Your physical environment

Today's computing environment has changed: more smaller boxes such as the ones used in this class are becoming more common. And even those are now being partitioned into smaller logical computers, such as with **Xen** and **vmware** (changing the trend seen in the mid to late 90s when large computers such as Sun Enterprise 10000 class were being divided into a 8 or 16 "domains".)

Instead of seeing large boxes sitting on their own in a computer floor, generally smaller computers are “rack-mounted”, and generally follow the convention of specifying their height in “u” notation:

☞ 1U == 1.75 inches

☞ 2U == 3.5 inches

☞ 3U == 5.1 inches

☞ 4U == 7 inches

☞ 5U == 8.34 (or 8.75) inches

Fortunately, the physical challenges are less with smaller boxes: they are less bulky, they weigh less, they (generally) consume less power and less cooling than the behemoths of old. Some challenges do grow: more power sockets are necessary to handle more machines in a single rack.

Maintenance? Should I buy a hardware maintenance contract?

☞ If you are buying essentially commodity equipment (i.e., it is using standard low-end parts that can be easily purchased from multiple vendors), then this question becomes less one for the system administration group than for the budget department. If you are purchasing this equipment from a major vendor, you are likely to receive a warranty that is sufficent (3 years being about the maximum that you want to plan to keep any piece of equipment these days.)

☞ If you are however purchasing non-commodity equipment, then you may well want to consider not only buying a hardware maintenance agreement, but also consider buying extra equipment for your own pool of spares.

☞ Third party maintenance: while this can make economic sense for commodity equipment (and is more of an economic question than a system administration one), having had experience with third party maintenance on non-commodity equipment, I would recommend that you stay away from it. The ugliest experience of all is when the hardware maintainer maintains that it is a software fault and the original manufacturer exclaims that it is a hardware problem.

☞ Onsite maintenance versus swap maintenance: onsite maintenance means that the service provider will send someone to work on your problems; with swap, you get to do the diagnosis (perhaps with the vendor helping on the other end of a telephone line), and then the vendor dispatches a new board or other equipment, generally via overnight mail.

Safe board-handling

☞ As we discussed during the installation of the ethernet card, it is best practice to use a grounding strap to minimize discharge of static electricity.

☞ Reseating boards: years ago, it was quite common to take an ordinary pencil eraser and clean the board contacts if a machine was behaving oddly. While USAH still recommends this (page 747), usually just reseating a board is more than sufficient to correct these type of problems.

- ☞ Simply pressing down on socketed ICs does still work, although finding a socketed IC is getting very rare these days.

Monitors: going to LCDs

Fortunately, we are moving away from the old CRTs, some of which we still have here in room 016, and toward the use of LCD technology. These are much easier to move, and in the event of an earthquake (not likely here in Tallahassee!), are much less dangerous if they fall on or near you.

But for those times when you must work with the old glass CRTs, keep in mind that they are relatively fragile and also are not safe to open up. If they don't work, there are not any system administrator-level parts inside of a monitor that are serviceable.

Memory

As USAH points on page 747, memory these days is a commodity item. Prices are very attractive (a gigabyte of memory from Newegg is currently around $25 as of this morning for high-end applications), and generally you should not buy memory even for a non-commodity machine from the vendor if you can avoid it.

Memory is very static sensitive, and you should strongly consider using a grounding strap whenever you work with it.

Removing memory is generally more challenging than inserting it. Generally, it snaps in quite easily, but removal on some (older) machines is not as easy — the worst in my memory was a box from Sun where I succeeded in breaking more than one (expensive) memory board on a particularly bad day.

Preventive maintenance (also known as "p.m.")

In the past, preventive maintenace was a big job duty of system administrators. Cleaning printers, cleaning tape units, vacuuming or blowing out dust: there was certainly a plebian side to being the system administrator. Fortunately, there is less call for such mundania these days except for dust problems, which are still not unusual.

