How to Defeat an IDS/IPS, an Overview

Antonio Martin (tmemail@gmail.com)

In understanding any defensive (and applicable to offensive) technology, its limitations and weaknesses must be thoroughly explored. A thorough understanding of the capabilities must be considered for proper deployment and operation. As with most marketing literature attempting to sell their defensive product, it become easy to identify what can be stopped / blocked by the products in question. The sales literature will outline, in some depth and embellishment, the strengths and capabilities presenting a picture of a single product that does all. The reality of any defensive system is a network must be protected with multiple layers, a defense in-depth, where products work in conjunction. Even in such a situation, no defensive system is fool proof.

As you find weaknesses within a system, the results can be applied as opportunities for new technologies, fixes or exploits. The dissemination of exploits and weaknesses always leads to the debate of security verse open dialog.

The US government's view on most weaknesses, as applied to current and even future systems, is usually contained within high level securities, limiting those who know about the threats. This results in many problems never being discovered by those who would exploit them prior to their being fixed; only those most intimate with the system will find vulnerabilities and are already trusted. Conversely, as is typical with any bureaucracy, since a solution might be a long way in coming, those who rely on said systems might not be considered in the need to know. Thus they are left working with known flawed systems that could be exploited, possibly risking lives.

In the other spectrum, the commercial world experiences many exploits being first discovered and then published openly. The argument being that a free flow of information will encourage fixes and advance technologies more rapidly. The real world results tends to be those who would exploit systems, be it a Windows™ based or Linux, can take advantage of the weakness well before any solution is available.

Both stances have issues and are rife with criticism and mudslinging.

The purpose of this document is to demonstrate to security professional some limits of the capabilities of an Intrusion Detection and Intrusion Prevention System. It is intended to be a learning tool for those who protect systems, and thus, the topic matter and depth is carefully pruned / selected.

Part of understanding how to attack a network requires an insight as to how a network administrator and/or architect view security. The prime considerations being:
Cost of Security in Dollars – The best security is the world will be for not if companies cannot afford the technology. This cost is usually a troubling upfront price with a less troubling, reoccurring yearly maintenance or update charge.
Cost of Time to Administration – As more parts are added to a system, the more difficult it is to administer. Since many of the choices and recommendations for security technology are made by those who are responsible for such operations, their own "pain" factor is weighed into the recommendation. If a system is considered difficult to maintain and operate, an administrator will shy from its deployment as the difficulty will be theirs.

Cost of Productivity Lost – Further adding to this mix, as the levels of the security increase, productivity decreases as users face greater complexity; this will result in a backlash against the network administration staff.
All Verses
Cost of Not Deploying Technology *X* – This is a valid consideration when trying to balance budgets, time and schedule. How much protection does the system afford and is that extra level of protection worth while? If the means is not utilized, what could happen and how much will it cost to repair or fix verse the percentage chance that it will happen.

