



Current Issues in Network Security

© 2008 James Shewmaker
and SANS Institute

Setting the Stage

- Pendulum of Computing is swinging away from the desktop
 - Device Mobilization
 - Distributed Data Storage
 - Relationships are the new Crown Jewels
- Any point defense can be defeated
 - Defense-in-Depth catches most
 - Must be prepared to Handle an Incident

Copyright 2008 James Shewmaker and SANS Institute

2

This presentation is designed to provide a current look at what we are dealing with in Network Security.

With computing in general decentralizing (mobile PCs, PDAs, phones, etc.) it is getting difficult to protect everything from any possible attack. Sometimes, it isn't even the data we are worried about; it's the relationships of the data (not just my email address, but the folks I send email to might be interesting to an attacker).

Sometimes security is left as an afterthought, and often thought of like "We're a specialized operation; nobody would want to attack us." Unfortunately, just the fact that you exist is enough to be a target. Attacks are often scripted so they don't know what they are attacking until they have broken in and take a look around. On the other hand, if you are specifically targeted for political or business reasons, it often involves more damage. Either way, when the attacks happen you need to be able to handle the situation.

We will start by establishing our framework for Incident Handling so we can deal with these attacks, then we will cover a few of the most current and prolific attacks or tools that you will encounter.

Why is Incident Handling Important?

- Sooner or later an incident is going to occur
 - Do you know what to do?
- It is not a matter of “if” but “when”
- Planning is everything
- Similar to backups
 - Hopefully you never NEED them, but if a major problem occurs, you are going to be glad that you have them available.

Copyright 2008 James Shewmaker and SANS Institute

3

It does not matter how big your company is or what type of business you are in; sooner or later you are going to have an incident. Companies of all sizes and types have had incidents. In some extreme cases, those that were not prepared and did not handle it correctly are no longer around to talk about it. When it comes to having to deal with an incident, it is not a matter of IF an incident is going to occur but WHEN is it going to occur. Unfortunately, some companies choose to deal with an incident by ignoring it. However, as you can imagine, this is very risky to do. I bring this up because some companies I talk to say, “I have never had an incident in 2 years so why do I have to worry about it?” In this case, the truth of the matter is, they probably have had several incidents. Yet, since they failed to detect them, these organizations took a stance of ignoring each incident. As we stated, this practice is very dangerous and it is only a matter of time until this catches up with you.

One of the main reasons for a module on incident handling is this central idea: planning is everything. If you are prepared and know what to do, dealing with an incident can be fairly straightforward. On the other hand, if it catches you off-guard, there can be many sleepless nights.

Do not get discouraged if you do all of this planning and do not use it right away. Do not say, “I have done this planning and have not had an incident in 3 months.” Think of it like your system backups. You might not need to use a backup every day. However, if a problem ever does occur (and it will), you will be glad that you had prepared.

Incident Definition

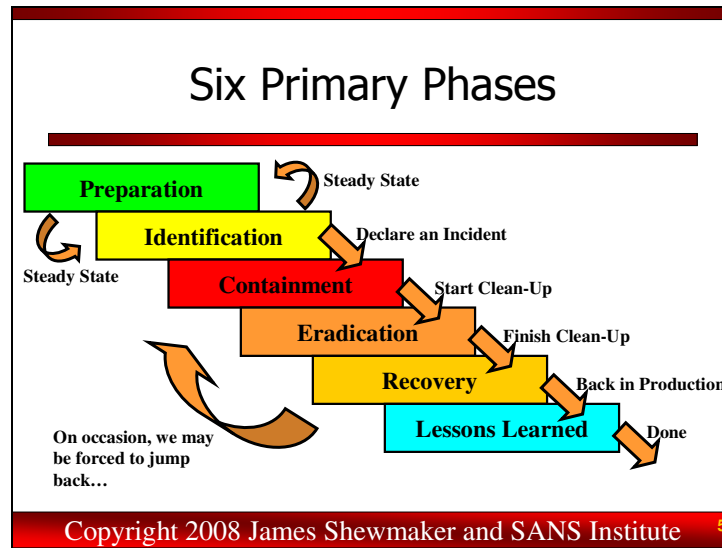
- The term "incident" refers to an adverse event in an information system and/or network...
- ...or the threat of the occurrence of such an event
- Focus is on detecting deviations from the normal state of the network and systems
- Examples of incidents include:
 - Unauthorized use of another user's account
 - Unauthorized use of system privileges
 - Execution of malicious code that destroys data
- *Incident implies harm, or the attempt to harm*

Copyright 2008 James Shewmaker and SANS Institute ⁴

What is normal? Keep in mind bad things happen in the best of circumstances and we may have to deal with legal issues

This slide and the next one are for the purpose of defining what we mean when we use a word like "incident" or "event." Incident, as we are using it, refers to actions that result in harm or the significant threat of harm to your computer systems or data. Looking for incidents involves finding deviations from the normal state of the network and systems. There are several important points for an incident handler that flow from this definition. First, because we are dealing with harm or potential harm, our task is to limit the damage. We want to be careful to choose courses of action that do not cause further harm.

Secondly, your organization may well have a right to redress. We cover this in depth later, but there are criminal and civil law remedies associated with computer incidents. In either case, the incident handler should proceed in a manner that does not preclude use of the evidence gathered in a court setting. A handler does not know in advance whether a given case will go to court. Although only a small fraction of most cases end up in court, you need to treat all of them from the outset as though they may go to court. Don't worry; that's not an enormous burden. It just means doing your job thoroughly and documenting your actions carefully.



The six steps in incident handling are preparation, identification, containment, eradication, recovery, and lessons learned. The steps serve the handler as a compass or a roadmap, a way to keep in mind what they are trying to do and the things they need to do next.

The steady-state, day-to-day practices of most incident handlers are the first two steps: preparation and identification. We spend a lot of our time getting ready to fight the next battle, and looking for events that could be signs of trouble.

Once we've identified an incident (that is, events that indicate harm or the attempt to do harm), we move into containment. Then, the general flow is down the page. You move from containment to eradication to recovery to lessons learned. Don't skip steps! Also, I caution you. Please try to complete an entire given step in the containment and later phases before moving to the next phase for a single incident. In other words, for one incident, don't contain it partially on a few systems, and then move to eradication on those machines while containment on other systems begins. Do all of containment first, then move to eradication, and so on. You will likely get organizational push-back on such an approach, but it is really the best way to go to successfully handle incidents.

Also, while the general flow of this process is down the page, sometimes you have to jump back up given changing circumstance. You might be in the midst of the recovery phase, when your attacker or malicious code sneaks back in. You've got to be flexible enough to jump back and redo the containment phase, then eradication, and then return to recovery.

Preparation Overview

- The goal of the preparation phase is to get our team ready to handle incidents
 - Policy
 - People
 - Data
 - Software/Hardware
 - Communications
 - Supplies
 - Transportation
 - Space
 - Power and Environmental Controls
 - Documentation

Copyright 2008 James Shewmaker and SANS Institute

6

The goal of the preparation phase is to get our team ready to handle incidents. This slide serves as an overview of the elements needed to prepare the team for an incident; these are actually the fundamentals of contingency planning and it is advisable to have these basics covered. As we move through the preparation section we will discuss these items further.

Identification Phase

- The goal of the identification phase is to gather events, analyze them, and determine whether we have an incident
 - Look for harm, the attempt to harm, and deviations from the norm
- Sometimes we identify the wrong event
 - Must be verifiable
- Public buildings are equipped with fire alarms
 - Who is authorized to pull the handle?
 - Who is authorized to decide if it is safe to re-enter the building?

Copyright 2008 James Shewmaker and SANS Institute

7

Know what normal is and do not rely on special tools, know your equipment and how things should behave.

The goal of the identification phase is to gather events, analyze them, and determine whether we have an incident. In essence, we are looking for harm (or an attempt to harm), as well as deviations from normal operations.

Bad things™ can happen when an unqualified, unauthorized person makes the call on an incident and is wrong. Fifty thousand dollars later, after three days off-line, you question the said individual, “What were you thinking?” The answer is usually the same, “I, uh, thought it was nothing.”

After a fire alarm is pulled, qualified fire-fighters who know the signs to look for come to the site and investigate. Only then, does the person in charge at the scene authorize re-entry into the building. This should be the paradigm we work under; be willing to alert early, have trained people give the situation a look and be able to stand down if nothing is wrong at a minimum of expense, bother and wasted time.

Containment

- The goal of the containment phase is to stop the bleeding
 - Prevent the threat or harm, at least from getting worse
- We'll discuss:
 - The sub-phases of Containment
 - Methods for short-term containment
 - System Back-Up
 - Methods for long-term containment

For containment, we want to stop the bleeding. How can we arrest the attacker in his/her tracks before he/she causes more damage? Sometimes we just want to slow them down significantly while we put our real defense in place. This is also the phase where we make any backups to preserve evidence.

Containment – Personal Networking

- For external attacks, coordinate closely with your Internet Service Provider
 - They might help you in identification, containment and recovery
 - Especially for large packet floods, bot-nets, worms, and virulent spam
 - Your information may save someone else a lot of pain
 - We need to work together as a community to foil widespread attacks
- You might need someone's ISP to take down a bad system
- Local and regional Law enforcement contacts are good
- Don't forget business partners
 - Customers, EDI connections, etc.

Copyright 2008 James Shewmaker and SANS Institute

9

Find out who your ISP security contact is before an incident occurs. Find out who your key business relationship's contacts are as well.

Before the large effective denial of service attacks, many organizations felt they could handle all incidents internally. Now, most organizations realize they will need to coordinate with their ISPs to bring some incidents under control, especially in large packet floods, wide-scale botnets and their communications channel, worms, and virulent spam. Many ISPs are quite experienced at network-based exploit and denial of service types of incidents, and are very efficient at handling them. ISPs often keep system and even network logs, at least for a reasonable time frame. Also, they may be able to spare other folks that get their service from the trouble that you had to go through.

If you don't know any Law Enforcement type folks you would work with during an incident, figure it out! If you don't know where to start, call your local law enforcement and figure out who to call when. Google for Infraguard and meet up with the local chapter folks before you need them.

By the way, there is one other group that you should be aware of that is extremely experienced in handling incidents; these are the computer and network staff of colleges and universities. It can be a good idea to make contact with the computing staff at your local educational institution. If you build a relationship with them, they can really help you when you are under fire.

Also stop to consider any Business to Business applications and who they affect. Maybe your attack is coming from a supplier or a branch office?

Backup – It Sounded So Easy

- Making a backup under fire is a lot harder than one might expect
- Practice!
 - dd for Unix/Linux and Windows (my favorite)
 - Ghost (the latest versions – default is not bit-by-bit... you have to configure it)
 - Safeback - commercial, and designed for evidence collection
 - Available at <http://www.forensics-intl.com/safeback.html>
 - Drive Duplicator hardware & write blockers

Copyright 2008 James Shewmaker and SANS Institute ¹⁰

I know it sounds like a lot of money, but a 1 TB drive for <\$200 is common.

