A Methodology for Evaluation of Host-Based Intrusion Prevention Systems and Its Application

Keith G. Labbe, Neil C. Rowe, and J. D. Fulp

Extended Abstract

Host-based intrusion-prevention systems are currently popular technologies which try to prevent exploits from succeeding on a host. They are like host-based intrusion-detection systems [1] but include means to automatically take actions once malicious activities or code are discovered. This can include terminating connections, services, or ports; refusing commands; blocking packets from specific Internet addresses; initiating tracing of packets; and sending modified packets back to a user. Automated responses to exploits can be quick without human intervention. Around ten commercial vendors are currently offering intrusion-prevention products [2], and Snort-Inline [3] is a popular open-source tool. Total intrusion prevention is a difficult goal to achieve, since it takes time to recognize an exploit and by then the damage may be done. So it is important to have a way to test the often-broad claims of intrusion-prevention products.

The testing we propose is not as comprehensive as that offered by attack-traffic simulators like Skaion?s TGS (www.skaion.com) or by the DETER testbed (www.deterlab.net). But attack-traffic simulators, even when up-to-date, only model broad characteristics of attacks and not their context-dependent behavior, so they can produce significant numbers of false negatives. DETER emulates rather than executes malicious software to provide added safety, which is not quite the same. DETER also imposes several bureaucratic obstacles for getting approval for experiments and obtaining time on their hardware to run them; this bureaucracy requires motivation and time to navigate. For quick testing in depth of a new product that has not been evaluated in DETER, or for finding reasons to rule out a product, a simpler approach that is easier to set up is required.

A. Experimental Setup

Our research developed a simple methodology for testing dynamic intrusion-prevention systems and applied it to two products, McAfee Entercept version 5.0 and the Cisco Security Agent version 4.5. (More details are provided in [4].) These products were chosen because our School was considering purchasing them. Our tests used live viruses, worms, Trojan horses, and remote exploits, which were turned loose on an isolated two-computer network. The computers were networked together and no other network connections were used. This configuration allowed us to use live exploits without infecting other computers or being affected by them.

The victim computer used the Microsoft Windows 2000 Advance Server (Service Pack 0) operating system with SQL 2000 and IIS 5.0 installed. These service packs and versions were chosen because they are out-of-date, which ensured many vulnerabilities. We used Norton Ghost to create an image of the victim machine to enable repeated restorations after each attack. The attacking computer also used the Microsoft Windows 2000 Advanced Server operating system and included an email server. Service packs were only installed as required by the management server.

Core Impact version 4.0.1 and Metasploit version 2.3 were installed on the attacking computer for attack tools; they were chosen because they are among the most general tools available and we had experience using them legally. Other attack tools could easily be tested with our configuration. To check that all exploits worked properly, we confirmed they worked against an unprotected image of the victim machine. We also confirmed that exploits were persistent after a restart of the victim computer. The exploits we tested were:

- Reconnaissance: Super Scan 4
- Core Impact remote exploits: IIS CGI Filename Decode, IIS Unicode, IIS IDS-IDQ, SQL Server Hello, MSRPC DCOM, SQL Server CAN-2002-0649, IIS ASN.1 Big String SpNeGo, MSRPC LSASS Buffer Overflow
- Metasploit remote exploits: iis_nsiislog_post ? win bind payload, iis50_printer_overflow ? win bind payload, iis50_webdav_ntdll ? win bind payload, msrpc_dcom_ms03_026 ? add user payload
- Email exploits: Iworm.lovegate.i., Iworm.Loveletter, Iworm.Klez.h, Iworm.Moodown, Iworm.Navidad.b, Iworm.Netsky.d, Worm.Win32.Chainsaw.a, Worm.win32.Donk.c, Backdoor.SdBot.aa, Win2k.inta.1688, Iworm.Radix, Iworm.Mydoom.g
- Web page exploits: Netbus Trojan version 1.7, Trojan.Win32.virtualroot, IIS-Worm.CodeGreen.a, Willow.2013, Win2k.Stream, Win32.Cabanas.b, Win32.Ghost.1667, Win32.HLLO.Zori, Win32.Lash.d, Win32.Matrix.Ordy.a, Win32.Redemption.b, Iworm.Mydoom.h
- Exploits introduced on a floppy disk: IIS-Worm.IIS Worm, Worm.Win32.Lovesan.a, Worm.Win32.Muma.C, Worm1, Worm2,

Win32.Small.2280, Iworm.Aliz, Worm.win32.sasser.b, Worm.win32.welchia.g, I-worm.bagle.at, Trojan.call911, Iworm.alanis

B. Results

Entercept stopped only 8 of 36 malicious code exploits in disk, email, and Web attacks. 10 of those 36 were viruses, which Entercept did not claim to protect against. Some of the behaviors it missed, such as mass emailing to everyone in the address book, are well-known worm behaviors. Additionally, many successful exploits were persistent, continuing to run even after the computer was restarted, unless Entercept was started at a high security level. For reconnaissance-phase activities, Entercept blocked all but one exploit we attempted, the creation of an administrator-level user account on the victim computer, but this exploit is well-known. In general, the McAfee product performed well against remote exploits with one important exception, but poorly against reconnaissance, email, Web, and disk-based exploits.