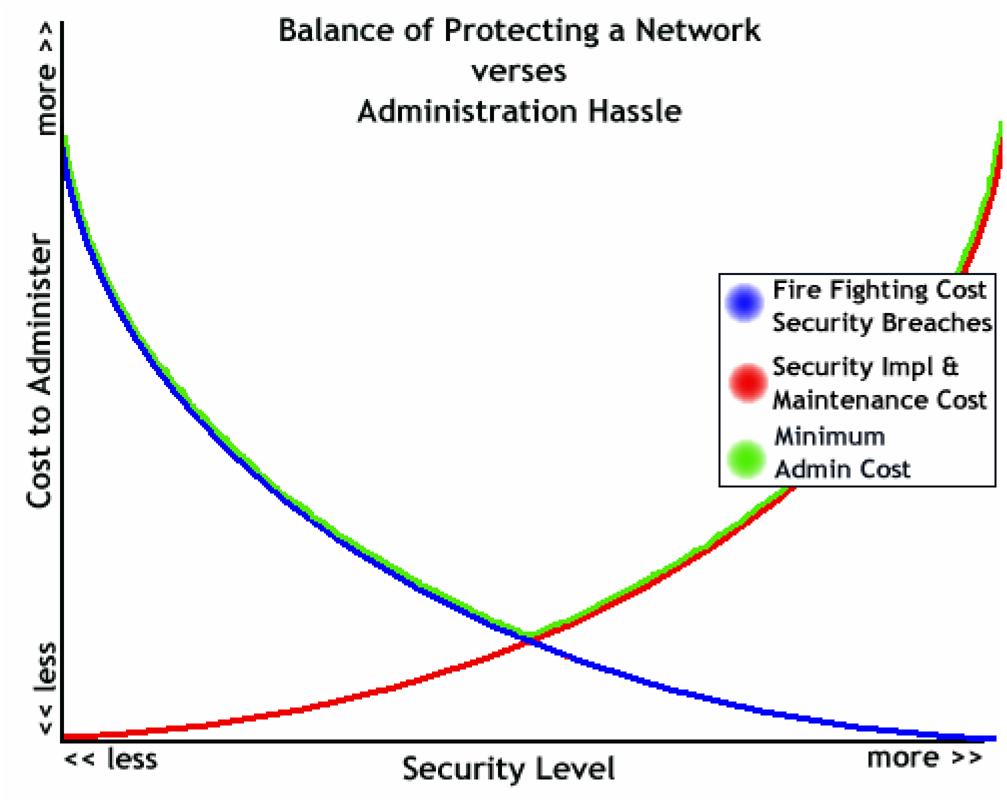

Figure 1. A representation of the security implementation for any corporate network. The more paranoid the security design (moving to the right on the X axis), the more cost in maintenance, hardware, software and manpower and a reduction in employee productivity verses a lazy/relaxed approach with higher costs in "fire fighting" issues as security issues arise and must be dealt with. With any company network, the balanced faced is usually a function of the tolerance a company has toward the perceived cost/pain of implementing security verse the cost/pain of not implementing security. In the case of no cost for security, an open network with no virus protection would

When a company spends the IT dollars to purchase, deploy and maintain an IDS, their network security can be considered leaning more to the paranoid end. The total extent to the deployment of IDF/IPS within the network will determine much as to the methods/possible attack vectors within a target network. The most common deployment is behind the firewall where it monitors traffic coming in and out of a network. A more paranoid, and costly approach is to examine the traffic on every switch and hub. This type of a deployment will not be required to monitor much more traffic flow since 80 percent of all network traffic is destined / from outside, but does face cost in switches that can be monitored and running lines to each switch / hub. Within this consideration, these exists some cost X, for a given size network Y to fully defend each switch, hub and node. Gaining insight into said company's cost X can be estimated from, size of the company, job postings, interviews and casual chat, as a "fellow IT person," with employees at bars. Knowing the extent of the IDS/IPD systems deployment can be valuable information and thus understanding and estimating a networks protection is critical.

> *To be certain to take what you attack is to attack a place the enemy does not protect.* Sun Tzu, The Art of War

To defeat an IDS/IPS system, an attack must attempt to avoid detection by those who administer of the system. This can be accomplished by avoiding detection or denying an IDS/IPS as a reliable consideration/factor. It should be carefully noted that while an IDS/IPS system might monitor and attack, it is those who administer the system who must be alerted and understand.

Prevent a Rate Based Detection

One of the principle means of intrusion detection is watching for out-of-pattern traffic on a network. These "finger prints" of viral activity are key indicators and should be understood so as to avoid such patterns.

- Traffic Mimicking:

  If it is possible to monitor the traffic on a network, then it would be possible for an attacking agent to mirror its data flow to look like normal traffic patterns. This would allow the flow to fall beneath the threshold of detection. This would require an attacking agent to be equipped with a pack sniffer and monitoring not just data rates, but also port and destination routing, building a map and then conforming to said maps patterns. About one year ago, the first virus was released in the world that had a packet sniffer and signaled the advancement of multi-mode attack capabilities. This type of attack would be more successful in a hub based network while a switched network would foil such attempts.

- Time and Rate Based Propagation:

  If an attacking agent is incapable of monitoring the network traffic for mimicking, because of complexity added to the executable or the network type (switched) does not allow, time and rate of propagation must be considered. With basic network pattern knowledge, a few rules can be devices to help a system fall within the normal traffic patterns for a network.
  Considering a corporate network:
  - Most activity will occur during business hours.
  - HTML activity will be highest during lunch hours.
  - 80/20 rule applicable to data flow on the network where 80 percent of all network traffic is flowing from or to the outside.
  - Email traffic occurs during all business hours, most heavily in the morning.