Pick your favorite tool (mine is dd), and then practice with it. There are many enhancements built onto dd, try dcfldd (<http://dcfldd.sourceforge.net/>) or a new set of patches that keep the dd version “truer” named dc3dd (<http://dc3dd.sourceforge.net/>). One of the nicest things about the enhancements is providing a hash of the input as it is read. This hash is treated as a signature of the data so you can verify it has not changed. The dd family of tools can be used to copy to/from any device/file/filehandle. This is helpful if you want to collect evidence from a drive and are interested in collected fragments of deleted files, partitioning information, or otherwise hidden info. If you do not frequently use dd, you will want to practice before using it. I use it all the time for backing up systems even outside of incident handling.

Backup – Collecting ESI

- Get a backup of the RAM if possible
 - Precedence set in some torrent related cases in 2007
 - Not as difficult as it sounds
 - Some malicious software never touches the hard drive; it's your only chance to find out what happened
- dd to the rescue

```
dd.exe if=\\.\PhysicalMemory of=\\filesERVER\data\ram.img  
dd if=/dev/mem | nc 10.10.10.10 1010
```

Copyright 2008 James Shewmaker and SANS Institute 11

Sometimes it really pays to collect a copy of what is in the system memory. RAM changes constantly, so if you are slow to acquire the evidence here it may be lost. Some attack tools only exist in memory—they never are written to the victim's hard drive. The Federal Rules for Evidence now define Electronic Stored Information (ESI) as including contents of RAM, so it may be required for evidence if it is the best evidence (logs are missing or manipulated). If you are not running a pirated content site you might not need to worry, but it's a great thing to have anyway.

In this slide we have two example syntaxes, one for Windows and one for Linux/BSD/UNIX. The first line we copy from the special alias devicename for RAM and output it to a fileserver's data share over windows networking. The second example is on Linux and this time we copy the RAM (using the netcat tool) to a machine with an IP address of 10.10.10.10 that collects evidence while listening on port 1010. I use these commands all the time to copy the evidence off to a safe place before taking the system offline to collect the evidence.

Once I've collected as much live evidence as I can, I will usually pull the power to the system (if you close the system down gracefully, you might trigger something the attacker left to clean up or simply disrupt the file access times so much you loose timeline evidence). Then I can make an offline backup and preserve the media evidence for later forensic analysis.

Sometimes you can't take the machine offline, but I do when I can. When I can't, the RAM evidence is even more important, and I make a live backup of the media on the system to my safe archive.

Eradication – Removing Malicious Software

- Remove the cause of the incident
 - Virus infestations
 - Backdoors
 - RootKits
- If there is anything like a rootkit, you should rebuild from scratch
 - Format the drive
 - Operating System (and PATCHES!)
 - Applications (and PATCHES!)
 - Data (the hardest one... possibly tainted back-up)
- Encourage the impacted business unit to rebuild, reviewed by the computer security team (including incident handlers)

Copyright 2008 James Shewmaker and SANS Institute 12

Test your systems with trusted binaries from a CD.

Viruses and other malicious software (malware) are very interesting. They can be easy to deal with after the anti-virus companies have analyzed them. You just let the software clean up the problem and it does a great job. Dealing with viruses that don't yet have anti-virus signatures is a much harder problem.

If the attackers change the operating system itself, such as installing a RootKit, you should rebuild from the original install media, and install patches while the machine is NOT connected to a network. Make sure you patch the system thoroughly, or the attacker will return. I usually burn service packs and quickfixes to a CD and install as much as I can before plugging it into any network. What if the attack happens while you are waiting for Windows Update to download and install 102 updates? (I saw this happen in February 2008 on a Windows XP Pro Service Pack 2 updating to the current patches). It's also more than just taking too long, how many of those patches are fixes for remote attacks? Even then, sometimes you have to re-apply the patches!

Also, encourage the impacted business unit to do the rebuild, under your supervision. That way, you can make sure they understand the build process. Furthermore, they can verify that the system is functioning properly. And, finally, you can verify that all patches are installed when they rebuild the machine. The last thing you need is the clean machine to "never worked right since security got involved." I've found that we will get blamed for the sky being blue as well, no need to provide fuel to the fire—get them to build it, ok it for them, then have them sign off that is working "normally." The media backups tend to serve a second purpose here as well. I've had to put the old infected system online and show them their new bug was in fact one that was there all along but they failed to recognize/remember it.

Recovery – Restore Operations

- Decide when to restore operations
 - I usually try for an off-hours timeslot
 - It's easier to monitor carefully then
 - Put the final decision in the hands of the system owners
 - Provide your advice, but they make the final call
 - Document your advice in a signed memo

Copyright 2008 James Shewmaker and SANS Institute 13

You are usually forced to do restore after hours. Oh, and be back early the next morning.

The decision of when to put the system back into business has to be made by the system owner. As a handler you can give them advice and try to be helpful, but this is their call. They are the ones that depend on this system.

Document your advice in a signed memo to the system owner. Remember, you may not get your way in this decision, but at least you've documented your recommendations. The key to recovery is to monitor the systems in production—this means not turning things on and catching up on sleep, it's turning them on and being sure they work normally.

Lessons Learned – Permanent Fixes

- Based on what you've learned, get approval and funding to fix:
 - Your processes
 - Your technology
 - Improved incident handling capabilities

We can always learn.

We want to have constant improvement, so the cause of the incident can be eliminated or minimized. You must go to management and make a compelling case for fixing the problem that caused the incident in the first place. This may mean an alteration to the processes or technology in your environment. Sometimes we can't learn much from an incident we haven't dealt with properly, but this is the chance to salvage as much value out of the cost of the incident you can. Maybe the business was down for two days and it was lost revenue. At least figure out how to minimize it next time if you can't outright prevent it.

General Trends – The Underground Community

- So, what are we seeing in the wild?
- Attack tools are getting easier to use and more easily distributed
- High-quality, extremely functional attack tools
 - Better quality than from some major software houses
- The rise of the anti-disclosure movement
 - Script kiddies are abusing tools
 - Vendors don't want vulnerabilities to be publicly released
 - Some groups are no longer releasing exploits publicly
 - The pendulum swings to and fro
 - Significant implications on disclosure with respect to the DMCA
- Here are some attacks you should be able to handle

Copyright 2008 James Shewmaker and SANS Institute 15

With the rise of various groups writing and releasing computer attack tools, a lot more information about security vulnerabilities is available to the general public. The less-informed attackers (often called “script kiddies” or “ankle biters”) will use this information in attacks. We must also use this information to defend ourselves. I’ve included several references at the end of the handouts to help you stay informed.

Recently, hacktivism has often been cited as a reason by attackers for the mayhem they cause. You see, they are merely trying to improve society by attacking computers and cyber civil disobedience, or so goes the claim. While some hacktivism appears legitimate (such as breaking down barriers between countries), other instances are purely an excuse.

Additionally, we see major debates on whether information about security vulnerabilities should be widely disclosed in public (full disclosure) or should be hidden until adequate defenses are released (anti-disclosure). There are significant legal issues here, including copyright protection and prohibitions against reverse engineering copy protection schemes manifested in the Digital Millennium Copyrights Act (DMCA).

Reconnaissance

- Reconnaissance is "casing the joint"
- Two general types of attackers:
 - Script kiddies – look for low-hanging fruit, and may skip this step
 - Attackers out to get a particular site – this step is extremely important
- Very helpful step for experienced attackers
- Recon includes system and port scans, but what about "Open Source Information?"

Copyright 2008 James Shewmaker and SANS Institute 16

Detailed reconnaissance really helps an attacker get a feel for your network before ever firing a packet in anger. The Internet itself is a treasure trove of information for a curious attacker.

To begin an attack, your adversary will gather as much information as possible from open sources. Think about attacks in the plain-old real world for a minute (I know it's hard to think about non-virtual things....but occasionally we must). Before bandits rob a bank, they will visit the particular branch, look at the times that the security guards enter and leave, and observe the location of security cameras. Additionally, they may even use white pages to find the address of the bank and a map of the city to plan their get-away path. This is the same first step in cyber-attacks.

Web Site Searches

- Search the target's own web sites
 - Press releases
 - White papers
 - Design documents
 - Sample deliverables
 - Open positions
 - Key people
 - Contacts
- Search related sites
 - Business partners, ISP, suppliers, portfolio customers

**Especially useful
for attackers!**

Copyright 2008 James Shewmaker and SANS Institute 17

This is where it gets fun ... maybe even vulnerability assessment reports? Cellphone records, addresses of Quest executives?

Corporate web sites often contain contact information with phone numbers, which are useful for war dialling and social engineering. Some sites even include a description of their computing platforms and/or architecture. Attackers grab a copy of your entire web site to look for special information about your organization.

Search engines are also quite useful. By searching for information about a target, an attacker can often learn about their platforms and architecture through UseNet postings of employees. Also, web sites will indicate business partners and other potential links useful in spoofing and other attacks. Advanced search engines (such as AltaVista) include the ability to search for sites linking to the target. Simply search on "link:www.[target_company].com" for all sites that link to the target.

Web Site Searches Defenses

- Preparation:
 - Limit and control information
 - Know what information a company is giving away and perform risk analysis
 - Make employment ads more general - If HR will let you! Good Luck!
 - Limit information on a website
 - Determine what other sites are linked to your company
- Identification:
 - Look for web spider/crawler activity
 - Logs show systematic access of entire website, page by page
 - That could also simply be the Google bot or other search engine
 - Someone just sucked down the entire contents of our site
- Cont, Erad, Recov: N/A

Copyright 2008 James Shewmaker and SANS Institute 18

Know your network, know what you are protecting. Do searches of your external web site every once in a while. Sometimes you find private XLS or DOC files that weren't supposed to be in the public view.

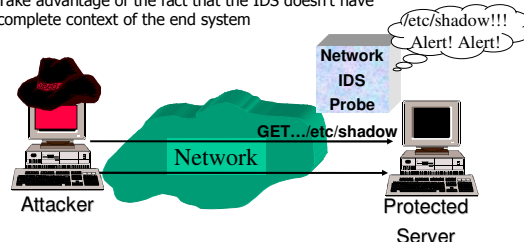
You should periodically check various open sources of information to see what your company is leaking. This analysis can be done by the security organization, legal department, and public relations, as all have a vested stake in protecting your corporate information.

For identification, have your web administrators look through their logs for an indication that someone has used a web spider (also known as a web crawler) to access every page on your site in a short period of time (say within 5 minutes). Most likely, this activity is just the crawler of a search engine (like the Google-bot). But, from another source, it could be a sign of pre-attack recon.

