Argument for slow infection and slow polymorphism

Argument for slow infection
and slow polymorphism
by Rogue Warrior

Slow infectors

Many people say that fast infectors are better than slow infectors but I have to disagree. The goal of a virus is to travel to as many hosts as possible. Agreed?

For a virus to do this it must do certain things:

1. Not be noticed (stealth).
2. Infect files which travel outside the current host.

Ok, so we write a stealth fast-infector and hey presto it's all fixed right? Wrong...

There is a reason for this, point #1 is really two points:

1.1 Cannot be detected (file/disk stealth).
1.2 Cannot be noticed by the user (speed, memory, disk space).

Most coders forget about 1.2 and their virus is has a very obvious appearance on the host system - causing it to slow down to a snail pace at some times.

In virus forums there have been many cases of users saying "I noticed something happening - so I investigated and found virus X - how do I remove it?" - Our goal is to avoid this suspicion in the first place.

Think about AIDS/HIV ? it lies dormant for up to 10 years infecting people it comes into contact with - but showing little in the way of symptoms.

The same applies to viruses they must not slow down the system or cause incompatibilities (QEMM exceptions/crashes etc) because human users are very good at noticing differences in the way their computer is acting.

This leads us to slow infection techniques, now if the goal is to get a virus to another system then all we need to do is infect files which are highly likely to be moved to another computer system, these files are:

1. Files on Floppy Diskettes.
2. Files on Network/Remote devices.
3. Files opened while in a communications program.
4. Files opened while in a compression program.
5. Files opened while in a backup program.

These files all have a chance of leaving the system.

Floppy disks :- 14-y/o pirates use these to swap their games, perfect way to reach out to other hosts.

Network devices :- useful if you have supervisor access you can by chance infect a file like login.exe and

then you have all 250 computers (or however many!)
working as infection posts.

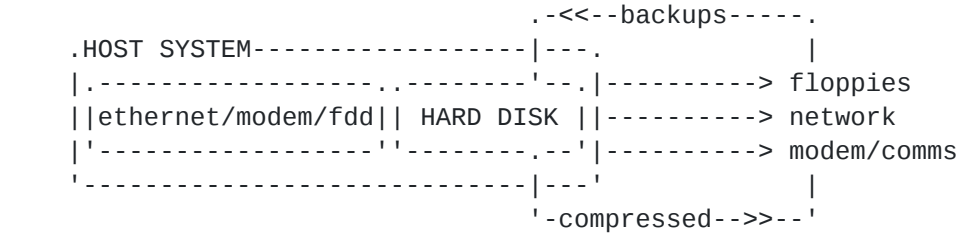
N.B.,
Novell Netware allows storage of complete bootable
disks on the server as files. So the network
client just boots up from a file on the server.
These files are useful targets also but I am
lacking a network to create/test such a virus.

Comms program :- anyone uploading a EXE/COM file is doing
it for a reason (probably to send to someone). Another
perfect way to infect one or *many* other hosts.

Compression program :- usually these are used to compress
software before uploading/downloading with comms programs
so if we infect EXE/COM programs while compression
is in progress then we have a good chance of making it
outside.

Backup program :- this is just to guard against removal of
our virus from the host - if the user discovers the
virus by using an AV program then we can reinfect.

Example diagram:



Infecting hard disk _files_ is useless since they never
travel outside - not like infecting files on floppies,
networks and comms channels.

Its a good idea however to make sure your virus is loaded
before other programs so that stealth is active. You
should either automatically infect \COMMAND.COM or make
the virus multipartite (infect mbr/track 0).

Implementation of this method:

The best implementation is this:

Floppy diskette and CDR (CD-ROM Writers) check:
Use AX=4408, DL=Logical Drive#, INT 21h

Network/Remote check:
Use AX=4409, DL=Logical Drive#, INT 21h
Use AX=440A, BX=File Handle, INT 21h

Comms/Compression/Backup check:

Use a lookup table (LUT) to disable stealth and
activate the fast-infector at opportune times:

'CHKDSK'   - Stops CHKDSK errors (well known).
'SCANDISK' - Ditto.
'NDD'   - Ditto.
'PKLITE'   - Protect virus in PKLITE compression wrapper.
'DIET'   - Protect virus in DIET compression wrapper.
'LZ'   - Protect virus in LZEXE compression wrapper.
'TM'   - Telemate, to infect EXE/COM uploads.
'TE'   - Telix/Terminate, to infect EXE/COM uploads.
'BACKUP'   - To infect executable backups.
'MSBACKUP' - Ditto.
'CPBACKUP' - Ditto.

Also use the following LUT to look when a compression
program output file is opened.

    'ZIP' - PKZIP tmpfile extension opened.
    'LZH' - LHA tmpfile extension opened.
    'ARJ' - ARJ tmpfile extension opened.
    'ARC' - ARC tmpfile extension opened.
    'RAR' - RAR tmpfile extension opened.

Check the file's extension when you intercept an
INT 21h AH=3C and INT 21h AH=3D call.

If the file which is opened is one of the above
then activate the fast infector and disable the
stealth.

When it is closed (use SFT's to check with AX=1220/INT2F
AX=1216/INT2F) then reenable the slow infector and
reenable the stealth.

==========

Slow Polymorphism

This is a less contentious issue, slow polymorphism is
designed almost entirely to annoy the anti-virus community.

These people receive lots of viruses everyday and they can't
go around disassembling each one to the most minute detail,
this is good for us because it means we can annoy them into
wasting time.

Together with anti-goat techniques explained by Rajaat this
will be rather annoying. The anti-virus community will have
to spend at least a day analyzing a good polymorphic virus
to:
    a) get it to infect their goat files
    b) get it to generate a huge number of samples
       (they did about 200,000 samples for SMEG I believe).

```
    BTW: If you decided to make your code dependant on the
         integrity of the anti-goat code then you will find
         the researchers have even more problems in spoon
         feeding the virus 200,000 good samples :)
```

Implementation

```
    Poly of course is based on random number generation how
    to select a random number which changes slowly? Well I
    can think of two nice ways:
        1.  BIOS date.      -very slow poly only changes with
                            each computer!
        2.  Today's date.   -quite slow.

    ------------------------------------------------------------

    1.  push    0FFFF
        pop     ds
        mov     si,0005             ;DS:SI -> FFFF:0005 (8 bytes).
        xor     bx,bx               ;value=0
        mov     cx,4                ;size=4 words.
    L1: lodsw                       ;fetch word.
        add     bx,ax               ;checksum.
        loop    L1                  ;next 3 words.
        mov     ds:rnd_seed,bx      ;set seed.

    ------------------------------------------------------------

    2.  mov     ah,2A
        int     21                  ;get date.
        rol     dx,cl               ;random adjustment.
        xor     dx,cx               ;place cx into eqn.
        mov     ds:rnd_seed,dx      ;set seed.
```