Given these simple definitions, limiting the number of attacks and timing them to fall within higher data rate periods, increases chances of success.

- Intelligent/Controlled Infection Paths

  Most worm like viral agents, once infecting a machine, have no insight as to what machines are already infected or have had attempts run against them. As a result, each new infection will create massive traffic as it wastefully attempts to attack machines already hit multiple times. This generates extensive traffic and can be a quick indication of an infecting worm. This can be minimized by utilizing a parent child relation where the parent assigns to the child, a range of address to attack.

Prevent a Deep Packet Inspection

Deep packet scanning is the ability to scan across multiple data packets for tell-tail signatures of viruses and malware. Thus it is an effective means for examining data that might span multiple packets.

- Hidden or Unknown Signatures

  Deep packet scanning is easily by-passable because of the nature of the checking algorithms. Current virus signature checking mechanisms rely on finding fixed patterns of data that uniquely identify a virus or malware. This can be fool by altering the bit pattern in a sufficient manner while still retaining the functionality desired. Simplistic means of doing this are using a polymorph engine attached to the virus, altering the code structure and recompiling, or hand editing the machine language/assembler, moving sections around and linking logically the execution path with jump sequences.

- Breaking Signature Code Across Multiple Files

  If the signature of a viral system can be distributed across multiple paths, detection can become more difficult if not impossible. The purpose of this code is to reassemble an obscuficated virus that can bypass deep packet inspection systems; the code is part of the initial infection vector. It is small, and its functionality can be rewritten in a multitude of ways. This allows for a new attack to be generated quickly with simplicity of design and functionality. The executable will drop in as a root kit, remaining undetectable. It will then download, via http like means, 5 .jpg files. It will then reassemble a worm whose code image / binary as been spread across the jpg files.

  Code snippet example for reassembling viral code from multiple file parts. First an agent downloads five ".jpg" files with valid JPG headers; the data internal is the malicous code that will be reassembled. These look like normal html image requests. The application then takes the five files and runs the following…

```
image = malloc(size of  data* 5);  // Buffer to hold image
for ( i = 0; i < size of  data; i++) {
        image[i] = file1 jpg [i + jpg_header_offset];
```

```
                image[i+1] = file2jpg[i + jpg_header_offset];
                image[i+2] = file3 jpg [i + jpg_header_offset];
                image[i+3] = file4 jpg [i + jpg_header_offset];
                image[i+4] = file5 jpg [i + jpg_header_offset];
        }
```

This code snippet can be hand crafted and reworked in so many different variations that the detection of the signature for this Virus Reassembly code will be impossible.

Denial of Service/Denial of Confidence
A system is only as good as it is trusted.

- Disable the IDS / IPS:
  If possible to identify the IP of the IDS/IPS, run a DOS on the IDS or generate such a large amount of valid appearing traffic that the IDS has trouble parsing and monitoring it all.

- Raise the Noise Floor
  Another option is to generate a massive number alerts and threat responses on unrelated ports and from unrelated computer systems. The idea is to throw the IDS/IPS system over the edge in its monitoring and triggers, thus allowing a normal attack to proceed unmolested. While alarms will be generated, reaction time for the IT staff, especially during late hours, will be slow and a quick attack should be successful.

- Erode System Confidence:
  Generate, over time, a number of alarms that appear to be malfunctions, generating user distrust in the capabilities of the monitoring system. If a foot hold can be gained inside of a system, more so in an unmonitored switched network, a single compromised machine could infect other local machines with a payload that generates massive traffic and then erases itself. This would trigger alarms, the administrative staff would investigate the problem, only to find nothing wrong. Continue to do this over the course of a week, during random times in the night hours. Then, on a Saturday night, a successful attack could be carried out; alarms will be triggered but ignored.

When penetrating into any system, a set of goals, a strategy should be outlined. Special attention should be paid for any network that does employ an IDS / IPS.
Goals:

1. Scout Target
2. Find a Single Point of Penetration
3. Spread and Infect
4. Execute Mission
5. Terminate Traces (optional)

1. Scout Target
Gain as much insight into your target as possible. View online job postings and cached histories looking at listed technologies required for experience, track users/employee postings via email addresses on bulletin boards and Usenet groups. Use information in gathered names to further research personnel locations, homes (attacking their minimally secured wireless networks), gather an email list, find local hangout points of employees during lunch or after hours, seek out coffee shops frequented with open wifi, find unattended laptops, purchase spam email listings and sort emails addresses for target company. Seek a job interview using false name used to gather information from interviewers about network structure, security policies and even tours of facilities. There are extensive means of gathering information about a company's finances and network from information externally exposed.