I recently was using Google to search for VMware escape vulnerabilities and stumbled across another SANS Instructor's paper on some very cutting edge material. Unfortunately this information was supposed to be private but somehow Google found it. Protect yourself from this sort of thing by getting creative with looking for items on and around (or even on another site that talks about yours). Once you know it is there, you can take some steps to remove the material.

IDS Signature Matching

- Recall how IDS signature matching works...
 - Look for anything matching the string "/etc/shadow" and warn me
 - I don't want anyone grabbing a list of user accounts
 - How can we slip things by the network IDS?
 - Take advantage of the fact that the IDS doesn't have complete context of the end system



Copyright 2008 James Shewmaker and SANS Institute 19

Intrusion Detection Systems listen in on networks looking for signatures of known attacks. Similar to virus detection software, they perform pattern matching. They look for a certain pattern and if they find it, they set off an alarm.

Many networks are configured with packet filtering to protect a given server or a whole network. They are set up to allow some packets in, such as those destined for a web server (port 80). And, they are configured to deny other packets, such as those destined for a telnet server (port 23).

These fragmentation attacks exploit the fact that a network-based IDS and packet filters do not have the complete context of how a packet will be treated and reassembled at the end system. The network-based IDS must do virtual packet reassembly, and do it in the same manner as the protected hosts, which may use multiple different operating systems.

For example, let's look at how an IDS might do signature matching. Suppose I have a signature that looks for anything containing the text "/etc/shadow" and then warns me when it sees this pattern. I don't want anyone to grab my password file with its very valuable hashes that they might try to crack.

How can we slip things by this signature and network IDS?

We could take advantage of the fact that the IDS doesn't have complete context of the end system when it reassembles packet fragments.

Let's Create Some Interesting Frags

- IP allows packets to be broken down into fragments for more efficient transport across various media
- The TCP packet (and its header) are carried in the IP packet
- Many types of fragmentation attacks possible
- Two examples
 - Tiny fragment attack
 - Fragment Overlap attack

Copyright 2008 James Shewmaker and SANS Institute 20

To support different transmission media, IP allows for the breaking up of single large packets into smaller packets, called fragments. The higher-level protocol carried in IP (usually TCP or UDP) is split up among the various fragments.

We will use this fragmentation capability built into IP to bypass some network-based IDS systems. Also, older packet filtering devices do not handle packet fragments properly, allowing an attacker to avoid detection.

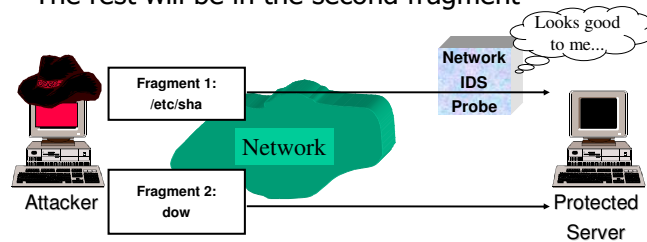
There is a wide range of fragmentation attacks that are possible. We will look at two examples:

Tiny fragment attack

Fragment Overlap attack

Tiny Fragment Attack

- Make a fragment small enough so that part of the offending string is in the first fragment
- The rest will be in the second fragment

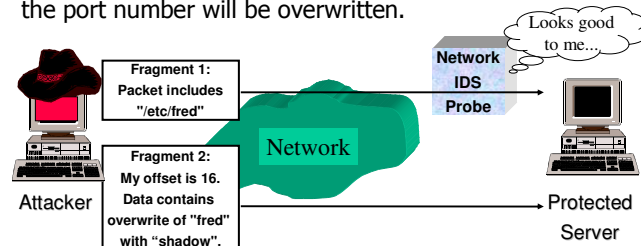


Copyright 2008 James Shewmaker and SANS Institute 21

The tiny fragment attack is designed to fool the IDS by creating an initial fragment that is very small. It's so small that no single fragment has everything necessary to match a signature. A very stupid IDS sensor may allow this type of attack to pass by unnoticed, because it doesn't reassemble the packets. Most IDS tools today are capable of detecting this type of attack. Still, a large number of tiny fragments will be a burden for the IDS, which has to reassemble the fragments for analysis.

Fragment Overlap Attack

- Fragment Overlap attack - In the second fragment, lie about the offset from the first fragment. When the packet is reconstructed at the protected server, the port number will be overwritten.



Copyright 2008 James Shewmaker and SANS Institute 22

A more insidious fragment attack is the Fragment Overlap attack. For this scenario, the attacker creates two fragments for each IP packet. One fragment has the TCP header, including a string that is not being looked for. The second fragment has an offset value that is a lie. The offset is too small, so that when the fragments are reassembled, the second fragment overwrites part of the first, particularly the part of the first fragment that includes some data that is sensitive.

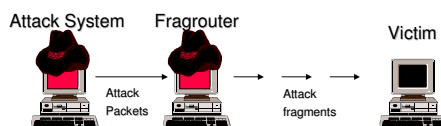
The IDS ignores the first fragment (after all, it doesn't match the signature). The device then might ignore the second fragment (after all, it's just a fragment of the previous packet that it allowed, which doesn't include a complete match for the signature).

When the two fragments arrive at the targeted protected server, they are reassembled. The reassembly overwrites the sensitive data.

But surely you are wondering why the virtual fragment reassembly buffer of the IDS cannot just reassemble the packet in the proper way... The problem with this is described on the next slide.

IP Fragment Attack Tools

- Note that Nmap can do scans with fragments (essentially the Tiny fragment attack)
- More importantly, fragrouter can be used to create nasty fragmentation attacks
 - 35 different fragmentation and related TCP obfuscation recipes
 - Written by Dug Song
 - <http://www.w00w00.org/files/sectools/fragrouter/>
 - With fragrouter, all packets entering one interface go out the other interface fragmented
 - The attacker can specify how fragmentation will occur
 - Helps bypass intrusion detection systems and get through some IPS tools



Copyright 2008 James Shewmaker and SANS Institute 23

The next question might be, “Well how can I create fragments?” A tool called fragrouter allows attackers to create nasty fragmentation attacks. It was written by Dug Song and is available from <http://www.w00w00.org/files/sectools/fragrouter/>. With fragrouter, all packets entering one interface go out the other interface fragmented. The attacker can specify how fragmentation will occur, choosing from several options.

Numerous fragment attack types are supported by fragrouter (over 35 varieties), including:

-F1: *frag-1* : Send data in ordered 8-byte IP fragments.

-F2: *frag-2* : Send data in ordered 24-byte IP fragments.

-F3: *frag-3* : Send data in ordered 8-byte IP fragments, with one fragment sent out of order.

-T1: *tcp-1* : Complete TCP handshake, send fake FIN and RST (with bad checksums) before sending data in ordered 1-byte segments. This option is highly useful in evading IDS and IPS systems.

-T5: *tcp-5* : Complete TCP handshake, send data in ordered 2-byte segments, preceding each segment with a 1-byte null data segment that overlaps the latter half of it. This amounts to the forward-overlapping 2-byte segment rewriting the null data back to the real attack.

-T7: *tcp-7* : Complete TCP handshake, send data in ordered 1-byte segments interleaved with 1-byte null segments for the same connection but with drastically different sequence numbers.

IP Fragmentation Attacks – Defense

- Preparation:
 - Reassemble packets before making filtering or intrusion detection decisions
 - A firewall can do this, imposing its impression of the reassembly before the IDS and end system get the packet
 - Keep your IDS and IPS up-to-date
 - Supply IDS and IPS with recommended resources (network performance, processor, RAM, and hard drive)
 - For sensitive systems, use host-based IDS in addition to network-based IDS and IPS
- Identification:
 - IDS signatures indicate heavy fragmentation or overlapping fragments
 - IPS can block overlapped fragments
- Cont, Erad, Rec: N/A

Copyright 2008 James Shewmaker and SANS Institute 24

To avoid these problems, make sure your systems reassemble packets before making filtering or intrusion detection decisions. A firewall can do this, imposing its impression of the reassembly before the IDS, IPS, and end system get the packet. Everything after the firewall will have the same interpretation of the packet, because the reassembly by the firewall forces the fragments into a single packet.

Also, make sure you follow carefully your vendor's specifications for the processing power, RAM, network, and other performance characteristics for your IDS sensors and IPS tools. Those analytic capabilities require resources to keep up with the attacker's tricks.

Furthermore, a host-based IDS has the local TCP/IP stack and the complete context of the end system. Therefore, it can avoid the problems associated with fragmentation attacks.

Of course, a minimal set of absolutely required ports should be open in the first place. This minimizes the chance that an attacker will find a sensitive port through scanning with or without fragments.

Web Scanners

- Automated program that scans sites looking for known, vulnerable material
 - Look for specific scripts' names that are known to have problems
 - For example: AWstats, phpBB, phf, showcode.asp, campas, aglimpse, etc.
- Some Web scanners are very basic
 - cgiscan: Several tools with this name
 - cgichk.c and ucgi.c
- Note that they are sometimes called Web/CGI scanners, but they also find vulnerable PHP, JSP, ASP, and other related technologies
- My favorites are nikto (command line) and wikto (gui)

Copyright 2008 James Shewmaker and SANS Institute 25

Keep in mind there are a lot of CGI pages created that have a lot of bad functions.

Admin

Access a printer

AWstats

phpBB??????

There are a lot of web scanners out there. Each looks for the presence of known, vulnerable scripts on a target web site. You can find a variety of such scanners at <http://www.packetstormsecurity.org>.

One of the best free web scanners today is Nikto.

Whisker/Nikto IDS Evasion Tactics

- A paper on techniques by Rain Forrest Puppy at <http://www.ussrback.com/docs/papers/IDS/whiskerids.html>
- Application-level IDS evasion
 - A very active area of research in the computer underground
- Consider what an IDS looks for in CGI exploits:
 - GET /cgi-bin/broken.cgi HTTP/1.0
- Whisker supports ten modes of IDS evasion
 - 9 at application level and 1 at transport level

Copyright 2008 James Shewmaker and SANS Institute 26

Fragrouter gave us the ability to evade some IDS by fragmenting packets at the IP layer. Nikto includes the IDS evasion capabilities originally built into the Whisker tool. Whisker (and Nikto) evade IDS at the application layer, by messing with the way it formats requests for CGI scripts.

A nice paper on these techniques by RFP is located at <http://www.ussrback.com/docs/papers/IDS/whiskerids.html>

Application-level IDS evasion is a very active area of research in the computer underground.

Consider what an IDS looks for in CGI exploits. Most IDS have a variety of signatures that look for something like:

GET /cgi-bin/broken.cgi HTTP/1.0

Whisker (and Nikto) morph this request so that it doesn't match any signatures. It supports ten modes of IDS evasion:

9 at the application level

1 at the transport level

Ways Whisker/Nikto Avoids IDS

- URL Encoding
 - GET /%63%67%69%2d%62%69%6e/broken.cgi HTTP/1.0
- ../ directory insertion
 - GET ../cgi-bin../broken.cgi HTTP/1.0
- Premature URL ending
 - GET /HTTP/1.0\r\n
 - HEADER: ../../cgi-bin/broken.cgi HTTP/1.0\r\n
- Long URL
 - GET /thisisabunchofjunktomaketheURLlonger../cgi-bin/broken.cgi HTTP/1.0

Copyright 2008 James Shewmaker and SANS Institute 27

Various IDS engines respond differently to each technique.

