be accessible for `clamd`
- it can't use external unpackers

5.3 Clamuko

Clamuko is a special thread in `clamd` that performs on-access scanning under Linux and FreeBSD and shares internal virus database with the daemon. **You must follow some important rules when using it:**

- Always stop the daemon cleanly - using the SHUTDOWN command or the SIGTERM signal. In other case you can lose access to protected files until the system is restarted.
- Never protect the directory your mail-scanner software uses for attachment unpacking. Access to all infected files will be automatically blocked and the scanner (including `clamd`!) will not be able to detect any viruses. In the result **all infected mails may be delivered.**

For example, to protect the whole system add the following lines to `clamd.conf`:

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

You can also use clamuko to protect files on Samba/Netatalk but a far more better and safe idea is to use the **samba-vscan** module. NFS is not supported because Dazuko doesn't intercept NFS access calls.

5.4 Output format

5.4.1 clamscan

`clamscan` writes all regular program messages to **stdout** and errors/warnings to **stderr**. You can use the option `--stdout` to redirect all program messages to **stdout**. Warnings and error messages from `libclamav` are always printed to **stderr**. A typical output from `clamscan` looks like this:

```
/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
```

When a virus is found its name is printed between the `filename:` and `FOUND` strings. In case of archives the scanner depends on libclamav and only prints the first virus found within an archive:

```
zolw@localhost:/tmp$ clamscan malware.zip
malware.zip: Worm.Mydoom.U FOUND
```

TIP: *You can force clamscan to list all infected files in an archive using –no-archive (this option disables transparent decompressors built into libclamav) and enabling external decompressors: –unzip –unrar....*

```
zolw@localhost:/tmp$ clamscan --no-archive --unzip malware.zip
Archive:  /tmp/malware.zip
  inflating: test1.exe
  inflating: test2.exe
  inflating: test3.exe
/tmp/clamav-77e7bfdbb2d3872b/test1.exe: Worm.Mydoom.U FOUND
/tmp/clamav-77e7bfdbb2d3872b/test2.exe: Trojan.Taskkill.A FOUND
/tmp/clamav-77e7bfdbb2d3872b/test3.exe: Worm.Nyxem.D FOUND
/tmp/malware.zip: Infected.Archive FOUND
```

5.4.2 clamd

The output format of `clamd` is very similar to `clamscan`.

```
zolw@localhost:~$ telnet localhost 3310
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
SCAN /home/zolw/test
/home/zolw/test/clam.exe: ClamAV-Test-File FOUND
Connection closed by foreign host.
```

In the **SCAN** mode it closes the connection when the first virus is found.

```
SCAN /home/zolw/test/clam.zip
/home/zolw/test/clam.zip: ClamAV-Test-File FOUND
```

CONTSCAN and **MULTISCAN** don't stop scanning in case a virus is found.
Error messages are printed in the following format:

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

6 LibClamAV

Libclamav provides an easy and effective way to add a virus protection into your software. The library is thread-safe and transparently recognizes and scans within archives, mail files, MS Office document files, executables and other special formats.

6.1 Licence

Libclamav is licensed under the GNU GPL v2 licence. This means you are **not allowed** to link commercial, close-source applications against it[7]. All software using libclamav must be GPL compliant.

6.2 Supported formats

6.2.1 Executables

The library has a built-in support for 32/64-bit Portable Executable files and 32-bit ELF files. Additionally, it can handle PE files compressed or obfuscated with the following tools:

[7] You can still use clamd or clamscan instead

- Aspack (2.12)
- UPX (all versions)
- FSG (1.3, 1.31, 1.33, 2.0)
- Petite (2.x)
- PeSpin (1.1)
- NsPack
- wwpack32 (1.20)
- MEW
- Upack
- Y0da Cryptor (1.3)

6.2.2 Mail files

Libclamav can handle almost every mail file format including TNEF (winmail.dat) attachments.