A false positive was occurred, at level 2 protection, every time the management console was started, a report that ?explorer.exe? was attempting to modify the management server. Other false positives occurred at level 3 protection when the user attempted to access Command Prompt or any Administrator Tools in the Windows Control Panel.

The Cisco Security Agent prevented all 12 of our remote phase exploits. Although two of those exploits resulted in a crash of the victim computer, execution of the exploits was blocked. The Agent also stopped 17 of our 36 malicious code exploits, more than twice the number of exploits that Entercept stopped. Additionally, only a few of the exploits were persistent after restart. As for the reconnaissance phase, our exploits were able to determine what ports were open, the RPC endpoints, and several other pieces of information, but their efforts to access most information were thwarted. The Cisco Security Agent appeared to prevent exploits from rewriting the system registries and adding new files, actions that if performed would have allowed the exploits to continue to run after a restart. However, it provided poor protection against the Netbus Trojan although it is not a new exploit. In general, the Cisco product was good against reconnaissance, remote, and email attacks; fair against the disk attacks; and poor against the Web attacks.

Attempting to open the Windows Control Panel via the Toolbar resulted in a false positive from the Cisco Security Agent. Also, in a Microsoft Outlook Email client, clicking on the ?Send/Receive? email button on the victim computer resulted in a false positive on the attacking machine; so to send or receive email, we had to disable the security on the latter.

C. Discussion

In summary, neither product fulfilled the claims of its marketing literature, but the Cisco product did better. McAfee Entercept did not stop most of our attacks against it even though most of those attacks were clearly within its claims of coverage. In addition, its alerts were insufficiently specific as to the signatures that were detected, so a security manager would have trouble deciding what to do next in many cases.

Although the Cisco Security Agent did not stop all our exploits, it did an adequate job minimizing the damage of those that were successful. Additionally, its ability to thwart reconnaissance efforts and remote exploits will reduce the chances of a successful remote attack, and the claim of its management server to manage up to 100,000 host agents is impressive. However, the Cisco product was more difficult and time-consuming to install than the McAfee product.

The exploits provided by Core Impact and Metasploit were not very recent, so it is likely that the products would perform still worse on more recent exploits. These results on dynamic intrusion prevention are consistent with reported poor results on static intrusion-prevention products [5]. The lack of commonly accepted standards and benchmarks for these products is a clear obstacle to progress since unwarranted claims have previously not been easy to test. But our experiments have shown a relatively simple way to test them.

We deliberately did not install more elaborate networking, a firewall, antivirus software, an intrusion-detection system, or an operating system with up-to-date patches in our tests because the products tested claimed such additional features were unnecessary to obtain useful protection from the product. We also thought it helpful to see what the products could do alone since a product is less useful if it only works in a specific environment or with specific additional software. However, it would be useful to measure performance under such modified configurations that are more like those of real systems to give more specific guidance to security managers.

REFERENCES

[1] Proctor, P. E., *Practical intrusion detection handbook*, Upper Saddle River, NJ: Prentice-Hall PTR, 2001.
[2] Desai, N., ?Intrusion Prevention Systems: The Next Step in the Evolution of IDS?, retrieved from www.securityfocus.com/infocus on May 25, 2006.
[3] The Honeynet Project, *Know Your Enemy: Learning about Security Threats, Second Edition*, Boston, MA: Addison-Wesley, 2004.
[4] Labbe, K. G., ?An Evaluation of Two Host-Based Intrusion-Prevention Systems,? M.S. thesis, U.S. Naval Postgraduate School, June 2005, www.cs.nps.navy.mil/people/faculty/rowe/oldstudents/labbe_thesis.htm.
[5] Wilander, M., and Kamkar, M., ?A Comparison of Publicly Available Tools for Static Intrusion Prevention,? 7th Nordic Workshop on Secure IT Systems, Karlstad, Sweden, November 2002.

Manuscript received on May 3, 2006. This work was supported in part by NSF under the Cyber Trust Program. Contact the authors at Code CS/Rp, 833 Dyer Road, U.S. Naval Postgraduate School, Monterey, CA 93943. Email: klabbe, ncrowe, and jdfulp at nps.edu. Opinions expressed are those of the authors and do not represent the policy of the U.S. Government.

This paper appeared in the Proceedings of the 7th IEEE Workshop on Information Assurance, West Point, NY, June 21-23 2006.