URL Encoding converts the HTTP request into a different representation, by changing ASCII characters into their hexadecimal values and pre-pending them with a “%” character.

```
GET /%63%67%69%2d%62%69%6e/broken.cgi HTTP/1.0
```

../ directory insertion includes directory navigation to the current working directory (known as “.”).

```
GET ../cgi-bin../broken.cgi HTTP/1.0
```

Premature URL ending stops URL and includes the reference to the cgi script in the header. It amazes me that this particular option works, but it does.

```
GET /HTTP/1.0\r\n
```

```
HEADER: ../../cgi-bin/broken.cgi HTTP/1.0\r\n
```

Long URL formatting just changes into a non-existent directory, and then to its parent. The idea here is that the web server itself will parse out this nonsense, but the IDS is likely just looking in the first 30 or so characters for signature matching.

```
GET /thisisabunchofjunktomaketheURLlonger../cgi-bin/broken.cgi HTTP/1.0
```

More Whisker/Nikto IDS Evasion Modes

- Fake Parameter
 - GET /index.htm?param=../cgi-bin/broken.cgi HTTP/1.0
- TAB Separation
 - GET<tab>/cgi-bin/broken.cgi<tab>HTTP/1.0
- Case Sensitivity
 - GET /CGI-BIN/broken.cgi HTTP/1.0
 - Windows systems are case insensitive
- Windows Delimiter
 - GET /cgi-bin\brotken.cgi HTTP/1.0

Copyright 2008 James Shewmaker and SANS Institute 28

IDS vendors have been incorporating checks for these evasion tactics over the last year. You have to make sure you've updated your IDS signatures (and do it frequently).

Fake Parameter mode involves inserting a parameter (that is, a variable) with a given value. The value is the cgi script. Again, it surprises me that this works, but it does.

```
GET /index.htm?param=../cgi-bin/broken.cgi HTTP/1.0
```

TAB Separation just substitutes tabs instead of spaces. Again, it functions on the web server, but doesn't match some signatures.

```
GET<tab>/cgi-bin/broken.cgi<tab>HTTP/1.0
```

Case Sensitivity can be altered when attacking a Windows box, which has weird case insensitive attributes. For the most part, Windows systems are case insensitive.

```
GET /CGI-BIN/broken.cgi HTTP/1.0
```

Windows Delimiters can be used as well when attacking Windows boxes. Instead of using the forward slash (/) like the RFC requires, an attacker could use the backslash (\).

```
GET /cgi-bin\brotken.cgi HTTP/1.0
```

For any of these, the signature isn't going to match, as we've varied the input at the application layer.

Vulnerability Scanners – Defense

- Preparation:
 - Close all unused ports
 - Shut off all unneeded services
 - In Windows, stop or delete services in services control panel, as discussed earlier
 - In Unix, edit /etc/inetd.conf or /etc/xinetd.d files, as well as rc.d files (remember chkconfig from earlier?)
 - Apply all system patches
 - Keep up-to-date!
- Identification:
 - Utilize an Intrusion Detection System signatures
 - Most vulnerability scanners trip hundreds of signatures
- Cont, Erad, Rec: N/A

Copyright 2008 James Shewmaker and SANS Institute 29

Again, this is not rocket science...still, it is difficult to keep up with all the patches on numerous types of systems. However, we must keep our critical systems patched and up-to-date.

The following are some key defense measures one can take:

Close all unused ports by shutting off all unneeded services.

In Windows, stop or delete services in services control panel, using the techniques we discussed during the port scanning section.

In Unix, edit /etc/inetd.conf or /etc/xinetd.d, as well as your rc.d files (as we covered with chkconfig earlier).

Make sure you keep your systems up to date by applying all system patches.

Finally, utilize an Intrusion Detection System, which specialize in detecting this type of scanning tool. Popular IDS solutions include:

Commercial

Sourcefire Intrusion Sensors, ISS RealSecure, Cisco Secure IDS (Formerly NetRanger), others...

Freeware

Snort, Shadow, Courtney

Exploit Mess

- There are thousands of different exploits
- There are hundreds of different payloads
- Creating new exploits and payloads from scratch can be a chore
- Stitching them all together can also be a pain
- No standards
- Lots of overlapping functionality
- Exploit quality varies greatly

Copyright 2008 James Shewmaker and SANS Institute 30

There are thousands of different exploits in the wild that can be used for compromising systems. However, most exploits are one-off affairs, written in a custom fashion to provide some form of access to a target. Some give a command shell, others run a single command. Some are in C, others are in assembly language. Some run on Windows, others in Linux. Some are high quality, while others stink and barely operate.

There is no rhyme or reason to a lot of these things floating around on the Internet. How could someone tame such a mess?

Taming the Mess: The Metasploit Exploit Framework

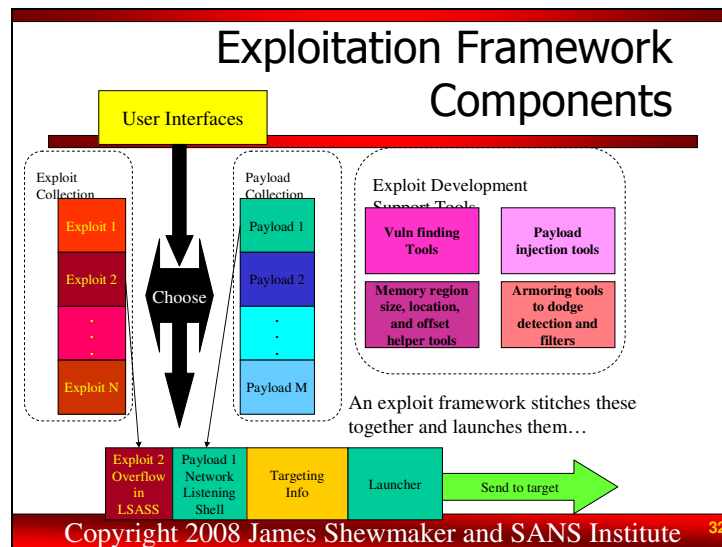
- Written by H.D. Moore, originally released Oct 2003, at www.metasploit.com
- Regular updates followed throughout 2004-2007
- Runs on Windows, Linux, BSD, and MacOS X
- A modular interface tying together:
 - Exploit, Payload, and Targeting (dest IP addr, port, options)

Copyright 2008 James Shewmaker and SANS Institute 31

To help tame this mass of different exploits, H.D. Moore and his team at Metasploit have released the Metasploit framework. This tool, which runs on Linux, *BSD, MacOS X, and Windows, creates a modular interface for tying together exploits, payloads, and targeting information. By creating a simple yet powerful architecture for stitching together custom exploits from modular building blocks, the Metasploit framework is an ideal tool for attackers and penetration testers.

Using the framework, the attacker first selects a particular exploit from the pool included in the tool. Currently, over one hundred exploits have been defined for the tool. Then, the attacker selects a payload that the exploit will cause the target machine to run. Many payloads have been defined as well. Finally, the attacker configures targeting information, including the destination IP address and port number, as well as any options required for the particular exploit or payload.

Then, the Metasploit framework does its magic. It creates a custom package including the exploit, the payload, and the targeting info, and launches this package against the victim machine. By assembling the package on the fly, the attacker has amazing flexibility in launching an attack.



Here are the essential components of Metasploit. The tool holds a collection of exploits themselves, little snippets of code that force a victim machine to execute the attacker's payload. Metasploit has over one hundred different exploits today, including numerous common buffer overflow attacks. Next, the tool offers a set of payloads, the code the attacker wants to run on the target machine. Some payloads create a command-shell listener on a network port, waiting for the attacker to connect and get a command prompt. Other payloads give the attacker direct control of the victim machine GUI across the network by surreptitiously installing VNC, the GUI remote-control tool. Users of any of these exploit frameworks don't even have to understand how the exploit or payload works. They simply run the user interface, select an appropriate exploit and payload, and then fire the resulting package at the target. The tool bundles the exploit and payload together, applies a targeting header, and launches it across the network. The package arrives at the target, and the exploit triggers the payload, running the attacker's chosen code on the victim machine. These are the things that script-kiddie dreams are made of.

Script kiddies aside, in addition to the exploits and payloads, the frameworks also feature a collection of tools to help developers create brand-new exploits and payloads. Some of these tools review potentially vulnerable programs to help find buffer overflow and related flaws in the first place. Others help the developer figure out the size, location, and offset of memory regions in the target program that will hold and run the exploit and payload. Some include code samples to inject a payload into the target's memory, while still others help armor the resulting exploit and payload to minimize the chance it will be detected or filtered at the target. And, here's the real power of these tools: if a developer builds an exploit or payload within the framework, it can be used interchangeably with other payloads or exploits as well as the overall exploit framework user interface.

Exploits Currently Included Metasploit Framework

- New exploits released on a regular basis
- Most recent exploits focus on client-side attacks and appliance vulnerabilities
- So, it's not just Windows, not just services any more

-Windows WMF flaw	-AWstats configdir CGI flaw
-Mozilla CompareTo	-RealServer Linux Overflow
-3Com FTP Server	-Arkeia Backup Client
-Barracuda anti-spam firewall appliance	-Veritas Backup (several)
-Google search appliance	-CA Brightstor Backup (several)
-Microsoft IIS (several)	-MS Exchange 2000
-Windows PnP Overflow	-FutureSoft TFTP server
-Microsoft ASN.1 attack against LSASS	-Gnu MailUtils Imap server (format string)
-RPC DCOM	-HPUX FTPD file directory listing
-LSASS	-HPUX LPD flaw

Copyright 2008 James Shewmaker and SANS Institute 33

The Metasploit framework currently includes over one hundred different exploits. Given the flexibility of the tool, and the prolific work of the tool's authors, I expect we'll see many, many exploits added to this list in the future. Once new holes are discovered and exploitation code is written, adding the new exploit to the Metasploit framework is quite straight forward. The current exploits include some of the biggies of recent months. Look at the following exploits as an indication of the evolution of the attacks:

Note that many of the exploits are targeting client-side components, such as browsers (the WMF flaw and Mozilla CompareTo exploits).

Also, note the recent rise in exploits for appliances (3Com FTP Server, Barracuda anti-spam firewall, and the Google search appliance).

Note also that some exploits are buffer overflows (RPC DCOM and LSASS), while others are problems with user input checking in CGI scripts (AWstats CGI flaw).

Furthermore, note that a fair number of these exploits target back-up solutions (Arkeia, Veritas, and CA's products specifically).

