Creating Antivirus Signatures
http://www.TheBillyGoatCurse.com

Antivirus scanners work with a signature, which means they compare a file to a list of known viruses. If the file shows up in the list, it means it is a virus and is dealt with accordingly. A signature is made of a small part of the full virus, typically a part of the file that is unique.

The Clam Antivirus Project (http://www.clamav.net/, http://clamav.sourceforge.net) is an open source virus scanner available for free. Clam allows its users to create their own virus signatures, which is helpful if you discover a piece of malware that is not currently detected by Clam. This tutorial will show you how to create a signature file that can be used by any newer version of Clam on any platform it is used on, although I have used and written this tutorial based on my experience on the Windows port, ClamWin (http://www.clamwin.net). Necessary files to complete this tutorial are located here:

http://www.antionline.com/attachment.php?s=&postid=794651

1. Strings

Strings is a tool that spits out strings contained in a file. If you are on Windows, get it from this address:

http://www.sysinternals.com/ntw2k/source/misc.shtml

*nix boxes should already have it. Once downloaded, make sure it is in a folder included in your path. If you have no idea what that means, put the strings.exe file in your c:/windows/system32 folder. Inside the .zip file (attached) is our virus. To make our signature, we need to find a spot inside of it that is unique. We are going to hope that our virus has a string that is unique inside of it. Open a command prompt or terminal and run this command in the folder where virus.exe lives:

strings virus.exe > stringout.txt

You should now have a file called stringout.txt in that folder. Open that text file in a text editor and investigate.

Did you spot it? UltraVirus5000?

Our virus writer (me) left behind a unique trait in his file. We will use this spot to make our signature.

2. Hex Editing
Open a Hex Editor that will let you copy paste. I found that some hex editors do not have good copy paste functionality; I found this freeware that does (http://www.chmaas.handshake.de/delphi/freeware/xvi32/xvi32.htm). Open virus.exe with the hex editor.

On the far right column, you will see an ASCII representation of the file. The middle column is the Hex representation of the file. If your hex editor will let you, search for the string "UltraVirus5000". That spot should now be highlighted on the right and middle columns.

If we were only going to use the string "UltraVirus5000" for our signature, then Clam would identify every file with "UltraVirus5000" as a virus. That is called a false positive, and it is bad. So we are not going to use "UltraVirus5000" as our signature, but instead use "UltraVirus5000" and the binary surrounding it.

Highlight "UltraVirus5000" and about 10 characters before and after it. You should see the Hex being highlighted in the middle column as well. We now need to copy the hex, and this will differ between hex editing applications. Remember copy the highlighted section in the middle column, (the hex), not the ASCII on the right.
##Note | Do not select hex that starts in 00, or it won't work well with older versions.

3. Building the Signature

Copy the hex into a plain text editor like vi or notepad. Remove all the spaces using find -> replace. Now is when we name our virus. Clam has provided us with advice on naming.

-remember about the (Clam) marker (it's automatically removed by the parser)
-use the most popular name of the virus/worm
-don't use white characters or slashes in virus names

prefixes for particular malware
- -Worm for worms
- -Trojan for backdoor programs
- -JS for Java Script malware
- -VBS for VBS malware
- -W97M, W2000M for Word macros
- -X97M, X2000M for Excel macros
- -DoS for Denial of Service attack software
- -VirTool for virus construction kits
- -Dialer for dialers
- -Joke for hoaxes

More info here:
http://securityresponse.symantec.com/avcenter/vnameinfo.html
http://securityresponse.symantec.com/avcenter/reference/virus.and.vulnerability.pdf

##IMPORTANT: -don't use white characters or slashes in virus names ##

In our text editor, here is the format for a signature (single line)

Worm.BugBear.A (Clam)=63023a204170616368651933 2e3236202855a25 1b1db7678291d44a5653a2760a56eadb00a022d74

You will replace the hex above with the hex you pasted into the editor. "(Clam)" will be removed by the parser, and you can put whatever you want.

Save this file with a .db extension. In your ClamWin.conf, (Documents and Settings in Windows, probably /etc in *nix) find your database folder on the line:
database = C:\whateverpathyourdatabaseisin

Save your .db in the path shown on that line.

4. Test

Run Clam on the virus extracted from the .zip folder. If it is detected, then you have been successful in making a signature. If not, make sure you followed the steps correctly.

False positives are dangerous. Make sure you do not use signatures widely until you test them first.

Wildcards

There may be a scenario where you want to include 2 hex strings your definition.
Although this isn't a perfect example, it is similar to this programming syntax:

```
if(virus.contains(HexString1) && virus.contains(HexString2))
virus.quarantine();
```

This wildcard syntax is similar to that, however, the first hex string has to appear before the second hex string.

So heres an example, you have :

Hex String 1=384250530001000000000000000030000

and...

Hex String 2=000000000003842494D04240000000012293C3F787061636B657420626567696E3D27

that appear in different places in the virus. Some virus are polymorphic, and their contents will change. Although, some of these so called polymorphic worms will maintain some areas that don't change. This is a scenario where a signature like this is helpful.

