GIAC Certified Security Essentials (GSEC) Practical Examination

Scott Alan Sumner
March 16, 2001

Hostile Code: A Holistic Methodology

Summary
A good deal of contemporary security doctrine discuss virus; yet this is only a subset of the malicious or hostile code that exists. Various terms and definitions for hostile code are subject to semantic dissection, but in simplest terms, hostile code is code that is when executed causes some harm on the systems it resides on or is connected to. The purpose of this paper is to discuss a reasonable and effective approach at reducing the risk associated with this type of threat, with a particular emphasis on layering the various tools and techniques herein.

Definitions
For the purposes of this document, a virus is defined as segment of computer code which attaches or insinuates itself to another piece of code. A virus can infect a file (file infector), a boot record (boot sector) and the most recent evolution, the macro-virus. One of the most useful classification schemes is the SANS Incident Handling Summary Description[25].

- Program Infector
  - Direct Action –This type finds a specific program to attach itself to.
  - Resident Virus – Installs itself into memory, usually a reserved OS segment.
  - Cluster – Modify associated file system information of a specific program.
  - COM – Specific to WinTel .COM files
    - Prepending – Inserts itself in the very beginning of the target code.
    - Appending – Appends itself to the end of the target code, but place a JMP to the appropriate location at the very beginning of the target code.
    - Overwrite – Completely overwrite the target code, destroying it.
  - EXE – Specific to WinTel .EXE files.
    - The virus alters the value of both the Code Segment and Instruction Pointer to accommodate the payload.
- Boot Infector
  - Floppy Boot Record Infector.
  - Master Boot Record Infector.
  - DOS Boot Sector Infector.
- Multipartite – Hybrid between a boot infector and program infector
- Macro – Target data files, and are written in a specific macro language

Other types of hostile code include "Trojans[2]", which are a stand-alone piece of software that presents a usable set of functions, which concurrently executes an unintended payload and "Worms[4,14,15,17,18,20,21,23,24]", which is are stand-alone pieces of software, that may or may not spread itself, but exists without being parasitic.

Hostile web pages are URL's that execute hostile code when accessed. In particular, the Java implementation has historically posed some risks.

There is some cross-pollination between the types, hybrids and there are some sorts of hostile code that defy clear classification altogether. Trojans can open web pages; Web pages can load virus, etc. The classic definitions tend to break down, and sometimes these cross environments. It is rare than an organization runs a homogenous environment across the board. Workstations may run Windows 98/NT, Servers may run Netware and everything points to a Solaris Firewall. This is a two-edged sword, in that the difference in environments limits the propagation, but the effort required to maintain consistent protection in all the environments is proportionally larger.

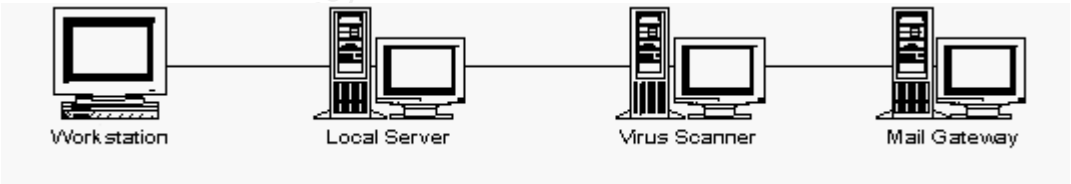
Although most code is either application or OS specific, v irus and trojans have been found on IBM mainframes and Palm Pilots. Hostile Code can infect just about any environment[5,6].

Approach

A traditional approach is to dictate, "Install virus scanning software into each and every workstation" with the belief that is sufficient to mitigate the risk to a minimum. This is fundamentally flawed for a number of reasons:

- If the virus signatures aren't up to date, you are at risk.
- If the signatures are flawed, you are at risk.
- If your software is disabled, you are at risk.
- If the virus has no known signature, you are at risk.
- If the software is misconfigured, you are at risk.
- The workstations are not the only environment that can be infected.