Finally, note that some work against Windows (Windows PnP, etc.), while others are targeted to other systems (HPUX, etc.)

Payloads Currently Included in Metasploit Framework

- Goal: Make reliable exploits that work in most environments and clean up after themselves
- Many payloads to choose from!
 - Bind shell to current port
 - Bind shell to arbitrary port
 - Reverse shell back to attacker (shoveling shell!)
 - Windows VNC Server DLL Inject
 - Reverse VNC DLL Inject (shoveling GUI!)
 - Inject DLL into running application
 - Create Local Admin User

Copyright 2008 James Shewmaker and SANS Institute 34

The primary goals of the Metasploit payloads include functioning in most environments (i.e., working across various Operating System patch levels, hotfix installs, service packs, etc.) and cleaning up after themselves (i.e., don't leave the system or a service crashed). Most of the payloads are available in several forms: Assembly language (ASM), C language, Perl, and Win32 EXE format. The C, Perl, and EXEs are essentially just loaders for the machine language routines.

The payloads available within the framework include:

Bind Shell to current port – this opens a command shell listener using the existing TCP connection used to send the exploit.

Bind Shell to arbitrary port – this opens a command shell listener on any TCP port of the attacker's choosing.

Reverse Shell – this payload shovels a shell back to the attacker on a TCP port. The attacker will likely have a Netcat listener waiting to receive the shell.

Windows VNC Server DLL Inject – this payload allows the attacker to remotely control the GUI of the victim machine, using the Virtual Network Computing (VNC) tool, sent as a payload. VNC runs inside the victim process, so it doesn't need to be installed on the victim machine. Instead, it is inserted as a DLL inside the vulnerable process.

Reverse VNC DLL Inject – this payload inserts VNC as a DLL inside the running process, and then tells the VNC server to make a connection back to the client, in effect shoveling the GUI. That's scary and amazing at the same time.

Inject DLL into running application – this payload injects an arbitrary DLL into the vulnerable process, and creates a thread to run inside that DLL.

Create Local Admin User – this payload creates a new user in the administrators group with a name and password specified by the attacker.

Newer Metasploit Payloads - The Meterpreter for Windows

- The Meterpreter – general-purpose module giving ability to load and interact with DLLs in real time, after exploitation has occurred, and interact across the network with the DLL
- Has been used to create command-line access within a running process
 - Doesn't create separate process (no extraneous cmd.exe)
 - Helps attackers avoid getting spotted by living just inside the exploited process
 - Doesn't touch hard drive, unless you want it to
 - Helps attackers to evade some forensics analysis techniques
 - Doesn't need any system-provided command executables for its command shell; they are built-in to the meterpreter
 - A little command-line environment that includes the basics
 - Helps attacker function on a system that has been seriously stripped down
 - Easily extendable by adding new DLLs
- Currently, only implemented for Windows targets, but concepts could be ported

Copyright 2008 James Shewmaker and SANS Institute 35

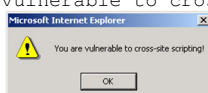
One of the most promising recent payloads of Metasploit is the Metasploit Interpreter, called the Meterpreter for short. This general-purpose payload for Windows targets carries a DLL to the target box to give command-line access. Ho-hum... you say. We've had that with the connect a shell to arbitrary port payloads in the past. What makes the Meterpreter special?

Its beauty lies in three aspects: 1) The Meterpreter does not create a separate process to execute the shell (such as cmd.exe or /bin/sh would), but instead runs it inside the exploited process; 2) The Meterpreter does not touch the hard drive, but gives access purely by manipulating memory; 3) the Meterpreter includes its own set of commands that are injected into a target process, so it doesn't need to use any executables on the target machine; 4) The Meterpreter can load new modules, dynamically changing its functionality while still inside the memory of the exploited process.

Those last two aspects are the most powerful. For example, rather than relying on the "hostname.exe" command, the Meterpreter includes its only module for printing system information, loaded into the DLL dynamically. An attacker can inject additional modules into the Meterpreter on the fly, with whatever abilities the attacker can dream up and code, all using the Meterpreter command channel to interact with the attacker.

What is Cross-site Scripting?

- Consider a web site that gathers user input
- User input is sent back to user's browser without filtering
 - "You just typed in the following, right?" [user_input]
- Attacker crafts URL with a script in it
 - Script in the URL is sent to server as user input
 - User input displayed back to user; script "reflected" back to client
 - Script runs on client browser
- Which do you want to search for? I want to search on:
 - `<SCRIPT LANGUAGE=Javascript>alert ("You are vulnerable to cross-site scripting!");</SCRIPT>`



So there, I can hack myself!

Copyright 2008 James Shewmaker and SANS Institute 36

Cross-site scripting allows an attacker to steal information (such as cookies) from users of a vulnerable web site. So, if your on-line bank is vulnerable, I might be able to steal your banking cookies.

Cross-site scripting is based on web applications that reflect user input back to a user. Many web applications do this. Consider a search engine. You type in the search string, and the application says back to you "You just searched for:" followed by what you typed. Or, how about a loan application? You type in your address, and the application says back to you: "You said your address was this:" followed by what you typed.

Cross-site scripting involves sending scripting code (usually Javascript or VBScript) to a web application that sends data back to the browser. The web server has an application that reflects user input back to a web browser. When the code gets to the browser, it is executed. Upon reaching a browser, the script on this page pops up a dialogue box.

You may be thinking: So what?!?!? I can type in scripts and hack myself... what's the big deal?

Launching Cross-site Scripting Attacks

- Attacker's script must be sent to the victim
 - URL embedded in an e-mail or newsgroup posting
 - URL provided on a third-party web site (either clicked on by victim user or automatically loaded when visiting a malicious web site)
 - Inter-user communication within the target site (i.e., message board, etc.)
- Amazing website with various Cross-Site Scripting encoding tools at <http://ha.ckers.org/xss.html>

Copyright 2008 James Shewmaker and SANS Institute 37

To accomplish the dirty deed, the attacker must get the malicious browser script onto the victim's machine. This can be accomplished in at least three ways:

The first method involves sending the victim an e-mail. I could formulate an e-mail that says, "Get \$20 in your account now, if you just click here". Users might click the e-mail link, forwarding the input to the web app, which reflects it back to the browser, where it runs.

Secondly, I could just put a link on my web site, saying "Click here for a great deal!" But the link under the text would include a URL with the embedded Javascript. It flows to your browser when you click, and runs there.

Thirdly, a lot of web sites allow one user to post data that other users see, such as a message board like slashdot (www.slashdot.org). I post an article that includes Javascript. When you view my article, the Javascript is downloaded to your browser, runs, and steals your cookies.

Using one of these three mechanisms, the attacker gets the user to view a web site to see data entered by the attacker.

The website ha.ckers.org/xss.html includes tools to help attackers encode their Cross-Site Script attacks for various browsers (IE versions, Firefox, Mozilla, etc.), with various functionality, and in various encoding schemes (ASCII, Hex, Unicode, etc.).

Cross-site Scripting Walk-thru

- 0) Victim uses a web site that sets cookies on the victim's browser
- 1) Victim clicks on a URL or visits a web site that includes the malicious script
- 2) Victim user's browser transmits malicious code to the vulnerable target site as a web request
- 3) Target site *reflects* the malicious code back to the victim user's browser in the response to the request
- 4) Malicious code executes within victim user's browser under the security context of the target site

Copyright 2008 James Shewmaker and SANS Institute 38

Here's how cross-site scripting works, in five short steps.

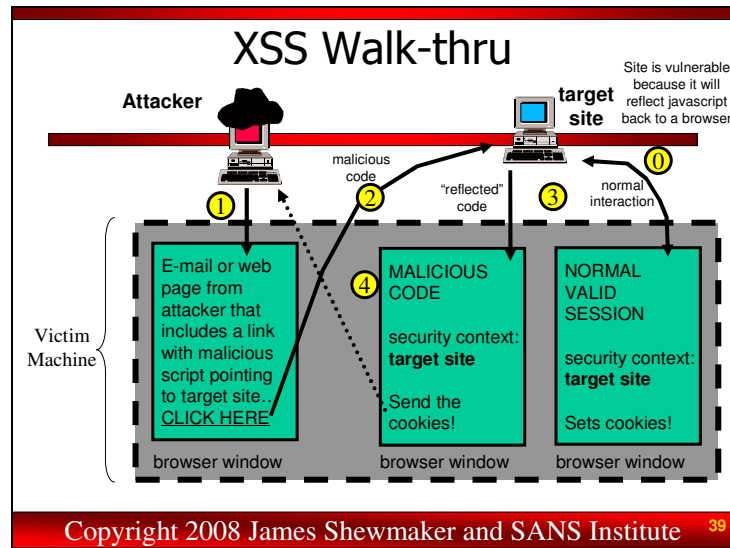
Step 0: The victim user sets up an account on a web server that is vulnerable to cross-site scripting. At some point in the application, a user's input is reflected back to the user without any filtering of scary scripting characters. The web application also might store cookies on the browser.

Step 1: The attacker sends the victim an e-mail, or tricks the victim into visiting a web site with a link. The victim clicks on the link that includes the embedded Javascript.

Step 2: The victim's browser transmits the script to the web application as user input.

Step 3: The web application reflects the user input back to the victim's browser. This user input, remember, is the script.

Step 4: The script runs on the victim's browser. This script runs in the security context of the target web site (the browser believes, after all, that the script came from the web site). Therefore, the script can grab all cookies for this site and send them via http or e-mail them to the attacker. Alternatively, the script could download an ActiveX control or Java program with a backdoor and install it on the victim machine.



This picture shows the details of the attack described in words on the previous slide. If you don't want to flip back a page, here is the description again:

Step 0: The victim user sets up an account on a web server that is vulnerable to cross-site scripting. At some point in the application, a user's input is reflected back to the user without any filtering of scary scripting characters. The web application also might store cookies on the browser.

Step 1: The attacker sends the victim an e-mail, or tricks the victim into visiting a web site with a link. The victim clicks on the link that includes the embedded Javascript.

Step 2: The victim's browser transmits the script to the web application as user input.

Step 3: The web application reflects the user input back to the victim's browser. This user input, remember, is the script.

Step 4: The script runs on the victim's browser. This script runs in the security context of the target web site (the browser believes, after all, that the script came from the web site). Therefore, the script can grab all cookies for this site and send them via http or e-mail them to the attacker.

Now, the attacker has the victim's cookies, which could include sensitive data. Alternatively, a session credential might be sent from the victim to the attacker, so the attacker could login to the application as the victim.

Not Just Scripts!