The way to combine these strings is with the wildcard *. This is not the typical wildcard, where it means it can be any byte, but instead it can be any NUMBER of ANY TYPE of bytes. Here is the signature that requires those 2 strings, appearing one after another.

The signature:

Viri.A (Clam)
=384250530001000000000000000030000*000000000003842494D04240000000012293C3F787061636B657420626567696E3D27

This means it will need to have the hex string before the *, and the second string anywhere after the first hex string.

Advanced ClamAV Signatures

We will cover new signature syntax included in ClamAV .80rc 1,2, and 3. New features include extended wildcards, MD5 signatures, and an extended signature format. Stable versions of .80 are not released at the time of this tutorial, so this serves as a preview and may not be identical to the signature syntax at the time of its release.

Required files: http://www.antionline.com/attachment.php?s=&postid=794651

Extended Wildcards

New wildcards will be included in new versions of ClamAV. These include ?, {n}, {-n}, {n-}, and (a|b).

Wildcard- ?
We will be using the polymorphicworm.A series for this section (A series). The A series is a mass mailing worm, using a randomly generated subject line that is changed in each sent binary(i.e. g3t s8m5 v7ag28 , get some viagra). We will create a signature that will target all 3 in the A series.

The ? mark is simple, it is like a regular wildcard that you will find anywhere else. Run the A series through strings, and you will see a similarity in the subject line. "g3t s8m5 v7ag28", "g6t s2m7 v8ag65", "g9t s7m3 v6ag18" are all very similar. Only parts of them are polymorphic, not the whole subject line. We will be able to a create a signature that replaces the numbers with the "?" wildcard.
The hex string is:
67 33 74 20 73 38 6D 35 20 76 37 61 67 32 38

When we apply the "?" wildcard for that string, it is:
67 ?? 74 20 73 ?? 6D ?? 20 76 ?? 61 67 ?? ??

Add some binary before and after to prevent false positives, remove spaces and format:
Hoax.Series.A (Soda)=C300000000000000067??742073??6D??2076??6167????005589E583EC3453

This can get a bit buggy when you use a lot of wildcards. When in doubt, add more binary to the sig.

Wildcard- {n}

We will be using the polymorphicworm.B series for this section (B series). The B series has polymorphic code, but the 2 polymorphic hex strings always maintain a certain length between eachother. We will create a signature that will target all 3 in the B series.

Run through the series in strings, you will see "polymorphiccode" twice in each virus. Notice that the second string is 18 bytes away from the first in each instance. Here is the signature:
Hoax.Series.B (Soda)
=E583EC0883C4F46A02A120724100FFD0E879FFFFFFC9C3000000000000000706F6C796D6F7270686963636F6465??{18}706F6C796D6F7270686963636F6465??0089F65589E583EC5453E8

The first hex string starts with binary, ends with hex for "polymorphiccode" and ?? for the random number. {18} means the next string will appear 18 bytes away from the first. The second string begins with "polymorphiccode, ?? for the random number, and binary to avoid a false positive.

You can also use {-n}, which means the next string will appear less than n bytes away, or {n-}, which is more than n bytes away.

Wildcard- (a|b)

We will be using the polymorphicworm.C series for this section (C series). The C series has polymorphic code, but the polymorphic hex string only changes a certain character. We will create a signature that will target all 3 in the C series.

This wildcard allows us to specify specific values to look for, instead of all values, like the ? wildcard.

Run the C series through strings. You will see "Hello Rod", "Hello Tod", and "Hello Nod". Instead of using a "?" wildcard, we can specify the values we want to detect. The only difference in the string is the beginning of the name. Our signature will look like this:

Hoax.Series.C (Soda)=FFC9C3000000000000048656C6C6F20(4E|54|52)6F640089F65589E583EC3453E864

(4E|54|52) Translates to R, T, or N, the beginning of Rod, Tod, and Nod. This helps minimize a false positive in bigger situations. That signature also begins and ends with binary.

Extended Signature Format

New Clam versions will allow an extended signature format, which should allow more targeted and faster performance. Here is the syntax:

MalwareName:TargetType:Offset:HexSignature

MalwareName

Anything you want, just don't include colons or bad characters.

TargetType

TargetType is the type of file you want to appy the hex signature to.
0- Any file
1- Portable Executable Format
2- OLE2 Component (i.e vb script)
3- Normalised HTML (Will decode javascript)
4- Mail
5- Graphics

Offset

Where does the signature appear in the file (similar to {n} wildcard)
*- anywhere
n- n bytes from start of file
EOF - n- End of file, minus n bytes.

HexSignature

Same as the old format.

MD5 Signatures

MD5 signatures are a very quick and very accurate way to identify a static virus. Any sort of polymorphism will defeat a MD5 signature. To create an MD5 signature, we will use a program included in the clam download called sigtool. The syntax is:

```
sigtool —-md5 target.ext
```

The output is a proper signature, so it would be correct to use syntax like this:

```
sigtool --md5 sub7.exe > test.hdb
```

Our MD5 signature has to have an .hdb extension.

FYI- None of the attached files are actually viruses. The examples shown are not typical signature examples, but they demonstrate the syntax. Custom AV signatures have a great risk of a false positive and can cause serious damage if not tested properly before use. **Make sure you test your signatures before you implement them on production machines.**