2. Single Point of Penetration
The next goal of any attack is to gain a "toe hold" into the target system without being detected. These means can vary widely; some of the more common means are…

a. Unsecured or rogue access wifi points
b. Targeting a specific user (laptop coming in and out of the network)
c. Email link embedding or attachments
d. Port 80 exploits (in or out)
e. Human Engineering

Knowledge gained in Goal 1 will help decide and appropriate attack strategy. Several points of attack should be planned, judged by metrics of percentage chances of success verses percentage chance of being noticed. This will allow for an ordered implementation for an increasing level of attack while staying quiet.

3. Spread and Infect
Once a "toe hold" can be established into a system, the propagation code must first study its current environment so as to mask its spreading in the network. A slow, rated based attack with an assigned vector list per infection point, prevent attempting to compromise already infected machines and move traffic slowly on known, used ports. An open wifi spot provides an excellent source of base traffic analysis, viewing ports, source and destinations, rates/volume and time.
For any volume N on a given port set X, there exists some Y volume that will fall within the noise of the target environment. Further enhances to the maximize a Y would be maximizing each time the network is utilized so as not to attack machines that have been targeted already and spreading during peak traffic times.
Or
Repeating the method used successfully to penetrate and infect a single target, if applicable.

4. Execution of Mission

a. DOS from within. This is a simplistic attack that blasts internal systems, denying effective usage. It tends to be effective but quickly seen.

b. Destruction of internal data. This can be very difficult to stop since it can happen so rapidly. A good backup, off site plan will prevent, but is still costly to recover from.
c. Gather sensitive data, mask it and push out again uses rate based methods.

5. Terminate Traces (optional)
If needed, cover tracks. Over write code on file multiple times. Leave hidden root kits for later exploit. Destroy all files, format drives, overwrite BIOS, render machines worthless.

The following scenario is designed to be an example of one possible case of a successful penetration into a corporate network. Given the task is to defeat a system employing an IDS, the following scenario plays out.

The purpose: break into a company's network, collect information, get it out of the network and then create havoc.

Target is a company with approximately 100 employees that was acquired by a larger a year ago. The smaller company still retains its own IT staff and thus it is safe to assume that the networks are connected but separate. Thus the goal is to gather data from smaller company's network; parent company's data might be of value on the open market. The target will be documents of interest, Word, Accel, .cpp, .h, dB container files, etc.

Job postings for the company listed a required skill for a certain IPS appliance, but the deployment and configuration is unknown. In actuality, the company utilizes a single IPS system that sits behind the firewall monitoring data coming in and out of the network.

Searching online for all strings of "@targeted-company.com" nets a list of email addresses and references full user defined names john.doe@targeted-company.com and jane.smith@targeted-company.com. Referencing the names and using whitepages.com results in addresses and phone numbers. Potential to drive by each address looking for an 802.11b or unencrypted network. Continuing the search on all online forums and other posting by possible employees where they use their work address.

Given the extent of data collected, an email based attack is considered best. The first attempt is to email jane.smith@targeted-company.com with an imbedded html iframe, taking advantage of an internet explorer weakness, to download a small executable of a root kit virus reassembler. Wait 24 hours and monitor the web site for hits; negative results thus try the next vector.

Send an email with an attachment with the virus reassembler embedded. The email is sent to john.doe@targeted-company.com with a subject "John, read this, Jen" with Jen being listed in the phone book as residing at the same residence. Four hours later, a hit as the web server is pinged, signaling the root kit installed successfully.
The next work day, noon time cycle, the root kit detects web/http usage; it request an html page from remote site, downloads it and download the five jpg files pointed to within the HTML. The jpg files are parsed, the virus is reassembled and the jpg files, only saved in memory, are freed, removing any possibility of inspection. In the web

server, the jpg files are removed. The virus is executed and hidden from detection by the root kit; the virus stays dormant, sniffing the network.