6.2.3 Archives and compressed files

The following archive and compression formats are supported by internal handlers:

- Zip (+ SFX)
- RAR (+ SFX)
- Tar
- Gzip
- Bzip2
- MS OLE2
- MS Cabinet Files (+ SFX)
- MS CHM (Compiled HTML)
- MS SZDD compression format

- BinHex
- SIS (SymbianOS packages)
- AutoIt

6.2.4 Documents

The most popular file formats are supported:

- MS Office and MacOffice files
- RTF
- PDF
- HTML

6.2.5 Others

Libclamav can handle various obfuscators, encoders, files vulnerable to security risks such as:

- JPEG (exploit detection)
- RIFF (exploit detection)
- uuencode
- ScrEnc obfuscation
- CryptFF

6.3 API

6.3.1 Header file

Every program using libclamav must include the header file `clamav.h`:

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

6.3.2 Database loading

The following set of functions provides an interface for loading the virus database:

```
const char *cl_retdbdir(void);

int cl_load(const char *path, struct cl_engine **engine,
    unsigned int *signo, unsigned int options);
```

`cl_retdbdir` returns the default (hardcoded) path to the directory with ClamAV databases. `cl_load` loads a single database file or all databases from a directory (if `path` points to a directory). The second argument is used for passing in the engine structure which should be previously initialized with NULL. A number of loaded signatures will be **added** to `signo` [8]. The last argument can pass the following flags:

- **CL_DB_STDOPT**
  This is an alias for a recommended set of scan options.
- **CL_DB_PHISHING**
  Load phishing signatures.
- **CL_DB_PHISHING_URLS**
  Initialize the phishing detection module and load .wdb and .pdb files.
- **CL_DB_PUA**
  Load signatures for Potentially Unwanted Applications.
- **CL_DB_CVDNOTMP**
  Load CVD files directly without unpacking them into a temporary directory.

`cl_load` returns 0 (`CL_SUCCESS`) on success and a negative value on failure.

```
    ...
    struct cl_engine *engine = NULL;
    unsigned int sigs = 0;
    int ret;

ret = cl_load(cl_retdbdir(), &engine, &sigs, CL_DB_STDOPT);
```

[8] Remember to initialize the virus counter variable with 0.

6.3.3 Error handling

Use `cl_strerror` to convert error codes into human readable messages. The function returns a statically allocated string:

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

6.3.4 Engine structure

When all required databases are loaded you should prepare the detection engine by calling `cl_build`. In the case of failure you should free the memory occupied by the engine with `cl_free`:

```
int cl_build(struct cl_engine *engine);
void cl_free(struct cl_engine *engine);
```

In our example:

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

6.4 Database reloading

The most important thing is to keep the internal instance of the database up to date. You can watch database changes with the `cl_stat` family of functions.

```
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);
```

Initialization:

```
    ...
    struct cl_stat dbstat;

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

To check for a change you just need to call `cl_statchkdir` and check its return value (0 - no change, 1 - some change occured):

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

Remember to reset the `cl_stat` structure after reload.

6.4.1 Data scan functions

It's possible to scan a file or descriptor using:

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

int cl_scandesc(int desc, const char **virname, unsigned
long int *scanned, const struct cl_engine *engine, const
struct cl_limits *limits, unsigned int options);
```

Both functions will store a virus name under the pointer `virname`, the virus name is part of the engine structure and must not be released directly. If the third argument (`scanned`) is not NULL, the functions will increase its value with the size of scanned data (in `CL_COUNT_PRECISION` units). Both functions have support for archive limits in order to protect against Denial of Service attacks.

```
struct cl_limits {
    unsigned long int maxscansize;  /* during the scanning of archives this
                                     * size will never be exceeded
                                     */
    unsigned long int maxfilesize;  /* compressed files will only be
```

```
                                    * decompressed and scanned up to this size
                                    */
    unsigned int maxreclevel;      /* maximum recursion level for archives */
    unsigned int maxfiles;         /* maximum number of files to be scanned
                                    * within a single archive
                                    */
    unsigned short archivememlim;  /* limit memory usage for some unpackers */
};
```

The last argument (`options`) configures the scan engine and supports the following flags (that can be combined using bit operators):

- **CL_SCAN_STDOPT**
  This is an alias for a recommended set of scan options. You should use it to make your software ready for new features in the future versions of libclamav.
- **CL_SCAN_RAW**
  Use it alone if you want to disable support for special files.
- **CL_SCAN_ARCHIVE**
  This flag enables transparent scanning of various archive formats.
- **CL_SCAN_BLOCKENCRYPTED**
  With this flag the library will mark encrypted archives as viruses (Encrypted.Zip, Encrypted.RAR).
- **CL_SCAN_MAIL**
  Enable support for mail files.
- **CL_SCAN_MAILURL**
  The mail scanner will download and scan URLs listed in a mail body. This flag should not be used on loaded servers. Due to potential problems please do not enable it by default but make it optional.
- **CL_SCAN_OLE2**
  Enables support for OLE2 containers (used by MS Office and .msi files).
- **CL_SCAN_PDF**
  Enables scanning within PDF files.
- **CL_SCAN_PE**
  This flag enables deep scanning of Portable Executable files and allows libclamav to unpack executables compressed with run-time unpackers.