The first flaw is the one most likely to manifest itself, because any complex system will ultimately fail. Resources are finite, and eventually an update will fail to be applied. Layering the approach can reduce the risk. Here's an example, applied to an email stream:



A virus-scanning gateway resides in the stream. This will not deal with viruses brought in on floppy, and is also subject to the other problems listed above – but there are significant gains to be made. If the updates are applied to the mail scanner first, and if email is the primary mode of infection, then you have reduced the risk. If everything is working, then mail is a much less likely source of infection.

What do you if the signatures are flawed? This is not unheard of -- a signature late last year from one of the primary vendors completely flattened 8 out of 2000 workstations in our environment. That particular situation wasn't crippling, but it could have been much

worse. In this instance, traditional QA techniques were applied to the signature and the signature passed, but despite that, a number of systems had to be rebuilt.

Scanning on the local file services server helps. Again, it is subject to some the limitations of the workstation virus scanner, but it helps in the case of the occasional workstation missing an update and can limit the damage.

So what can you do if there is no known signature? Most of the time, the larger virus scanning software companies are fairly diligent about getting new signatures out. However, the fact remains – you can't get them all. Limited resources, new techniques, and the speed of propagation all factor into this. One method to limit this is to implement an email policy system. The best way to illustrate this is to list the settings:

Rule A: 10 identical attachments in a 30-minute period. React automatically, with an email message. Page Security and Operations. Perform signature upgrade.
Rule B: 25 identical attachments in a 30-minute period. React automatically, with an email message. Page Security and Operations. Block email with a specific attachment name (or all attachments).
Rule C: 50 identical attachment types in a 30-minute period. React automatically, with an email message. Page Security and Operations. Perform signature upgrade.
Rule D: 100 identical attachment types in a 30-minute period. React automatically, with an email message. Page Security and Operations. Block email with any attachments.

Note the cascading effect; the first three rule blocks identical attachments by name, the last blocks by type. Of course, this will shut down any SPAM efforts in your organization (e.g., "aggressive internet marketing") – but this is a powerful tool.

By limiting the broadcast of email, especially with attachments, you reduce the inroads that hostile code can gain. Like the other mechanisms, it is an incremental improvement – not complete unto itself.

Another mechanism that can be used to help against virus that has no known commercial signature file is to have a routine on your firewall, which looks for specific attachments. This works well when you have some detailed information about a piece of malware (such as an attachment name), the method of infection is email, and the virus companies haven't issued a virus engine update (or you haven't rolled it out, etc.).

<u>The whole nine yards</u>
Up to this point, we have discussed items that are fairly specific to virus – but there are many other varieties of hostile code in the wild. Implementing a trojan scanner can provide another measure of protection. Most of the more common trojans are not caught by the industrial virus scanners. The trojan are not as product-mature as virus scanners, but commercial packages do exist and are fairly straightforward to implement.

As for hostile Web Pages, a properly configured browser can limit or even prevent damage. Most of the more recent browsers have very granular security settings, which can allow legitimate applications to run, but prevent malicious code from executing.

An aggressive patch program can be a significant deterrent. It is important to not only patch the operating systems themselves, but components and applications. An excellent example of an application level patch that helps repel hostile code was the Microsoft Outlook patch that blocked most permutations of the ILOVEYOU[4] worm. Operating system components, such as the Microsoft Internet Explorer also should be aggressively patched.

Policies

An effective approach requires treatment in the organization policies. Examples of effective policies are efforts to minimize points of entry (use only one email system), directing the policies at users, not just employees; immediate punitive action at users who engage in virus generation/hacking/inappropriate behavior, and inhibit the disabling of software. Policies unto themselves are ineffective unless there is an audit of compliance followed by corrective action.