As USAH points out on page 748, some preventive maintenance is done due to human failings. Stacking books over vents is still all-too-common. Even rack-mounted systems are not immune to air problems. The worst that I have seen was was when cardboard was casually inserted into a rack system that had closed doors. Until I happened to open the doors on the rack for that machine (which was showing a significant over-temperature reading on its internal sensors), I had no idea why it was getting too warm.

But the parts that fail most often are fans and power supplies. Fortunately, these days it is becoming common on servers to have N+1 separate power supplies, where any one of them call fail and the machine will continue to run. Both failing fans and failing power supplies can now often be detected by the computer hardware itself (though in the case of fans, it may be indirectly through temperature monitoring on some equipment.)

Environment

As does USAH, I recommend keeping your machine room around 66 to 68 degrees Fahrenheit (19 or so Celsius) with about 30%-50% humidity. I have seen a large machine room which was being kept around 78 degrees, which I think was too warm.

However, cooling equipment can be a bit worrisome itself; if you have raised flooring, you may come in as I did one day to find that the cooling equipment had vented water down into the raised flooring creating a lake (or at least largish puddles as has occurred in the machine room we had in the first floor of MCH.)

While monitoring temperature is quite common on a lot of computer equipment, as USAH recommends, having a separate temperature monitoring setup is a good idea. USAH mentions a standalone product (Phonetics Sensaphone); another possibility is that other equipment such as PDUs often also have their own environmental monitoring capabilities. (A PDU is a power distribution unit. These are very common in large server rooms.)

Power

In providing power in a server room, many places, including our computer science server room, provide UPS power. Generally, the UPSs used in computer rooms provide power conditioning in addition to emergency power service in the event of power failure. Using such a UPS will provide your computers with a safe source of power, even in the event of brownouts or power surges, and will likely be able to filter line noise (especially important in some industrial scenes.)

Also in providing power, if you have any input as to the design of a new server room, I recommend that you look at putting your rack power provisioning over the racks rather than under, such as with raised flooring. As mentioned previously, raised flooring provides a great place for a new man-made lake, and that is the last thing that you want power cables to be sitting in. Try to get both sides of the racks powered (and if you have the luxury, try to get the power from two different sources, such as from two different UPS units or at least two separate PDUs.) When you have large power requirements coming in, using PDUs to monitor and distribute the power can be advantageous. I have seen a 400-500 server environment that was very successful with two huge UPSs feeding many PDUs that then distributed and monitored the power environment.

Also, check with your electrical experts as to any phase issues that they might perceive.

Security and Unix

As USAH puts it:

Unix was not designed with security in mind, and for that reason no UNIX system can be made truly secure.

While I think that statement is becoming less true with the compartmentalization now available via Xen and with the increased security from using mandatory access control systems such as SELinux, it is still generally true. Even the most security-conscious of all of the mainstream Unixes, OpenBSD, has had its flaws.

The basic flaws are in Unix are

☞ "Unix is optimized for convenience" – not for security.

☞ "Unix security is effectively binary"

☞ Administrative items are outside the kernel, not inside. (For instance, older systems can be found that actually had items such as shells as inside the operating system itself — which I am not sure was any safer than having it outside!)

As USAH has it on page 652:

$$Security = 1/(1.072 * Convenience)$$

General rules of security:

☞ Don't put files on your systems that are interesting to hackers. If you do, protect them cryptographically, such as with an encrypted file system like Loop-AES. Using an encrypted file system that works from a single ordinary file is both convenient for users, convenient for system administrators, and still gives better security.

☞ Keep your machines up to date with patches. While this advice is harder to follow in a production environment where patches may have unintended and unfortunate side-effects, getting behind on security patches is a bad idea. Fortunately, many production environments are isolated by firewalls not only from the world in general, but also from development, q/a, and general user environments – and even maybe running a local firewall also on the server. But some production servers do live closer to the real world, such as mail servers and web servers, and these are two of the most commonly hacked types of servers. Monitoring can also help you find and stop problems.

☞ "Don't provide places for hackers to build nests on your systems." Don't leave world-writable FTP servers running, don't allow poor passwords on machines exposed to the world, don't allow file-sharing services such as Winny to run on your systems.