- **CL_SCAN_ELF**
  Enable support for ELF files.
- **CL_SCAN_BLOCKBROKEN**
  libclamav will try to detect broken executables and mark them as Broken.Executable.
- **CL_SCAN_HTML**
  This flag enables HTML normalisation (including ScrEnc decryption).
- **CL_SCAN_ALGORITHMIC**
  Enable algorithmic detection of viruses.
- **CL_SCAN_PHISHING_BLOCKSSL**
  Phishing module: always block SSL mismatches in URLs.
- **CL_SCAN_PHISHING_BLOCKCLOAK**
  Phishing module: always block cloaked URLs.

All functions return 0 (`CL_CLEAN`) when the file seems clean, `CL_VIRUS` when a virus is detected and another value on failure.

```
    ...
    struct cl_limits limits;
    const char *virname;

memset(&limits, 0, sizeof(struct cl_limits));
limits.maxfiles = 10000;
limits.maxscansize = 100 * 1048576; /* 100 MB */
limits.maxfilesize = 10 * 1048576; /* 10 MB */
limits.maxreclevel = 16;

if((ret = cl_scanfile("/tmp/test.exe", &virname, NULL, engine,
&limits, CL_STDOPT)) == CL_VIRUS) {
    printf("Virus detected: %s\n", virname);
} else {
    printf("No virus detected.\n");
    if(ret != CL_CLEAN)
        printf("Error: %s\n", cl_strerror(ret));
}
```

6.4.2 Memory

Because the engine structure occupies a few megabytes of system memory, you should release it with `cl_free` if you no longer need to scan files.

6.4.3 Forking daemons

If you're using libclamav with a forking daemon you should call `srand()` inside a forked child before making any calls to the libclamav functions. This will avoid possible collisions with temporary filenames created by other processes of the daemon. This procedure is not required for multi-threaded daemons.

6.4.4 clamav-config

Use `clamav-config` to check compilation information for libclamav.

```
zolw@localhost:~$ clamav-config --libs
-L/usr/local/lib -lz -lbz2 -lgmp -lpthread
zolw@localhost:~$ clamav-config --cflags
-I/usr/local/include -g -O2
```

6.4.5 Example

You will find an example scanner application in the clamav sources (/example). Don't forget that all programs based on libclamav must be linked against it:

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

6.5 CVD format

CVD (ClamAV Virus Database) is a digitally signed tarball containing one or more databases. The header is a 512-bytes long string with colon separated fields:

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

`sigtool --info` displays detailed information on CVD files:

```
zolw@localhost:/usr/local/share/clamav$ sigtool -i daily.cvd
File: daily.cvd
Build time: 10 Mar 2008 10:45 +0000
```

```
Version: 6191
Signatures: 59084
Functionality level: 26
Builder: ccordes
MD5: 6e6e29dae36b4b7315932c921e568330
Digital signature: zz9irc9irupR3z7yX6J+OR6XdFPUat4HIM9ERn3kAcOWpcMFxq
Fs4toG5WJsHda0Jj92IUusZ7wAgYjpai1Nr+jFfXHsJxv0dBkS5/XWMntj0T1ctNgqmiF
+RLU6V0VeTl4Oej3Aya0cVpd9K4XXevEO2eTTvzWNCAq0ZzWNdjc
Verification OK.
```

6.6 Contributors

The following people contributed to our project in some way (providing patches, bug reports, technical support, documentation, good ideas...):

- Ian Abbott <abbotti*mev.co.uk>
- Clint Adams <schizo*debian.org>
- Sergey Y. Afonin <asy*kraft-s.ru>
- Robert Allerstorfer <roal*anet.at>
- Claudio Alonso <cfalonso*yahoo.com>
- Kevin Amorin <kamorin*ccs.neu.edu>
- Kamil Andrusz <wizz*mniam.net>
- Tayfun Asker <tasker*metu.edu.tr>
- Jean-Edouard Babin <Jeb*jeb.com.fr>
- Marc Baudoin <babafou*babafou.eu.org>
- Scott Beck <sbeck*gossamer-threads.com>
- Rolf Eike Beer <eike*mail.math.uni-mannheim.de>
- Rene Bellora <rbellora*tecnoaccion.com.ar>
- Carlo Marcelo Arenas Belon <carenas*sajinet.com.pe>
- Joseph Benden <joe*thrallingpenguin.com>