Education

Education is probably the most overlooked component, but it can have the biggest impact. A common mistake is to spend effort, time and money for education, and the following week a user launches a VB script because it purportedly has nude photos of a famous (or not-so-famous) person in it[20] – and renders the workstation inoperative. On the surface it appears that educating end-users is futile, and many people throw up their hands.

The important things to understand are:

a) End-users are not, nor will ever be, experts in social engineering and hostile code. That's your job.
b) If education is reasonably effective, for every duped person who runs a trojan, there are at least two or three who don't – and you will never see that intangible result.
c) Education is a dynamic, continuous process – it never stops.

Conclusion

The point of this is to emphasize that in order for an organization to be properly defended, a collage of techniques and tools must used in a complementary manner. The system is degraded each time something is missing, and the effect is cumulative. However, the benefit of having some overlap provides some addition protection in the event of a failure of a component.

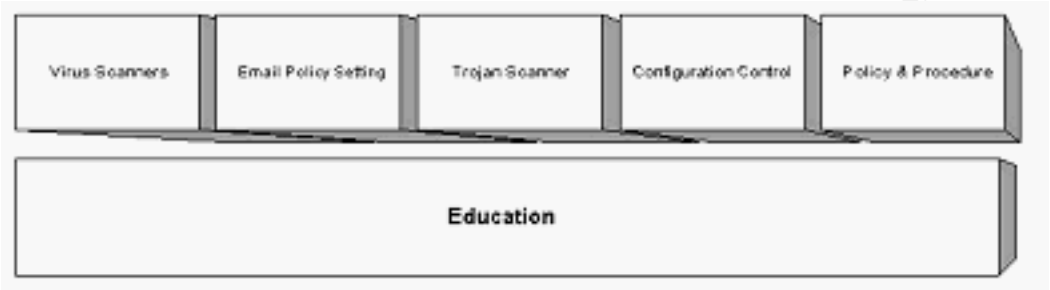


Figure 2

Finally, the practical issue is how do you sell this to your management? Small firms can't afford it and big firms are incredibly ponderous. Some folks have the autonomy, budgets and time to implement all of the afore-mentioned items; others have to pick and choose. Sometimes, timing is everything – it is a lot easier to justify this expense in the wake of a major event such as the "Melissa" incident. Implement what you can; it doesn't cost a lot to provide education to the IT department, especially if you are resourceful. Create an option paper, detailing pros and cons of implementing various components, and recommend a course of action. It may be a case that Senior Management is simply not aware of the risks.

The point of this document is to illustrate than an effective strategy for dealing with malicious software cannot rely on a single system or even set of systems. It must be layered, with complementary components. It must be dynamic, and periodically reviewed end to end. This holistic approach can be applied in the broader security sense as well.

References:

1. Schneier, Bruce *Secrets and Lies: Digital Security in a Networked World* New York: John Wiley & Sons, Inc. 2000. pg 151-175.

2. "Virus Trojans and Hoaxes" Infosyssec Security Products
URL http://www.infosyssec.com/infosyssec/virwor1.htm (2001)

3. "The Zkey Exploit: The Capability of Malicious JavaScript Code" SANS GSEC Practicals, David Rothermel
URL http://www.sans.org/infosecFAQ/malicious/zkey.htm (August 28, 2000)

4. "The ILOVEYOU Worm" SANS GSEC Practicals, Tracey Whalen
URL http://www.sans.org/infosecFAQ/malicious/iloveyou_worm2.htm (September 9, 2000)

5. "A Virus in the Palm of My Hand" SANS GSEC Practicals, Allan Hollowell
URL http://www.sans.org/infosecFAQ/PDAs/virus_in_hand.htm (September 13, 2000)

6. "The Palm OS and Malicious Code" SANS GSEC Practicals, Hugh Taylor
URL http://www.sans.org/infosecFAQ/PDAs/palm_OS.htm (September 12, 2000)