- Beyond inserting scripts, attackers could also insert text or even pictures to confuse web surfers
- A rash of these issues were discovered for news sites in September 2004
- CNN was a notable example, fixed in 24 hours
- E-mail to victim says to surf here:
 - <http://weather.cnn.com/weather/search?wsearch=%46%6f%64%22%20%3c%6d%20src%3d%74tp%2f%77w%2eobtu%73e%2e%6eet%2fa%2e%6apq%3e>
- That's really an encoded form of:
 - `http://weather.cnn.com/weather/search?wsearch=Florida" `
- Don't surf there now! Sometimes nasty pics...



Copyright 2008 James Shewmaker and SANS Institute 40

There are attacks that are related to cross-site scripting that don't actually involve scripts. Instead, another tag is inserted in a URL to confuse a user at a browser by loading an image or text from one website, making it appear to come from another site. Consider this example, which is one of a rash of these vulnerabilities discovered in September 2004.

The CNN weather website allows users to search for particular cities or states to find weather information. Someone discovered that a user could perform a search that includes an image tag, and the the CNN weather website would dutifully reflect that tag back to the browser unaltered. Then, the browser would fetch that image from another website and display it inline. During the height of the Hurricane season in September 2004, someone sent e-mail to a few people with a link to the CNN weather search page that looked like it was displaying the weather for Florida. But, in reality, the link was performing a bogus search that caused CNN to respond with an error message. But the error message included an image reference to a jpeg from another site altogether. The before-the-fix and after-the-fix web pages are shown on this slide. To disguise what was happening, the attacker substituted hex equivalents of some characters in the resulting URL. Several other news and financial sites were also vulnerable at this time, but quickly fixed the problem within 24 hours.

Pushing the Envelope: What Else Can XSS Do?

- As we've seen, XSS can let an attacker
 - Pop up dialog boxes - Boring!
 - Steal cookies - far more interesting
 - Make it look like content came from a site where it did not originate
- Harvest browser history
- Engage in transactions from within the browser against the vulnerable site
- Conduct a scan of an internal network
- Exploit administrative applications

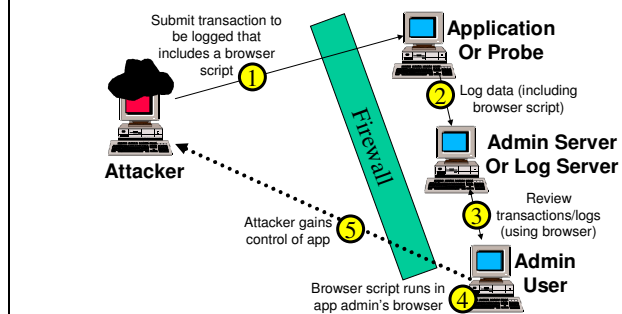
Copyright 2008 James Shewmaker and SANS Institute 41

We've covered three examples of XSS so far. First, we have the relatively pedestrian popping up of a dialog box. It's not all that powerful, but it could be used to trick users into typing in passwords or doing other things that are risky. For example, suppose a big bank were vulnerable to XSS. When a user accesses the bank, a dialog box opens telling the user to change their bank password to a stronger password like p@ssw0rd. A fraction of users will actually use that very value for their password, making the attacker's life easier. We've also seen how we can steal cookies using an XSS, a very powerful technique. And, we saw how cross-site content could make it look like a given website was hosting content that actually came from another location. But, wait... there's more that XSS can do... a whole lot more.

Some work has been done in getting a script to interrogate the browser's history in which it runs. A script cannot directly access the browser's history, but there are ways that the script can infer that a given URL was accessed by the browser. Particularly, the script could request that the browser surf to a page with a given link, and then check to see if that link when viewed in the browser has the style (such as the color) of an already explored link. If it does, that means the given link is in the browser's history. The script could cycle through hundreds or thousands of URLs this way to see which popular sites the browser has accessed.

Furthermore, because the script runs in the victim's browser, it can do anything that the browser itself can do on the vulnerable web site. So, the malicious browser script could, for example, engage in e-commerce transactions as that user on the vulnerable web application, perhaps transferring funds into the attacker's account or ordering merchandise to be delivered to the attacker. Or, the script, running from the vantage point of the victim's browser inside a network firewall, could start scanning or manipulating the internal infrastructure of the victim organization where the browser resides. Or, it could try to exploit administrative applications. Let's look at these possibilities in more detail.

Attacking Admins via XSS



Copyright 2008 James Shewmaker and SANS Institute 42

To visualize this kind of attack, suppose we have some sort of application that gathers input from a user and stores it, perhaps in logs for later administrator review. An administrator periodically views the stored content or logs that contain this user input sent by the attacker.

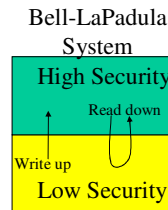
In Step 1, the attacker provides input of some kind that includes a browser script. In Step 2, the application logs this input from the user, perhaps passing it to a separate logging server. In Step 3, an admin user views the logs using a browser-based admin application.

In Step 4, the evil content the attacker inserted into the logs runs in the admin's browser, possibly stealing cookies from it and delivering them to the attacker. Alternatively, in Step 5, the script running in the admin's browser could alter the application in some way using the admin's credentials. It might even add the attacker as a new administrator account in the application.

We've seen numerous applications with this kind of vulnerability, including a cash management system in a bank, a credit card processing system, 3 enterprise anti-spyware tools, 1 enterprise information security tool, and others.

XSS via Other Mechanisms

- How to send scripts?
 - HTTP / HTTPS via web app (of course), E-mail, FTP
 - Other possibilities
 - US Postal Service
 - Mag Stripe
 - Electronic Data Interchange (EDI)? X.25? SS7?
 - With webified applications thirsty for scripts, any form of data input could be a vehicle for malware infiltration
 - Concerns for networks built around Bell-LaPadula model of “no write-down, no Read Up”
 - As designed, this stops info leakage. But it does allow for malware infection.



Copyright 2008 James Shewmaker and SANS Institute 43

Now, most people think that these XSS problems are associated only with content that arrives via the web itself. But, the problem goes further than that. Content arrives via all kinds of mechanisms into an organization, and is later viewed by employees who use browsers in web-based applications. For example, data may arrive via e-mail or FTP, and later be viewed in a web-based application by an employee. The data delivered via e-mail or FTP could contain browser scripts. The internal processes of the organization may store that data in a back-end database. At a later time, when an employee uses a browser with a web-based application to review the data, it runs in the user's browser.

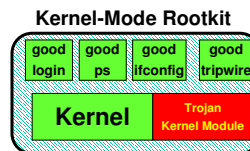
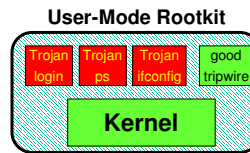
And the possibilities here are endless. In one penetration test, a company that receives hundreds of thousands of post cards in the mail was concerned about this kind of attack. When post cards were received, this organization scanned them in and used Optical Character Recognition (OCR) technologies to load the post card data into an internal application. At a later time, internal employees reviewed the data from the post cards using a web-based application. For the penetration test, the company paid testers to create two hundred post cards with browser script information carefully hand written on them. The testers then established web sites that would be accessed by any browser inside the company that rendered the content and ran the scripts on the post card. The testers then sent in the two hundred post cards and waited. Two days later, BINGO! Browsers of employees inside the organization did indeed access the tester's web sites, indicating that the content printed on the post cards did indeed execute inside the browsers of the target organization. Beyond the postal service, other vehicles for delivery of scripts include magnetic stripes of credit cards and Electronic Data Interchange (EDI) feeds.

In the end, any point of data input to an organization is an avenue for malicious script infiltration. And, any point of data egress is a possible avenue for data theft. This all comes back to limitations of the Bell-LaPadula security model, a common security model that many of our networks were designed around.

In that model, low security zones cannot read data from high security zones. Instead, they can write up to them. The high security zone can read down. But, in this model, malicious scripts in the low security zone can make it to the high security zone and wreak havoc there. Now, in a perfectly implemented Bell-LaPadula system, the script should not be able to exfiltrate data back into the low security zone. But, the damage caused in the high security zone could still be severe, and it is possible that the attacker could find some exfiltration point to squeeze out some data.

Upping the Ante with Kernel-Mode Rootkits

- Really quite amazing tools....
- User-mode Rootkits operate at the application level, replacing critical system executables
- Kernel-mode Rootkits operate at a more fundamental level, completely transforming your environment at the attacker's whim!



Copyright 2008 James Shewmaker and SANS Institute 44

Kernel-mode Rootkits represent the latest evolution of Rootkit-like tools. They use some ideas that were originally included in a tool named “sumfuq” for SunOS 4.1.X, and itf.c for Linux.

Because they run at the kernel-mode, kernel-mode Rootkits have much more power over the system than Rootkits that live at the process/program/application level. Detection is far more difficult, because detection programs themselves run at the process/program/application level and rely on the kernel to function.

For example, a good Tripwire routine running on a system can detect a user-mode Rootkit Trojan Horse program by opening the programs and looking at them. Keep in mind, however, that to open a program file, the file system integrity checker has to make calls into the system kernel. All access of the hard drive or any other hardware components of the machine must use the kernel.

With a kernel-mode Rootkit installed, the attacker doesn't have to modify the individual application programs. When the kernel is modified to lie to administrators, all files would be intact, while the actual kernel can give backdoor access and hide the attacker.

By operating at the kernel, the attacker can essentially create an alternate universe within your computer that looks intact and happy. Really, though, beyond this appearance, your system could be laced with Trojan Horse programs.

Types of Kernel Mode Rootkits

- In general, there are (currently) five different methods for manipulating the kernel being publicly discussed
 1. Loadable Kernel Modules (Unix) and Device Drivers (Windows)
 2. Altering Kernel in Memory
 3. Changing Kernel File on Hard Drive
 4. Virtualizing the System
 5. Running Programs Directly in Kernel Mode
- Each available on Linux and Windows

} The most popular method today

The real focus area lately in kernel-mode Rootkits is how to get them implemented. People have been doing loadable kernel modules and evil device drivers for kernel Rootkits since at least 1997. This technique remains the most popular today.

More recently, though, there has been tremendous progress in altering a live running kernel in memory and patching kernel images on the hard drive. That's some pretty nasty stuff!

In the future, we may even see attackers virtualizing systems to implement rootkits, or running programs directly in kernel mode. Although these latter two techniques haven't yet caught on, they are a possibility and have been discussed publicly.

Kernel-Mode Rootkit Defenses – Configuration Lockdown

- All of these attacks require the bad guy to have superuser privileges (root or Administrator)
- Harden the box by hand, or...
- Use a good security template
- The Center for Internet Security (CIS), in conjunction with NSA, NIST, and others, has developed a set of templates for Win, Lin, Solaris, HP-UX, Cisco Routers, and Oracle DBs
 - <http://www.cisecurity.org>
- They have scoring tools as well

Copyright 2008 James Shewmaker and SANS Institute 46

Hardening your systems to prevent superuser compromise is a critical first step. Microsoft ships Windows with a variety of security template files for workstations, servers, and domain controllers. However, these built-in security templates tend to be either way too weak so that any attacker can slice through them, or so strong that they render the system unusable in a real-world environment. What the world needs is a reasonable security template that isn't too weak, or too strong, but is just right for most environments.