As Japanese Bring Work Home, Virus Hitches a Ride

By Bruce Wallace, Times Staff Writer
March 21, TOKYO
So far it has spilled military secrets and the private phone numbers of TV stars, airport security access codes and elementary school children's grades.
And the dirty work of this computer virus may not be done.
With almost daily reports of more private information being pumped from personal computers and splashed over the Internet, there is a growing unease that Japan is under insidious attack from within.
The culprit is a digital worm that infects computers using the file-sharing Winny software, a Japanese computer program that, like the infamous Napster, was designed to allow people to easily swap music and movie files.

From the *Los Angeles Times*, March 21st, 2006 at http://www.latimes.com/news/nationworld/world/la-fg-computer21mar21,0,5159274.story

Japanese power plant secrets leaked by virus

Mystery malware and file sharing linked to third breach

By John Leyden
Published Wednesday 17th May 2006 16:06 GMT

Sensitive information about Japanese power plants has leaked online from a virus-infected computer for the second time in less than four months. Data regarding security arrangements at a thermoelectric power plant run by the Chubu Electric Power in Owase, Mie Prefecture in central Japan spilled online this week as a result of an unnamed virus infection, the Japan Times reports.
The name and addresses of security workers, along with other sensitive data including the location of key facilities and operation procedures, found its way onto file-sharing networks. A 40 year-old sub-contractor at the plant who installed the Share file sharing programs on his PC is suspected of provoking the security flap.
The power plant suffered a similar incident in January over data that found its way onto

the Winny file sharing network, the most popular P2P network in Japan, which boasts an estimated 250,000 users. That incident provoked a management edict designed to prohibit the use of file sharing programs, so the occurrence of a similar problem only four months later is doubly embarrassing for Chubu Electric Power.

Chubu Electric is not the only power firm with problems in this area of net security, however. In June 2005, nuclear power plant secrets had been leaked from a PC belonging to an worker at Mitsubishi Electric Plant Engineering, anti-virus firm Sophos notes. That breach, just like the January security flap at Chubu Electric, was also linked to virus infection and the Winny file sharing program.

From the *Register*, May 17th, 2006 at

http://www.theregister.co.uk/2006/05/17/japan_power_plant_virus_leak/

Rules, continued

- ☞ Use an IDS
- ☞ Monitor your tools reports
- ☞ Learn more about security
- ☞ Watch for the unusual, particularly in your logs and /tmp directories.

How is security compromised?

The weakest link is often the human element. Social engineering takes advantage of the fact that people generally are not distrustful, such as demonstrated by Nigerian 419 schemes and by phishing. Education is the only answer, and even then, education is only as good as the most recent attack – the latest scheme may catch even a user wary of previous methods.

Software bugs are the second major category for security compromises. Patching is the main defense as this category.

"Open doors" are the third way. While some software still may have backdoors built-in, it is often the case that people also leave the front door wide open. This is particularly true with items such as wireless routers, which are frequently configured "to just work." As a system administrator responsible for other people's data, preserving both user access and confidentiality for that data, you need to be aware of keeping doors closed.

Specific areas: `/etc/passwd` and `/etc/shadow`

The most important concern is ensuring that users use appropriate passwords. You can (1) suggest that they use good passwords (2) try to enforce that they use good passwords and (3) try to check that they are using good passwords (the program **Crack** by Alec Muffett has been the traditional means for this, but I am not sure how well it is being kept up. I think "John the Ripper" is probably being kept more up-to-date.)

Allowing group accounts is almost always more trouble than trying to keep `/etc/group` up-to-date.

Password ageing: it is generally recommended that passwords be changed on a regular basis, although there has been some back-and-forth discussion on that issue. Also, there is a contingent that are against any multi-use of a single password, with schemes such as **OPIE** that try to fix this problem.

SETUID programs

☞ Be sparing in your use of setuid programs. Don't write scripts that need to be setuid to root; instead, have them run as root, such as with **cron**.

☞ If you really, really need to set up a setuid program, try to have it setuid to a uid > 0.

☞ Have a continuous check for setuid programs appearing on your machines.