7. "Email Viruses - Comparison of Novell GroupWise 5.5 and Microsoft Outlook" SANS GSEC Practicals, Katherine Wamer
URL http://www.sans.org/infosecFAQ/win/email_virus.htm (September 8, 2000)

8. "Information System Security Education, Training, & Awareness for Web Administration – An Integral Part of Defense-in-Depth" SANS GSEC Practicals,Ray Letteer
URL http://www.sans.org/infosecFAQ/legal/infosec_edu.htm (September 16, 2000)

9. "VBS Viruses – Why and how do they spread so quickly?" SANS GSEC Practicals, Stephen Wing
URL http://www.sans.org/infosecFAQ/malicious/VBS.htm (August 29, 2000)

10. "Layered Security: An ISP Case Study with Cisco and Solaris" SANS GSEC Practicals, Rockie Brockway
URL http://www.sans.org/infosecFAQ/firewall/layered_sec.htm (October 19, 2000)

11. "Anti-Virus Architecture: A 4-Layered Approach" SANS GSEC Practicals, Larry Sobers
URL http://www.sans.org/infosecFAQ/malicious/anti-virus.htm (October 31, 2000)

12. "Merry Christmas - The NAVIDAD Virus" SANS GSEC Practicals, David Cullison
URL http://www.sans.org/infosecFAQ/malicious/navidad.htm (November 20, 2000)

13. "Computer Virology: Protection, Prevention, Identification & Containment" SANS GSEC Practicals, Jules Fiorentino
URL http://www.sans.org/infosecFAQ/malicious/virology.htm (November 22, 2000)

14. "The Kak Worm: New Public Enemy #1?" SANS GSEC Practicals, Safka
URL http://www.sans.org/infosecFAQ/malicious/kak.htm (July 16, 2000)

15. "Cyber Threats: Viruses, Worms, Trojans, and DoS Attacks" SANS GSEC Practicals,
Scott Hobbs
URL http://www.sans.org/infosecFAQ/malicious/threats.htm (December 18, 2000)

16. "How Viruses Attach" SANS GSEC Practicals, Bernard McCargo
URL http://www.sans.org/infosecFAQ/malicious/attach.htm (December 8, 2000)

17. "What is the VBS.Stages.A Worm?" SANS GSEC Practicals, Patricia Dooley
URL http://www.sans.org/infosecFAQ/malicious/VBS_stages.htm (November 28, 2000)

18. "The W32.Navidad@M Worm" SANS GSEC Practicals, Michael Stan
URL http://www.sans.org/infosecFAQ/malicious/W32navidad.htm (December 1, 2000)

19. "The Davina Virus – can we defend against the latest vectors?" SANS GSEC Practicals, Scott Morley
URL http://www.sans.org/infosecFAQ/malicious/davina.htm (February 16, 2001)

20. "VBS Worms Generator" SANS GSEC Practicals, Alberto Grazi
URL http://www.sans.org/infosecFAQ/malicious/VBS_worms.htm (February 21, 2001)

21. "The FunLove Virus Worm" SANS GSEC Practicals, Douglas Dodge
URL http://www.sans.org/infosecFAQ/malicious/funlove.htm (December 2, 2000)

22. "Virus Generators and their implications" SANS GSEC Practicals, Bryan Fansler
URL http://www.sans.org/infosecFAQ/malicious/generators.htm (February 19, 2001)

23. "The Morris Worm: how it Affected Computer Security and Lessons Learnd (sic) by it" SANS GSEC Practicals, Larry Boettger
URL http://www.sans.org/infosecFAQ/malicious/morris.htm (December 24, 2000)

24. "The W32.HLLW.Bymer Worm" SANS GSEC Practicals, John Bartolick
URL http://www.sans.org/infosecFAQ/malicious/bymer.htm (February 10, 2001)

25. "Malicious Software (Malware)" SANS GIAC Incident Handling Foundations, Fred Kirby
Available via URL http://www.sans.org (2001) pg 3 - 8