- Hilko Bengen <bengen*vdst-ka.inka.de>
- Hank Beatty <hbeatty*starband.net>
- Alexandre Biancalana <ale*seudns.net>
- Patrick Bihan-Faou <patrick*mindstep.com>
- Martin Blapp <mb*imp.ch>
- Dale Blount <dale*velocity.net>
- Serge van den Boom <svdb*stack.nl>
- Oliver Brandmueller <ob*e-Gitt.NET>
- Boguslaw Brandys <brandys*o2.pl>
- Igor Brezac <igor*ipass.net>
- Mike Brudenell <pmb1*york.ac.uk>
- Brian Bruns <bruns*2mbit.com>
- Len Budney <lbudney*pobox.com>
- Matt Butt <mattb*cre8tiv.com>
- Christopher X. Candreva <chris*westnet.com>
- Eric I. Lopez Carreon <elopezc*technitrade.com>
- Ales Casar <casar*uni-mb.si>
- Jonathan Chen <jon+clamav*spock.org>
- Andrey Cherezov <andrey*cherezov.koenig.su>
- Alex Cherney <alex*cher.id.au>
- Tom G. Christensen <tgc*statsbiblioteket.dk>
- Nicholas Chua <nicholas*ncmbox.net>
- Chris Conn <cconn*abacom.com>
- Christoph Cordes <ib*precompiled.de>
- Ole Craig <olc*cs.umass.edu>

- Eugene Crosser <crosser*rol.ru>
- Calin A. Culianu <calin*ajvar.org>
- Damien Curtain <damien*pagefault.org>
- Krisztian Czako <slapic*linux.co.hu>
- Diego d'Ambra <da*softcom.dk>
- Michael Dankov <misha*btrc.ru>
- Yuri Dario <mc6530*mclink.it>
- David <djgardner*users.sourceforge.net>
- Maxim Dounin <mdounin*rambler-co.ru>
- Alejandro Dubrovsky <s328940*student.uq.edu.au>
- James P. Dugal <jpd*louisiana.edu>
- Magnus Ekdahl <magnus*debian.org>
- Mehmet Ekiz <ekizm*tbmm.gov.tr>
- Jens Elkner <elkner*linofee.org>
- Fred van Engen <fred*wooha.org>
- Jason Englander <jason*englanders.cc>
- Oden Eriksson <oeriksson*mandrakesoft.com>
- Daniel Fahlgren <fahlgren*ardendo.se>
- Andy Fiddaman <af*jeamland.org>
- Edison Figueira Junior <edison*brc.com.br>
- David Ford <david+cert*blue-labs.org>
- Martin Forssen <maf*appgate.com>
- Brian J. France <list*firehawksystems.com>
- Free Oscar <freeoscar*wp.pl>
- Martin Fuxa <yeti*email.cz>

- Piotr Gackiewicz <gacek*intertele.pl>
- Jeremy Garcia <jeremy*linuxquestions.org>
- Dean Gaudet <dean-clamav*arctic.org>
- Michel Gaudet <Michel.Gaudet*ehess.fr>
- Philippe Gay <ph.gay*free.fr>
- Nick Gazaloff <nick*sbin.org>
- Geoff Gibbs <ggibbs*hgmp.mrc.ac.uk>
- Luca 'NERvOus' Gibelli <nervous*nervous.it>
- Scott Gifford <sgifford*suspectclass.com>
- Wieslaw Glod <wkg*x2.pl>
- Stephen Gran <steve*lobefin.net>
- Koryn Grant <koryn*endace.com>
- Matthew A. Grant <grantma*anathoth.gen.nz>
- Christophe Grenier <grenier*cgsecurity.org>
- Marek Gutkowski <hobbit*core.segfault.pl>
- Jason Haar <Jason.Haar*trimble.co.nz>
- Hrvoje Habjanic <hrvoje.habjanic*zg.hinet.hr>
- Michal Hajduczenia <michalis*mat.uni.torun.pl>
- Jean-Christophe Heger <jcheger*acytec.com>
- Martin Heinz <Martin*hemag.ch>
- Kevin Heneveld" <kevin*northstar.k12.ak.us>
- Anders Herbjornsen <andersh*gar.no>
- Paul Hoadley <paulh*logixsquad.net>
- Robert Hogan <robert*roberthogan.net>
- Przemyslaw Holowczyc <doozer*skc.com.pl>