File permissions

In times previous, one notorious problem was that of processes that had to look into kernel memory to find information. This was particularly true of programs such as **top** and **ps** (today, we get around this problem by the `/proc` directory.) Devices that referred to kernel memory, such `/dev/kmem` often had inadequate and unsafe permissions, or programs such as **top** had too much privilege for their intended function.

Make sure that no files in `/etc` are publicly writable. There is no good reason for any file in that subdirectory to be writable to the public. No files in directories such as `/usr/bin` or `/usr/lib` needs to be world writable.

Check your device files and make sure that important devices such as disk drives are not world-writable.

How do you do file permission checks?

This is one area where the **find** program shines for one-off checks. While its syntax is somewhat recondite, it can help you discover all sorts of interesting things.

```
find /etc -type f -perm +022 -ls      # check /etc for any files that are
                                      # group or world writable

find /etc -type d -perm +022 -ls      # check /etc for any directories that are
                                      # group or world writable

find /etc -type f -perm +6000 -ls     # check /etc for any files that are setuid
                                      # setgid
```

```
find /etc -mtime -14 -ls                # check to see what files in /etc/ have
                                        # been modified recently
```

Remote Logging

You will find that remote logging is common in larger installations.

Even with standard **syslog**, this is very easy to set up using the "@" syntax. For instance, the syslog.conf man page gives these example lines (slightly modified for clarity):

```
# Kernel messages are first, stored in the kernel
# file, critical messages and higher ones also go
# to another host and to the console
#
kern.*                       /var/log/kernel
kern.crit                     @finlandia
kern.crit                     /dev/console
kern.info;kern.!err           /var/log/kernel-info
```

USAH further suggests physically printing security information on an old line printer to prevent hackers from erasing their tracks. While that is good advice, it is not so easy to find an inexpensive true “line printer” these days, but for most environments, a new dot matrix printer would likely suffice. Continuous sheet printer paper is still available, if a bit more expensive now than stock $81/2$ by 11 paper.

Secure terminals

USAH on page 660 mention secure terminals. However, these are now pretty much archaic; even finding a serial connector on many machines is becoming less common, though not yet rare.

[If you do find such a setup with serial ports, generally the serial ports are connected to a "terminal server" or to a true switch (or even cascade of switches) in order to provide more convenient access to many servers.]

hosts.equiv and ~/.rhosts

In a word "don't"!

These were bad ideas from the time that they originated, and they are unnecessary these days.

Instead, you should use **ssh** keys. This is very easy to do; in fact, you can do as simply as

```
[langley@host1 Slides]$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/langley/.ssh/id_dsa):
Enter passphrase (empty for no passphrase): [ some passphrase ]

Enter same passphrase again:
```

```
Your identification has been saved in /home/langley/.ssh/id_dsa.
Your public key has been saved in /home/langley/.ssh/id_dsa.pub.
The key fingerprint is:
bb:5b:f6:c4:ed:1b:32:74:90:12:30:ab:60:fd:4b:66 langley@host1
[langley@host1 Slides]$ scp ~/.ssh/id_dsa.pub host2:.ssh/authorized_keys2
The authenticity of host 'host1 (128.186.120.121)' can't be established.
RSA key fingerprint is d1:99:0b:9c:b1:ce:87:7d:b7:8e:9a:b5:f1:aa:bc:b9.
Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'host2,128.186.120.121' (RSA) to the list of known hos
langley@host2's password: [ use your password for host2 this time, not your new p
id_dsa.pub                                    100%  615     7.1MB/s   00:00
[langley@host1 Slides]$ ssh host2
Enter passphrase for key '/home/langley/.ssh/id_dsa': [ some passphrase ]
```

So make sure that **telnetd**, **rshd**, and **rlogind** are disabled.

More old programs with bad security implications

Don't run any of these:

- **rexecd**
- **rexd** (on Suns)
- **tftpd**
- **fingerd**

NIS and NIS+

Again, NIS (and its "successor", NIS+) is a bad idea. For the purpose of authentication, better solutions exist such as LDAP accessed via PAM. For the purpose of resolving hostnames into host numbers, DNS is better. For the purpose of sharing other data, simply storing local copies of any reasonably static data is an easy solution.