The Center for Internet Security (CIS), the National Security Agency, and the SANS institute, together with a variety of other organizations, embarked on creating just such a template. They spent several months devising various standards that would meet the most pressing needs of all of these organizations. Finally, they achieved consensus and released several system hardening templates, available at <http://www.cisecurity.org>. These templates apply to Windows, Linux, Solaris, HP-UX, Cisco Routers, and Oracle Databases, as well as numerous other types of platforms (Microsoft Exchange Server, Microsoft SQL Server, Apache, BIND, Novell eDirectory, etc.). They serve as an excellent starting and reference point for your security configuration. You can tweak them to make them stronger, or loosen their restrictions for your environment.

The Center for Internet Security has also released free scoring tools, so you can check to see how well your security settings match a given template, such as the Win2K Pro Gold Template. You run scoring tools on a local system to compare your security stance to a baseline template, giving you a summary score between 0 and 10. The higher your score, the more closely you match the template used for comparison.

Hiding Files in NTFS

- If system is running NTFS, alternate data streams are supported
- Multiple streams can be attached to each file or directory
- Attacker's files can be hidden in a stream behind normal files on the system
 - Such as notepad.exe or word.exe (or anything else!)
- Use the **type** command built into Windows

```
C:\> type hackstuff.exe > notepad.exe:stream1.exe
```
- Or, use the **cp** program from the NT Resource Kit

```
C:\> cp hackstuff.exe notepad.exe:stream1.exe
```
- To get data back, it can be copied out of the stream

```
C:\> cp notepad.exe:stream1.exe hackstuff.exe
```
- Alternatively, you can create an alternate data stream attached to a directory by simply typing:

```
C:\> notepad <file_or_directory_name>:<stream_name>
```

Copyright 2008 James Shewmaker and SANS Institute 47

Why? Apple compatibility!

The Windows NT File System (NTFS) supports a feature known as file streaming. File streaming applies only to NTFS partitions; it does not apply to FAT partitions. Each file on an NTFS partition acts kind of like a chest of drawers. The name of the chest is the name of the file itself. Underneath this name, there can be arbitrary numbers of data streams associated with the file's name. Each data stream acts as a drawer into which a user can place data. All streams are still associated with the name of the original file. The first stream associated with a file is the contents of the file itself, the thing you see when you look at it in the Windows Explorer. When you look at a file in the Windows Explorer, you see the file name followed by the size of the first stream.

An attacker can create additional streams associated with any file or directory name on the system. The attacker can then use these streams to hide their sensitive information, such as attack tools or sniffer logs. Any file or directory on an NTFS partition can be used to hide such information, such as Notepad or Word. To create and interact with file streams, the type command built into Windows can be used. Alternatively, the cp program may be used from the Windows NT Resource Kit. The attacker simply copies their information to be hidden into the file name, followed by a colon, followed by the stream name assigned by the attacker. This stream acts like another drawer in the chest of drawers.

Alternate Data Streams in NTFS

- The hidden file in the stream will follow the other file around through normal copying between NTFS partitions
- On Linux machines that have connected to a Windows box with NTFS, smbclient can get data from ADSs
- But, Pre-Vista Windows offers no built-in capability for finding or deleting a stream
 - To delete a stream, you could move file to FAT partition and then move it back
- Vista only shows it when using "dir /r" not in the GUI

Copyright 2008 James Shewmaker and SANS Institute 48

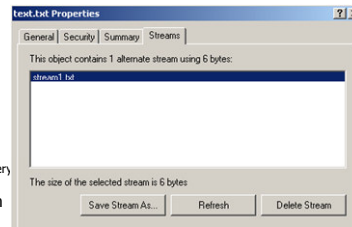
When the data is hidden in the stream, the Windows Explorer and dir command still only show the name and size of the first stream of data (the top drawer). The name of subsequent streams and the size of subsequent streams are not listed in Windows Explorer or dir. When you move a file that has many streams associated with it, all of the streams move on the system. For example, if you copy a file between partitions or move it across a Windows network share, Windows will send all of the streams. Therefore if you have a small file, such as say 6 KB, and it has a large stream associated with it, say 1 MB, you'll be moving all of the contents of the file even though it appears to be only 6 KB long. Therefore if you move such a file via Windows file sharing, it will take quite a lot of time as it transfers over more than 1 MB of data.

Interestingly, the Linux smbclient program can also read data from Alternate Data Streams from a Windows share, but the attacker needs to know the stream name to be able to refer to it and pull out the data.

It's important to note that Windows offers no built-in capability for deleting a stream, however. To delete a stream using built-in Windows functionality, you could move the file with the stream to a FAT partition (such as a floppy disk) and then move it back. The stream will then be dropped.

Finding Hidden Streams

- Use anti-virus tool to find malicious code in streams (nearly all have it now)
- Many anti-spyware tools lack ADS detection functionality
- Third party tools for finding alternate data streams in NTFS
 - LADS by Frank Heyne, at www.heysoft.de/index.htm
 - Streams by Mark Russinovich, at www.sysinternals.com
 - Includes an option for deleting a stream – very useful feature!
 - Streams shell extension utility by Ryan Means, at http://www.giac.org/certified_professionals/practicals/gcwn/0230.php



To detect malicious software hidden in a file stream, you must make sure to use an updated anti-virus tool. Over the last couple of years, most major anti-virus manufacturers have included the ability to search through file streams to find malicious software. Prior to that, anti-virus tools did not look into additional streams. Today, many anti-spyware tools do not look into ADSs during their real-time or on-demand scans. Thus, spyware can stay undetected on a system by loading into an ADS and running from inside it.

LADS is a tool dedicated to finding alternate data streams in NTFS. It is very well written, and I strongly recommend that you get a copy and become familiar with it for future forensic use. Another useful tool is "streams" by Mark Russinovich, at www.sysinternals.com. The "streams" program includes a very handy option for deleting a stream without impacting the host file. Finally, Ryan Means wrote a shell extension utility that adds stream viewing capabilities to Windows in the Properties window for each file, as shown on the screenshot above.

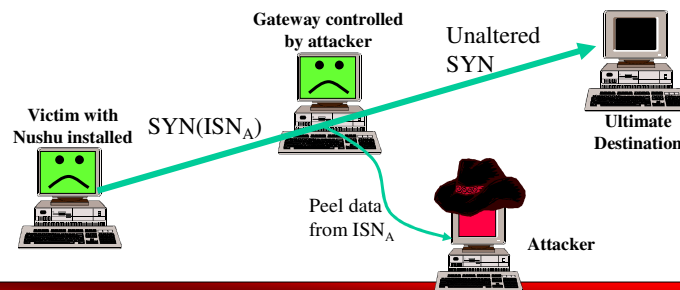
Passive Covert Channels

- Most covert channels generate their own packets to send, hiding the data inside those packets
 - Covert_TCP
 - Various HTTP, HTTPS, and SSH tunnel programs
 - Loki, ICMP tunnel, etc.
- Passive covert channels utilize existing packets, inserting their data inside
- Joanna created a reference implementation released called Nushu
 - Check out <http://www.invisiblethings.org/papers.html>
 - Named after a secret language created by Chinese women centuries ago

Covert_TCP, although effective, isn't the only game in town when it comes to inserting data into the sequence number of the TCP header. A tool named Nushu utilizes this technique as well, but with an important twist. For each character it sends, the older Covert_TCP tool generates a packet containing that data, which it transmits either to the destination or a bounce server. In other words, it is an *active* tool, generating its own packets. Nushu, a tool written by Joanna Rutkowska, creates *passive* covert channels. Instead of sending its own packets to exfiltrate data, Nushu inserts its data for the covert channel inside of packets generated by other applications running on the machine where Nushu is installed.

Passive Covert Channels in Action

- Idea: Send data inside of SYN packets by tweaking ISNA
 - Paper contemplates using other TCP header fields, or even HTTP cookies
- Strip data off while its on its way to destination using a gateway

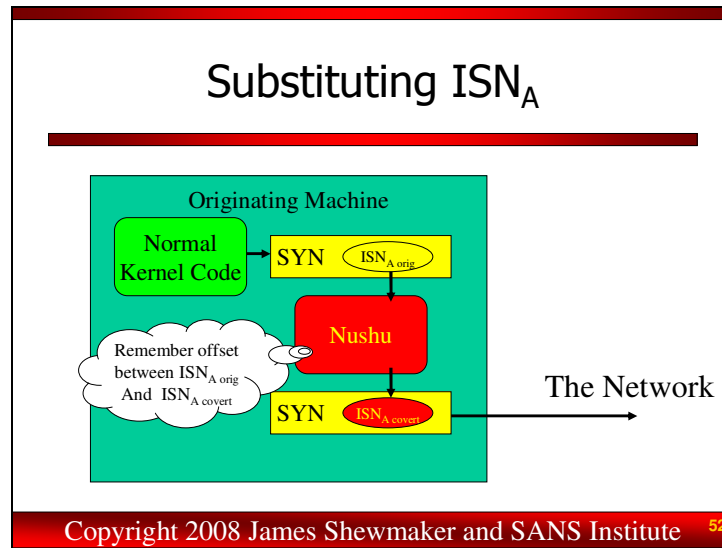


Copyright 2008 James Shewmaker and SANS Institute 51

The victim machine with Nushu running sends packets out across the network in its normal course of operation. These packets represent the actions of users and services on that system, potentially including HTTP, SMTP, FTP, or other protocols. Nushu, sitting silently on the victim machine, waits for new TCP connection initiations. When it sees the kernel of the victim system generate a new TCP session using the three-way handshake, Nushu alters the ISN_A of that packet to insert its data inside of it. The data is then transmitted across the network to its ultimate destination; a machine the attacker doesn't control or have any interaction with whatsoever. The attacker also must control a gateway on the network through which all of the victim's traffic is sent. This could be a router or gateway system run by the victim's ISP under control of the attacker, or a system the attacker inserted in the communication using the ARP cache poisoning or DNS spoofing attacks we discussed earlier in the class.

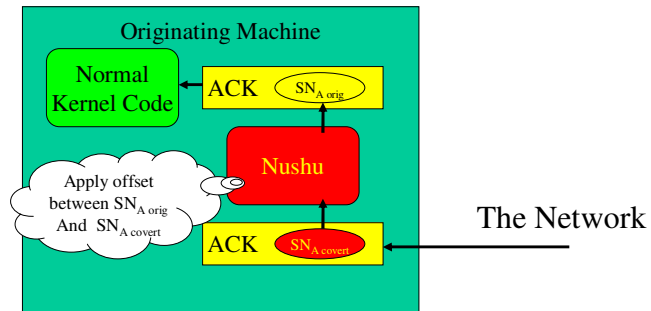
At this gateway system, the server component of Nushu copies data from the ISN_A and forwards the secret information to the attacker. The SYN packet itself is then dutifully carried to the ultimate destination, which responds with packets of its own. In a sense, the attacker is leaking data out of the victim machine using the first packet of the three-way handshakes engaged in by other applications. Now, the attacker will only be able to send up to 32 bits of data per outbound TCP session from the victim, as only the ISN_A can carry data (all of the other sequence numbers in outbound packets for that session are just increments of that ISN_A for each octet of data transmitted). Still, if the victim machine is a fairly busy server or a client actively surfing the Web, enough TCP session initiations will occur to set up a powerful and quite covert channel.