- Thomas W. Holt Jr. <twh*cohesive.net>
- James F. Hranicky <jfh*cise.ufl.edu>
- Douglas J Hunley <doug*hunley.homeip.net>
- Kurt Huwig <kurt*iku-netz.de>
- Andy Igoshin <ai*vsu.ru>
- Michal Jaegermann <michal*harddata.com>
- Christophe Jaillet <christophe.jaillet*wanadoo.fr>
- Jay <sysop-clamav*coronastreet.net>
- Stephane Jeannenot <stephane.jeannenot*wanadoo.fr>
- Per Jessen <per*computer.org>
- Dave Jones <dave*kalkbay.co.za>
- Jesper Juhl <juhl*dif.dk>
- Kamil Kaczkowski <kamil*kamil.eisp.pl>
- Alex Kah <alex*narfonix.com>
- Stefan Kaltenbrunner <stefan*kaltenbrunner.cc>
- Lloyd Kamara <l.kamara*imperial.ac.uk>
- Stefan Kanthak <stefan.kanthak*fujitsu-siemens.com>
- Kazuhiko <kazuhiko*fdiary.net>
- Jeremy Kitchen <kitchen*scriptkitchen.com>
- Tomasz Klim <tomek*euroneto.pl>
- Robbert Kouprie <robbert*exx.nl>
- Martin Kraft <martin.kraft*fal.de>
- Petr Kristof <Kristof.P*fce.vutbr.cz>
- Henk Kuipers <henk*opensourcesolutions.nl>
- Nigel Kukard <nkukard*lbsd.net>

- Eugene Kurmanin <smfs*users.sourceforge.net>
- Dr Andrzej Kurpiel <akurpiel*mat.uni.torun.pl>
- Mark Kushinsky <mark*mdspc.com>
- Mike Lambert <lambert*jeol.com>
- Thomas Lamy <Thomas.Lamy*in-online.net>
- Stephane Leclerc <sleclerc*aliastec.net>
- Marty Lee <marty*maui.co.uk>
- Dennis Leeuw <dleeuw*made-it.com>
- Martin Lesser <admin-debian*bettercom.de>
- Peter N Lewis <peter*stairways.com.au>
- Matt Leyda <mfleyda*e-one.com>
- James Lick <jlick*drivel.com>
- Jerome Limozin <jerome*limozin.net>
- Mike Loewen <mloewen*sturgeon.cac.psu.edu>
- Roger Lucas <roger*planbit.co.uk>
- David Luyer <david_luyer*pacific.net.au>
- Richard Lyons <frob-clamav*webcentral.com.au>
- David S. Madole <david*madole.net>
- Thomas Madsen <tm*softcom.dk>
- Bill Maidment <bill*maidment.com.au>
- Joe Maimon <jmaimon*ttec.com>
- David Majorel <dm*lagoon.nc>
- Andrey V. Malyshev <amal*krasn.ru>
- Fukuda Manabu <fukuda*cri-mw.co.jp>
- Stefan Martig <sm*officeco.ch>

- Alexander Marx <mad-ml*madness.at>
- Andreas Marx (http://www.av-test.org/)
- Chris Masters <cmasters*insl.co.uk>
- Fletcher Mattox <fletcher*cs.utexas.edu>
- Serhiy V. Matveyev <matveyev*uatele.com>
- Reinhard Max <max*suse.de>
- Brian May <bam*debian.org>
- Ken McKittrick <klmac*usadatanet.com>
- Chris van Meerendonk <cvm*castel.nl>
- Andrey J. Melnikoff <temnota*kmv.ru>
- Damian Menscher <menscher*uiuc.edu>
- Denis De Messemacker <ddm*clamav.net>
- Jasper Metselaar <jasper*formmailer.net>
- Arkadiusz Miskiewicz <misiek*pld-linux.org>
- Ted Mittelstaedt <tedm*toybox.placo.com>
- Mark Mielke <mark*mark.mielke.cc>
- John Miller <contact*glideslopesoftware.co.uk>
- Jo Mills <Jonathan.Mills*frequentis.com>
- Dustin Mollo <dustin.mollo*sonoma.edu>
- Remi Mommsen <remigius.mommsen*cern.ch>
- Doug Monroe <doug*planetconnect.com>
- Alex S Moore <asmoore*edge.net>
- Tim Morgan <tim*sentinelchicken.org>
- Dirk Mueller <mueller*kde.org>
- Flinn Mueller<flinn*activeintra.net>

