

## Implementing OpenSolaris' vscan service with ClamAV on ZFS filesystems.

John Weekley

I've currently got the ClamAV anti-virus toolkit scanning my emails, which is fine; but it only addresses one method viruses can sneak into and more importantly, out of my OpenSolaris systems. Recently, the vscan service was implemented in OpenSolaris Build 78. I looked around and found that it met my needs with some additional software bits.

To be clear, VSCAN can't determine whether a file is infected with a virus or not. vscan relies on third-party virus scanners that support the ICAP protocol (IETF RFC 3507) to do the actual content scanning. I have ClamAV already running, but it doesn't natively support the ICAP protocol. It needs a little help in the form of an ICAP server. I decided to use c\_icap to bridge the gap between the vscan service and ClamAV.

The pieces you'll need to do this are:

OpenSolaris build 78 or greater.

ClamAV <http://www.clamav.org>

c\_icap <http://c-icap.sourceforge.net/>

eicar.com [http://www.eicar.org/anti\\_virus\\_test\\_file.htm](http://www.eicar.org/anti_virus_test_file.htm)

Note: this is a harmless file designed to test out anti-virus products. It's incapable of infecting anything and is safe to download for testing purposes.

I chose to use Blastwave's (<http://www.blastwave.org>) Clam antivirus packages to eliminate some hassles with building ClamAV.

Once you have ClamAV working and you've downloaded the c\_icap client, you're ready to get to work.

Building c\_icap is fairly straightforward, if you use the bundled gcc. And don't try to use the "--enable-ipv6" which produced a broken icap-client that was unable to connect to the server.

I usually create a build script to do this:

```
make distclean
export LDFLAGS="-L/opt/csw/lib -R/opt/csw/lib -lclamav"
export CC=gcc
export CXX=g++
```

```
./configure \
--prefix=/opt/icap \
--with-clamav=/opt/csw \
--with-perl=/bin/perl \
--with-zlib=/usr/lib \
--enable-large-files
```

```
make
make install
and source it in:
. ./BUILD
```

I found that the “-with-clamav” option to configure didn't include the runpath for libclamav. To avoid some some troubleshooting later, explicitly set it in the LD\_FLAGS environment variable.

Once c\_icap's been built and installed, edit /opt/icap/etc/c-icap.conf, setting your logging preferences and network access control list(s). Mine are fairly simple for this HOWTO. I use syslog for logging, and my acl allows anyone on the 192.168.1.0/24 network access to the ICAP server.

```
/opt/icap/etc/c-icap.conf (changes only).
```

```
User nobody

##Specify wich logger to use.....
Logger sys_logger

## An example of acl lists for default_acl controller.
acl localnet src 192.168.1.0/255.255.255.0

##An example to specify access to server
icap_access allow localnet
```

I set the maximum sized object to be 10MB

```
# The Maximum object to be scanned.
srv_clamav.MaxObjectSize 10M
```

And commented out the Viralator portions, since I'm not going to use that.

```
# And here the viralator-like mode.
# where to save documents
# srv_clamav.VirSaveDir /srv/www/htdocs/downloads/
# from where the documents can be retrieved (you can find the get_file.pl script in contrib dir)
# srv_clamav.VirHTTPServer "http://fortune/cgi-bin/get\_file.pl?username=%f&remove=1&file="
# The refresh rate....
# srv_clamav.VirUpdateTime 15
# For which filetypes the "virelator like mode" will be used.
# srv_clamav.VirScanFileTypes ARCHIVE EXECUTABLE
```

Now that the server is configured

It's time to use that eicar.com test file:

Fire up the icap server:

```
$ sudo /opt/icap/bin/c-icap -N -D -d 10
```

The first argument *-N* prevents the c-icap server from forking in the background, the second argument *-D* enables the printing of messages to standard output, and the third argument *-d 10* enables the printing of full debugging information.

You'll see lots of diagnostic information.

Next, try out the icap client:  
\$ /opt/icap/bin/icap-client  
ICAP server:localhost, ip:127.0.0.1, port:1344

OPTIONS:  
Allow 204: Yes  
Preview: 1024  
Keep alive: Yes

ICAP HEADERS:  
ICAP/1.0 200 OK  
Methods: RESPMOD, REQMOD  
Service: C-ICAP/030606 server - Echo demo service  
ISTag: "5BDEEEA9-12E4-2"  
Max-Connections: 20  
Options-TTL: 3600  
Date: Sun, 04 Jun 2006 16:18:55 GMT  
Preview: 1024  
Allow: 204  
Transfer-Preview: \*  
Encapsulated: null-body=0

It works. Now we're ready to see if this thing can really catch a virus using the eicar.com that you downloaded.

```
/opt/icap/bin/icap-client -f $HOME/eicar.com \  
-s "srv_clamav?allow204=on&force=on&sizelimit=off&mode=simple"
```

ICAP server:localhost, ip:127.0.0.1, port:1344

VIRUS FOUND

You try to upload/download a file that contain the virus  
ClamAV-Test-File  
This message generated by C-ICAP srvClamAV/antivirus module

It found the test virus, so we're ready to hook up the vscan service to ClamAV with c-icap as the bridge.

Create a vscan scanning engine:

As root :

```
# ./vscanadm set-engine -p host=localhost <engine_name>
```

# ./vscanadm show to see the details of the engine that was created:

```
max-size=10M  
max-size-action=allow  
types=+*
```

```
avscan:enable=on  
avscan:host=vj
```

```
avscan:port=1344
avscan:max-connection=32
```

The default for max-size is 1 GB, I set mine to 10 MB since I'm not sure how this is going to act with respect to CPU & memory utilization.

```
# ./vscanadm -p max-size=10MB
```

This is a global setting and will affect all scanning engines.

Now you enable scanning on ZFS filesystems:

```
# ./zfs set vscan=on tank/home
```

```
$ zfs get vscan tank/home
```

```
NAME    PROPERTY VALUE    SOURCE
tank/home vscan   on      local
```

Shows that scanning is enabled for the tank/home filesystem.

Another check on the parent tank shows that it's not enabled:

```
$ zfs get vscan tank
```

```
NAME    PROPERTY VALUE    SOURCE
tank    vscan   off     default
```

Now for the acid test. If you've got that eicar.com file in the VSCAN enabled filesystem, try to open it:

```
$ cat eicar.com
```

```
cat: cannot open eicar.com: Permission denied
```

```
$ mv eicar.com virus.com
```

```
mv: cannot rename eicar.com to virus.com: Permission denied
```

Even as root, you can't do much with it:

```
# cat eicar.com
```

```
cat: cannot open eicar.com: Permission denied
```

```
# mv eicar.com virus.com
```

```
mv: cannot rename eicar.com to virus.com: Permission denied
```

```
$ ls -al eicar.com
```

```
-rw-r--r--  1 fubar staff    68 Dec  8 13:31 eicar.com
```

Yes, it really does exist.

You can delete it, assuming you've got the proper permissions:

```
$ rm eicar.com
```

```
$ ls -al eicar.com
```

```
eicar.com: No such file or directory
```

This is a quick 'n dirty howto, it's not very comprehensive and is only meant to give you some ideas about how to begin using the vscan service OpenSolaris. I don't know if VSCAN works with UFS

filesystems ( doubt it) , but most of my filesystems are ZFS these days, and I only download to specific (now vscan enabled!) filesystems .