NFS

NFS was not designed for security, and you should try to limit its deployment within strong firewalls. As USAH suggests on page 662, you should use access lists with fully qualified domain names (or ip numbers.) Squashing ids such as root is a very good idea when you export. When mount, setting “nosuid” is also a good idea.

sendmail

If you really want to do "best practices" with mail security, running **postfix** or **qmail**, which were designed with security in mind is probably "best practice." But if you do run **sendmail** (and there are some benefits to doing so such as access to **libmilter**), then you should stay abreast of any patches that come out for **sendmail**. If you are on a mainline Linux distribution such as RedHat/CentOS, just doing a regular **yum** should be sufficient.

Proactive approaches: nmap

One of the most useful programs for network security is a program called **nmap**. It can scan machines to see what services might be available. It can search large areas of a network for live machines. As its man page says:

```
DESCRIPTION
       Nmap is designed to allow system administrators and curious individuals
       to scan large networks to determine which hosts are up  and  what  ser-
       vices  they  are  offering.   nmap  supports a large number of scanning
       techniques such as: UDP, TCP connect(), TCP SYN (half open), ftp  proxy
       (bounce attack), Reverse-ident, ICMP (ping sweep), FIN, ACK sweep, Xmas
       Tree, SYN sweep, IP Protocol, and Null scan.  See the Scan  Types  sec-
```

tion for more details. nmap also offers a number of advanced features such as remote OS detection via TCP/IP fingerprinting, stealth scanning, dynamic delay and retransmission calculations, parallel scanning, detection of down hosts via parallel pings, decoy scanning, port filtering detection, direct (non-portmapper) RPC scanning, fragmentation scanning, and flexible target and port specification.

Significant effort has been put into decent nmap performance for non-root users. Unfortunately, many critical kernel interfaces (such as raw sockets) require root privileges. nmap should be run as root whenever possible (not setuid root, of course).

The result of running nmap is usually a list of interesting ports on the machine(s) being scanned (if any). Nmap always gives the ports "well known" service name (if any), number, state, and protocol. The state is either open, filtered, or unfiltered. Open means that the target machine will accept() connections on that port. Filtered means that a firewall, filter, or other network obstacle is covering the port and preventing nmap from determining whether the port is open. Unfiltered means that the port is known by nmap to be closed and no

```
firewall/filter  seems to be interfering with nmaps attempts to deter-
mine this.  Unfiltered ports are the common case  and  are  only  shown
when most of the scanned ports are in the filtered state.

Depending  on  options used, nmap may also report the following charac-
teristics of the remote host: OS in use, TCP sequencability,  usernames
running  the  programs  which  have  bound  to each port, the DNS name,
whether the host is a smurf address, and a few others.
```

nmap in its basic mode is quite fast:

```
# nmap diablo

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on diablo.cs.fsu.edu (128.186.120.2):
(The 1595 ports scanned but not shown below are in state: closed)
Port       State       Service
22/tcp     open        ssh
111/tcp    open        sunrpc
139/tcp    open        netbios-ssn
```

```
445/tcp    open        microsoft-ds
515/tcp    open        printer
4000/tcp   open        remoteanything

Nmap run completed -- 1 IP address (1 host up) scanned in 7 seconds
```

Here's what mail.cs.fsu.edu looks like:

```
# nmap mail.cs.fsu.edu

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on mail.cs.fsu.edu (128.186.120.4):
(The 1590 ports scanned but not shown below are in state: closed)
Port       State       Service
22/tcp     open        ssh
25/tcp     open        smtp
80/tcp     open        http
111/tcp    open        sunrpc
143/tcp    open        imap2
652/tcp    open        unknown
```

```
847/tcp    open        unknown
858/tcp    open        unknown
993/tcp    open        imaps
995/tcp    open        pop3s
2049/tcp   open        nfs

Nmap run completed -- 1 IP address (1 host up) scanned in 4 seconds
```

Nessus

Nessus (http://www.nessus.org) has matured a lot since USAH wrote their somewhat dismissive blurb on page 665. In fact, it has gone from being open source to closed source just recently. (There is an open source fork project at http://www.openvas.org).