- Hendrik Muhs <Hendrik.Muhs*student.uni-magdeburg.de>
- Simon Munton <simon*munton.demon.co.uk>
- Farit Nabiullin (http://program.farit.ru/)
- Nemosoft Unv. <nemosoft*smcc.demon.nl>
- Wojciech Noworyta <wnow*konarski.edu.pl>
- Jorgen Norgaard <jnp*anneli.dk>
- Fajar A. Nugraha <fajar*telkom.co.id>
- Joe Oaks <joe.oaks*hp.com>
- Washington Odhiambo <wash*wananchi.com>
- Masaki Ogawa <proc*mac.com>
- John Ogness <jogness*antivir.de>
- Phil Oleson <oz*nixil.net>
- Jan Ondrej <ondrejj*salstar.sk>
- Martijn van Oosterhout <kleptog*svana.org>
- OpenAntiVirus Team (http://www.OpenAntiVirus.org/)
- Tomasz Papszun <tomek*lodz.tpsa.pl>
- Eric Parsonage <eric*eparsonage.com>
- Oliver Paukstadt <pstadt*stud.fh-heilbronn.de>
- Christian Pelissier <Christian.Pelissier*onera.fr>
- Rudolph Pereira <rudolph*usyd.edu.au>
- Dennis Peterson <dennispe*inetnw.com>
- Ed Phillips <ed*UDel.Edu>
- Andreas Piesk <Andreas.Piesk*heise.de>
- Mark Pizzolato <clamav-devel*subscriptions.pizzolato.net>
- Dean Plant <dean.plant*roke.co.uk>

- Alex Pleiner <pleiner*zeitform.de>
- Ant La Porte <ant*dvere.net>
- Jef Poskanzer <jef*acme.com>
- Christophe Poujol <Christophe.Poujol*atosorigin.com>
- Sergei Pronin <sp*finndesign.fi>
- Thomas Quinot <thomas*cuivre.fr.eu.org>
- Ed Ravin <eravin*panix.com>
- Robert Rebbun <robert*desertsurf.com>
- Brian A. Reiter <breiter*wolfereiter.com>
- Didi Rieder <adrieder*sbox.tugraz.at>
- Pavel V. Rochnyack <rpv*fsf.tsu.ru>
- Rupert Roesler-Schmidt <r.roesler-schmidt*uplink.at>
- David Sanchez <dsanchez*veloxia.com>
- David Santinoli <david*santinoli.com>
- Vijay Sarvepalli <vssarvep*office.uncg.edu>
- Martin Schitter
- Theo Schlossnagle <jesus*omniti.com>
- Enrico Scholz <enrico.scholz*informatik.tu-chemnitz.de>
- Karina Schwarz <k.schwarz*uplink.at>
- Scsi <scsi*softland.ru>
- Dr Matthew J Seaman <m.seaman*infracaninophile.co.uk>
- Hector M. Rulot Segovia <Hector.Rulot*uv.es>
- Omer Faruk Sen <ofsen*enderunix.org>
- Sergey <a_s_y*sama.ru>
- Tuomas Silen <tuomas.silen*nodeta.fi>

- David F. Skoll <dfs*roaringpenguin.com>
- Al Smith <ajs+clamav*aeschi.ch.eu.org>
- Sergey Smitienko <hunter*comsys.com.ua>
- Solar Designer <solar*openwall.com>
- Joerg Sonnenberger <joerg*britannica.bec.de>
- Michal 'GiM' Spadlinski (http://gim.org.pl/)
- Kevin Spicer <kevin*kevinspicer.co.uk>
- GertJan Spoelman <cav*gjs.cc>
- Ole Stanstrup <ole*stanstrup.dk>
- Adam Stein <adam*scan.mc.xerox.com>
- Steve <steveb*webtribe.net>
- Richard Stevenson <richard*endace.com>
- Sven Strickroth <sstrickroth*gym-oha.de>
- Matt Sullivan <matt*sullivan.gen.nz>
- Dr Zbigniew Szewczak <zssz*mat.uni.torun.pl>
- Joe Talbott <josepht*cstone.net>
- Gernot Tenchio <g.tenchio*telco-tech.de>
- Masahiro Teramoto <markun*onohara.to>
- Daniel Theodoro <dtheodoro*ig.com.br>
- Ryan Thompson <clamav*sasknow.com>
- Gianluigi Tiesi <sherpya*netfarm.it>
- Yar Tikhiy <yar*comp.chem.msu.su>
- Andrew Toller <atoller*connectfree.co.uk>
- Michael L. Torrie <torriem*chem.byu.edu>
- Trashware <trashware*gmx.net>