Substituting ISN_A



To pull this off, Nushu is implemented as a special Linux kernel module, altering the kernel to intercept packets on their way out of the victim machine. Before the kernel transmits the packet, Nushu grabs it. Nushu then takes a piece of data it wants to transmit, and calculates the offset between the data it wants to send and the original ISN_A the kernel itself generated for that packet. Nushu will remember this offset for the duration of this session. Then, Nushu inserts its data into the ISN_A field and transmits the packet.

Dealing with ACKs



Copyright 2008 James Shewmaker and SANS Institute 53

Next, we can see why remembering that offset is so important. When a response packet is received from the ultimate destination, Nushu has to map the acknowledgment number (which is based on the Nushu-set sequence number in the original packet, simply incremented by one or the number of octets transmitted so far on that session) back to the original sequence number (incremented appropriately). Nushu subtracts the offset from the acknowledgment number field and hands the packet to the normal kernel routines for handling.

Nushu – How Much Data

Control byte	Data byte 2	Date byte 1	Data byte 0
--------------	-------------	-------------	-------------

- This is the format Nushu uses for the sequence number in the SYN packet
- Using this techniques, we can only carry 3 bytes per TCP connection
- If someone is surfing the web, and all data goes through gateway, this is still a reasonable data channel
- Interesting anomaly – local tcpdump of packets have different sequence numbers than network-sniffed packets!

Nushu does introduce an unusual anomaly when implementing this process. If investigators run a sniffer, such as tcpdump, on the victim machine, they will see the sequence numbers generated by the normal kernel code. They can then compare those local sequence numbers with the sequence numbers of supposedly the same packets sniffed from somewhere on the network between the victim and ultimate destination. By comparing these two sets of sequence numbers for what are supposed to be the same packets, they will see a difference! The sequence numbers in the packets sniffed locally versus the packets sniffed from the network will be different by the offset for each session.

Currently, Nushu is implemented only in Linux. This same kind of technique could be particularly insidious in Windows-based spyware, given the widespread use of Windows machines and the proliferation of Windows spyware, although such software hasn't yet been released publicly as of this writing.

Steganography (Stego)

- Steganography abbreviated as stego, not to be confused with stenography
- Involves concealing the fact that you are sending "sensitive" information
- Data hiding
- Relatively new field
- Can hide in a variety of formats
 - Images
 - Bmp, Gif, Jpg
 - Word Documents
 - Text Documents
 - Machine Generated Images
 - Fractals, complex crowds of animals/flowers/people...

Steganography is a fairly new, but very active and growing field. It involves hiding data within a file, such as an image or sound file, so that the meaning of the message and the fact that a message is being sent, is concealed. There are numerous methods allowing data to be embedded in a wide range of file types. Data can be hidden in images, such as bitmap, GIF, or Jpeg files; in Microsoft Word documents; or even in computer-generated pictures, like fractals or complex crowds of animals/flowers/people.

Hydan

- Hydan hides data in executables written for i386
 - Written by Rakan El-Khalil
 - Supports *BSD, Linux, and WinXP
 - <http://www.crazyboy.com/hydan/>
- Start with an executable, as well as message to hide
- Feed both through Hydan
- Hydan encrypts the data with blowfish with user-provided passphrase and then embeds the data
- Result: One executable, *same size*
- Take the resulting executable... it'll still run
- However, by sending it back through Hydan, the original message can be recovered



In early 2003, Rakan El-Khalil released the “Hydan” tool. It hides data in executable programs written in the x86 instruction set.

To use the tool, you’d start with an executable program, such as Microsoft Word. You’d also need a message to hide, such as a secret message, a picture, some other code, or anything, really.

You feed the executable and message into Hydan. You also tell it the passphrase you want to use.

The tool first encrypts the message with the blowfish encryption algorithm using your passphrase as a key.

It then embeds the encrypted message inside the executable program.

The result is a single executable that includes the hidden encrypted message. This executable is the same size as the original executable, and the exact same functionality.

Most importantly, by using Hydan again, the original message can be retrieved from the resulting executable.

How Hydan Hides Info

- First off, it just encrypts the message using blowfish
- Next, it uses polymorphic coding techniques to hide the data
- Hydan has several groups of functionally equivalent instructions
 - Add X, Y vs. Sub X, -Y
- By choosing an instruction from one group, we get a “zero” bit
- By choosing an instruction from another group, we get a “one” bit
- Just encode all the bits like that!
- Then, rewrite the polymorphic executable

So, how does Hydan accomplish this little feat?

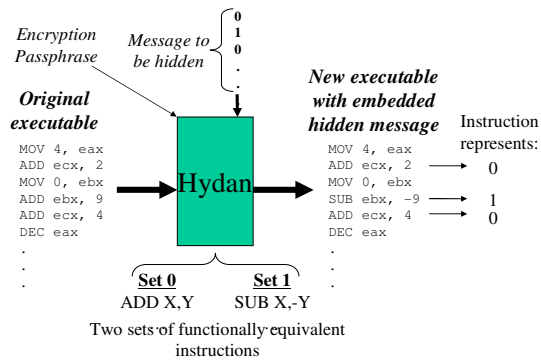
First off, it just encrypts the message to be hidden.

Then, it uses polymorphic coding techniques. Hydan defines different sets of CPU instructions that have the exact same function. For example, when you add two numbers, you can use the “add” or “subtract” instructions. You could add X and Y, or you could subtract negative Y from X. These have the exact same result.

Hydan takes the original executable, and rebuilds it by choosing instructions from one group or the other group of functionally equivalent instructions. If a given bit to be hidden is a zero, we will choose from the first group of instructions. If the bit is a one, we will choose a functionally equivalent instruction from the other group of instructions.

Then, after the entire code is rebuilt with these instructions, the executable is rewritten dynamically to the hard drive. The resulting executable has exactly the same size, and exactly the same function!

Hydan in Action



So, Hydan dynamically rebuilds the executable from the ground up, making substitutions of adds and subtracts to hide the necessary bits. The resulting executable's size is exactly the same because ADD and SUB are the same size. Now we can store a "0" or a "1" depending on whether the instruction is an ADD or SUB.

Uses for Hydan Technique

- Data hiding in any executable
 - User program, service, kernel module...
- Digital watermarking an executable
- Digitally signing an executable
- Polymorphic code for signature evasion
 - Not quite there yet with Hydan...
 - Stay tuned!

Copyright 2008 James Shewmaker and SANS Institute 59

If a company use this technique what happens to the Hash values??

Lets twist hydan and open it up so we can change the bot or worm

How can this functionality be used? Hydan can be employed to:

Hide data inside any executable to communicate covertly. The data could be crammed inside a user program, service, or even a kernel module using the technique.

Watermark an executable. By using Hydan, a user can mark an executable so that a copy of the program can be recognized easily.

Sign the executable. By using Hydan to embed a digitally signed hash of the original program, a user can verify that he or she was the author of a program. This could be useful in software copyrighting mechanisms and digital rights management for executables.

Finally, the technique could be used to implement signature evasion for network-based IDS tools and anti-virus programs that are looking for specific sequences of bits (signatures) to identify malicious software. Pretty evil! It's important to note that Hydan doesn't yet do this. It doesn't have enough different types of polymorphic substitutions to do effective signature evasion. Enough of the original program survives so that signature matching still works. However, these concepts could be extended to achieve signature evasion... Stay tuned!

Near-future Trends in Attacks

- Distributed attacks on the rise
 - Distributed scanning
 - Distributed password cracking
 - Worms, worms, and more worms
 - Bots, bots, and more bots
- The war extends beyond servers to the desktop
 - Cable modems & DSL: Always-on and high bandwidth
 - Virus/worm infection used to spread and stage attack tools
 - Target home users and telecommuters

Copyright 2008 James Shewmaker and SANS Institute 60

Distributed attacks help attackers by speeding things up, allowing them to consume more resources, and making them harder to detect. We are rapidly moving beyond simple DDoS. Some interesting distributed attack tools:

Distributed Scanners are also popular. Here are some examples:

Phpdistributedportscanner

<http://www.php-resource.de/scripte/show/934/>

The master controls scanning nodes using HTTP POST requests.

Dscan – client/server

<http://www.packetstormsecurity.org/distributed>

SIDEN – client/server

<http://siden.sourceforge.net/>

Additional Near-future Trends

- Client-side attacks proliferate
- Attacks against cell phones and PDAs
- AJAX and Web 2.0 security just starting
- Undermining user-based trust models
 - SSL
 - SSH
 - Active Browser Content – ActiveX
 - Others...

Copyright 2008 James Shewmaker and SANS Institute 61

We are seeing many attacks against client-side software, such as browsers, music players, image viewing tools, etc. This will continue to be a dominant vector for some time.

Also, as more features and power is added to smaller platforms like cell phones and PDAs, attackers will likely increasingly target these types of tools. With full-fledged operating systems and useful development environments, attackers are starting to create malicious code for cell phones. Watch for more of that in the near future.

User-based trust models are just plain scary, because users do not understand what's going on from a technology perspective. Whenever we have a technology that pops up a dialog box asking the user, in effect, "Something really dangerous is about to happen; do you want to let it?", we are going to have trouble. But, that is what we have created with SSL, SSH, ActiveX, and several other technologies. We either have to move users out of the trust loop (unlikely) or build our tools so that they understand what the implications are.

Contact me

- James Shewmaker
 - 1-562-438-5121
 - jim@bluenotch.com
 - jimshew@hotmail.com
 - <http://www.bluenotch.com/>

- 10% Discount use : **COINS-JS**

www.sans.org

Copyright 2008 James Shewmaker and SANS Institute 62

Please feel free to contact me, email is best. If you send me an email to jim@bluenotch.com I should be able to respond in a few days. Try not to have a SPAM-like subject and it will make it into my inbox. I have a copy of this presentation on my site, as well as some others you might find interesting.

The SANS Institute is trying to encourage more collaboration and build a better security community. SANS is extending a group discount to us, just use COINS-JS as a discount code and you will get 10% off a full length (6 day course).

This material is heavily edited but based on excerpts from SANS Security 504, Hacking Techniques and Incident Handling by Ed Skoudis.

I'll be sticking around for a while, so please feel free to ask me about this material, SANS, GIAC, or frankly anything else. Thanks!