The network is on a 100Mbit hub with fourteen other machines. During peak weekday usage, the virus selects the first machine and infects it. The child virus hides itself utilizing root kit methods waiting to execute its payload. The next weekday, the child infected machine shares an empty file called "tempp" and the parent virus machine drops a random text file into the directory. The shared directory is unshared and deleted; now the parent virus knows the child infection was successful.

The parent machine now infects the other machines over the next three days, first doing two, then five, and then the remainder. Each infected machine monitors the network for local IP connections, shared drives and builds a list of possible attack vectors. The parent machine shares a single directory and each child writes its own file listing the possible targets. The parent scans each file and gives each child a target so that no target it hit more then once. Before the all out attack, the system has been collecting passwords, sniffing and grabbing sql data, all .doc, .xls, .c, .ccp, .h and other files of interest. It zips them all up and then each one emails it out at staggered times to different addresses. The next day, at 3am, the infected machines begin an all out infection attack on the network. Each newly infected machine zips up files of interest and emails them out. By 6am the virus has propagated out to the larger corporate network and then each infection utilizes FTP to upload data on public servers. At 9am, each machine begins mass deletion plus a double overwrite of files, all server linked/shared files and then destroying all files of interest, then system files and finally overwriting the computer's BIOS.

The Aftermath

- Virus checking systems failed since the initial vector was a new pattern that escaped detection. It then was able to smuggle a worm into infected machines by masking the data and traffic as normal in appearance, embedded in jpg files.
- The virus was able to propagate behind the network unimpeded since internal traffic was not monitored, only that traffic propagating in and out. The infections remained undetected because it moved slowly, used hardly any resources on the systems and installed as root kits, thus able to hide itself from all detection.
- Communication of sensitive data to the outside was masked by normal activates, using email to move the data out of the system.
- The final, wide spread, infection push was able to succeed because of the investment choice made by the company not to monitor every connection point.
- The large FTP push at the end did not succeed as well as planned. Too many machines were infected and pipes to the outside world were clogged. The IDS/IPS system noted the mass scale traffic and was able to alert administrators who turned off all outside traffic.
- The destruction of the data and machines happened to quickly for the administrative staff to stop and infection "foot prints" were covered.
- With little information and few detectable patterns in logs, the IT staff assumed the virus penetrated that night or the prior evening but they could not find the means.
- The initial vector/parent infected machine did not self destruct; the root kit / virus / reassembler removed all traces of infection save for itself and powered the machine

down at 4am. It was considered safe since it was not on during the night of the infection.
- One month later, the root kit / virus / reassembler will wake, download an HTML page and then download the embedded jpg files…

An Intrusion Detection/Prevention System is designed to be yet another layer in a multi-layered defense for a corporate network. It will not be able to stop a specially tailored attack on a network but is highly effective in stopping the more generic attacks as seen in the Saser and other worms.

References:

Dumbill, Edd, "The Next 50 Years of Computer Security: An Interview with Alan Cox" O'Reilly Network, accessed September 20, 2005 http://www.oreillynet.com/pub/a/network/2005/09/12/alan-cox.html

"Beyond IDS: Essentials of Network Intrusion Prevention" Top Layer Networks, Inc. 2002

Minasi, Mark "Follow-Up: Why Microsoft Can't Stop Root Kits" accessed February 29, 2005 http://www.windowsitpro.com/Article/ArticleID/45518/45518.html?Ad=1

Oleg Kolesnikov, and Wenke Lee "Advanced Polymorphic Worms: Evading IDS by Blending in with Normal Traffic" Georgia Institute of Technology http://www.cc.gatech.edu/~ok/w/ok_pw.pdf

President's Information Technology Advisory Committee, "Cyber Security: A Crisis of Prioritization" accessed February 29, 2005 http://www.nitrd.gov/pitac/reports/20050301_cybersecurity/cybersecurity.pdf

Raja, Sanjay "Network Intrusion Prevention Systems - Why "Always On" Stateful Inspection and Deep Packet Analysis are Essential to Deliver Non-Stop Protection" Top Layer Networks, Inc. January 16, 2005

Schuman, Evan "When Safe Devices Become Smart and Dangerous" Ziff Davis Internet, accessed October 6, 2005 http://www.eweek.com/article2/0,1895,1863742,00.asp