- Matthew Trent <mtrent*localaccess.com>
- Reini Urban <rurban*x-ray.at>
- Daniel Mario Vega <dv5a*dc.uba.ar>
- Denis Vlasenko <vda*ilport.com.ua>
- Laurent Wacrenier <lwa*teaser.fr>
- Charlie Watts <cewatts*brainstorminternet.net>
- Florian Weimer <fw*deneb.enyo.de>
- Paul Welsh <paul*welshfamily.com>
- Nicklaus Wicker <n.wicker*cnk-networks.de>
- David Woakes <david*mitredata.co.uk>
- Troy Wollenslegel <troy*intranet.org>
- ST Wong <st-wong*cuhk.edu.hk>
- Dale Woolridge <dwoolridge*drh.net>
- David Wu <dyw*iohk.com>
- Takumi Yamane <yamtak*b-session.com>
- Youza Youzovic <youza*post.cz>
- Anton Yuzhaninov <citrin*rambler-co.ru>
- Leonid Zeitlin <lz*europe.com>
- ZMan Z. <x86zman*go-a-way.dyndns.org>
- Andoni Zubimendi <andoni*lpsat.net>

6.7 Donors

We've received financial support from: (in alphabetical order)

- ActiveIntra.net Inc. (http://www.activeintra.net/)
- Advance Healthcare Group (http://www.ahgl.com.au/)
- Allied Quotes (http://www.AlliedQuotes.com/)
- American Computer & Electronic Services Corp. (http://www.acesnw.com/)
- Amnesty International, Swiss Section (http://www.amnesty.ch/)
- Steve Anderson
- Anonymous donor from Colorado, US
- Arudius (http://arudius.sourceforge.net/)
- Peter Ashman
- Atlas College (http://www.atlascollege.nl/)
- Australian Payday Cash Loans (http://www.cashdoctors.com.au/)
- AWD Online (http://www.awdonline.com/)
- BackupAssist Backup Software (http://www.backupassist.com/)
- Dave Baker
- Bear and Bear Consulting, Inc. (http://www.bear-consulting.com/)
- Aaron Begley
- Craig H. Block
- Norman E. Brake, Jr.
- Josh Burstyn
- By Design (http://www.by-design.net/)
- Canadian Web Hosting (http://www.canadianwebhosting.com/)
- cedarcreeksoftware.com (http://www.cedarcreeksoftware.com/)
- Ricardo Cerqueira

- Thanos Chatziathanassiou
- Cheahch from Singapore
- Conexim Australia - business web hosting (http://www.conexim.com.au)
- Alan Cook
- Joe Cooper
- CustomLogic LLC (http://www.customlogic.com/)
- Ron DeFulio
- Digirati (http://oss.digirati.com.br/)
- Steve Donegan (http://www.donegan.org/)
- Dynamic Network Services, Inc (http://www.dyndns.org/)
- EAS Enterprises LLC
- eCoupons.com (http://www.ecoupons.com/)
- Electric Embers (http://electricembers.net)
- John T. Ellis
- Epublica
- Bernhard Erdmann
- David Eriksson (http://www.2good.nu/)
- Philip Ershler
- Explido Software USA Inc. (http://www.explido.us/)
- David Farrick
- Jim Feldman
- Petr Ferschmann (http://petr.ferschmann.cz/)
- Andries Filmer (http://www.netexpo.nl/)
- The Free Shopping Cart people (http://www.precisionweb.net/)
- Paul Freeman

- Jack Fung
- Stephen Gageby
- Paolo Galeazzi
- GANDI (http://www.gandi.net/)
- Jeremy Garcia (http://www.linuxquestions.org/)
- GBC Internet Service Center GmbH (http://www.gbc.net/)
- GCS Tech (http://www.gcstech.net/)
- GHRS (http://www.ghrshotels.com/)
- Lyle Giese
- Todd Goodman
- Bill Gradwohl (http://www.ycc.com/)
- Grain-of-Salt Consulting
- Terje Gravvold
- Hart Computer (http://www.hart.co.jp/)
- Pen Helm
- Hosting Metro LLC (http://www.hostingmetro.com/)
- IDEAL Software GmbH (http://www.IdealSoftware.com/)
- Industry Standard Computers (http://www.ISCnetwork.com/)
- Interact2Day (http://www.interact2day.com/)
- Invisik Corporation (http://www.invisik.com/)
- itXcel Internet - Domain Registration (http://www.itxcel.com)
- Craig Jackson
- Stuart Jones
- Jason Judge
- Keith (http://www.textpad.com/)

- Ewald Kicker (http://www.very-clever.com/)
- Brad Koehn
- Christina Kuratli (http://www.virusprotect.ch/)
- Logic Partners Inc. (http://www.logicpartners.com/)
- Mark Lotspaih (http://www.lotcom.org/)
- Michel Machado (http://oss.digirati.com.br/)
- Olivier Marechal
- Matthew McKenzie
- Durval Menezes (http://www.durval.com.br/)
- Micro Logic Systems (http://www.mls.nc/)
- Midcoast Internet Solutions
- Mimecast (http://www.mimecast.com/)
- Kazuhiro Miyaji
- Bozidar Mladenovic
- Paul Morgan
- Tomas Morkus
- The Names Database (http://static.namesdatabase.com)
- Names Directory (http://www.namesdir.com/)
- Michael Nolan (http://www.michaelnolan.co.uk/)
- Jorgen Norgaard
- Numedeon, Inc. creators of Whyville (http://www.whyville.net/)
- Oneworkspace.com (http://www.oneworkspace.com/)
- Online Literature (http://www.couol.com/)
- Origin Solutions (http://www.originsolutions.com.au/)
- outermedia GmbH (http://www.outermedia.de/)

- Kevin Pang (http://www.freebsdblog.org/)
- Alexander Panzhin
- Passageway Communications (http://www.passageway.com)
- Dan Pelleg (http://www.libagent.org/)
- Thodoris Pitikaris
- Paul Rantin
- Thomas J. Raef (http://www.ebasedsecurity.com)
- Luke Reeves (http://www.neuro-tech.net/)
- RHX (http://www.rhx.it/)
- Stefano Rizzetto
- Roaring Penguin Software Inc. (http://www.roaringpenguin.com/)
- Luke Rosenthal
- Jenny Sfstrm (http://PokerListings.com)
- School of Engineering, University of Pennsylvania (http://www.seas.upenn.edu/)
- Tim Scoff
- Seattle Server (http://www.seattleserver.com/)
- Software Workshop Inc (http://www.softwareworkshop.com/)
- Solutions In A Box (http://www.siab.com.au/)
- Stephane Rault
- SearchMain (http://www.searchmain.com/)
- Olivier Silber
- Fernando Augusto Medeiros Silva (http://www.linuxplace.com.br/)
- Sollentuna Fria Gymnasium, Sweden (http://www.sfg.se/)
- StarBand (http://www.starband.com/)

- Stroke of Color, Inc.
- Synchro Sistemas de Informacao (http://synchro.com.br/)
- Sahil Tandon
- The Spamex Disposable Email Address Service (http://www.spamex.com)
- Brad Tarver
- TGT Tampermeier & Grill Steuerberatungs- und Wirtschaftstreuhand OEG (http://www.tgt.at/)
- Per Reedtz Thomsen
- William Tisdale
- Up Time Technology (http://www.uptimetech.com/)
- Ulfi
- Jeremy Vanderburg (http://www.jeremytech.com/)
- Web.arbyte - Online-Marketing (http://www.webarbyte.de/)
- Webzone Srl (http://www.webzone.it/)
- Markus Welsch (http://www.linux-corner.net/)
- Julia White (http://www.convert-tools.com/)
- Nicklaus Wicker
- David Williams (http://kayakero.net/)
- Glenn R Williams
- Kelly Williams
- XRoads Networks (http://xroadsnetworks.com/)
- Zimbra open-source collaboration suite (http://www.zimbra.com/)

6.8 Graphics

The ClamAV logo was created by Mia Kalenius and Sergei Pronin from Finndesign (http://www.finndesign.fi/).

6.9 OpenAntiVirus

Our database includes the virus database (about 7000 signatures) from OpenAntiVirus (`http://OpenAntiVirus.org`).

7 Core Team

- aCaB `<acab*clamav.net>`, Italy
  Role: virus database maintainer, coder
- Mike Cathey `<mike*clamav.net>`, USA
  Role: co-sysadmin
- Christoph Cordes `<ccordes*clamav.net>`, Germany
  Role: virus database maintainer
- Diego d'Ambra `<diego*clamav.net>`, Denmark
  Role: virus database maintainer
- Luca Gibelli `<luca*clamav.net>`, Italy
  Role: sysadmin, mirror coordinator
- Nigel Horne `<njh*clamav.net>`, United Kingdom
  Role: coder
- Arnaud Jacques `<arnaud*clamav.net>`, France
  Role: virus database maintainer
- Tomasz Kojm `<tkojm*clamav.net>`, Poland
  Role: project leader, coder
- Tomasz Papszun `<tomek*clamav.net>`, Poland
  Role: various help
- Sven Strickroth `<sven*clamav.net>`, Germany
  Role: virus database maintainer, virus submission management
- Edwin Torok `<edwin*clamav.net>`, Romania
  Role: coder
- Trog `<trog*clamav.net>`, United Kingdom
  Role: coder