It is a full featured vulnerability scanner that is very popular for penetration testing. Unlike the older Satan and SAINT programs (though I see that SAINT appears to still be alive as a commercial product), Nessus uses its own X client (based on GTK) rather than a web interface.

tcpwrappers

The program **tcpd** (often called tcpwrappers) has an interesting history vis-a-vis firewalls. While logically firewalls would seem to be a more fundamental idea, **tcpd** actually showed up before good firewalls for Unix machines. The program **tcpd** lets you have reasonably fine-grained control over various ports on your machines, unlike, say, a firewall which basically lets you open or close a port, or the program **stunnel** which lets you (securely) redirect information coming into a port.

[Here's a list of firewall tools that were available circa 1996, when tcpwrappers were quite popular:

- ☞ **fwtk** (TIS firewall tool kit)
- ☞ **gau** (GATEWAY Access Utilities)

As you can see these are largely outdated.

fwtk (http://www.fwtk.org) was overkill for just single server firewall security, supporting proxies and other features. I believe that **gau** was also, though I haven't worked with it.]

COPS

Unfortunately, the program COPS seems to have fallen by the wayside, with no really good replacement out there.

It was useful, if verbose, about reporting many problems.

tripwire

tripwire is still active as open source, though the company http://www.tripwire.com would of course like to sell you their full change management system.

Cryptographic tools

- ☞ Kerberos – used for authentication. While USAH airily dismisses Kerberos with "In our opinion, most sites are better off without it", Kerberos probably deserves a bit more respect. It has been very intensely analyzed, and updates have resulted from such analysis.
- ☞ PGP / GnuPG – used largely for confidentiality (and to instill some confidence) in email via cryptography, it is also used in signing RPM packages. GnuPG (or **gpg**) complies with RFC2440

(http://www.ietf.org/rfc/rfc2440.txt) which codifies interoperability for OpenPGP.

☞ ssh – a replacement for **rsh** and **telnet**. While USAH talks on pp. 672-674 about the morphing of SSH to a commercial product, the open versions **openssh** and **openssl** are actually the more healthy versions and are found now almost universally. However, there has been some recent acrimony from the OpenBSD team about the burden of also being responsible for OpenSSH development; to quote a recent article *"Bigger than OpenBSD, our big contribution is*

OpenSSH," OpenBSD project leader Theo de Raadt told me in a 2004 interview. "It is now included in pretty much every non-Windows operating system made. It is included in network switches, in half of Cisco's products, and who knows where else. It is used by everything from Arrecibo to the Greek Army to who knows where else. And what have we gotten for it in return? Pretty much nothing at all."

☞ OPIE – As mentioned earlier, OPIE is an attempt to eliminate the problem of multiuse passwords, not by the RSA solution of a temporary numeric passwords generated every few seconds by a token, but by using pre-agreed passwords just one time.

Firewalls

In today's normal environment, you should try to run firewalls such as **iptables** on all of your servers.

As USAH puts it on page 677, computer server firewalls are, like the firewall in your car, are not a primary means of defense and should not lull a system administrator into a false sense of security. You should have also firewalls that protect your whole site, and in large sites, it is likely that firewalls should be established between production, q/a, and development. You need to continue using other tools that we have discussed, such as **nessus** and **tripwire** to look for vulnerabilities, both from the outside of your network (aka “outside penetration testing”) and from inside your network’s firewalls.

As we have discussed before, configuring **iptables** is not hard; an example set of rules would look something like:

```
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
```

```
-A RH-Firewall-1-INPUT -p tcp --destination-port 25 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

(Bonus question: what are the magic numbers 50 and 51 in this file?)

Security summary

You can find more about current security issues via

☞ http://www.cert.org – CMU's Computer Emergency Response Team (CERT)

☞ http://www.securityfocus.com – good general site, host of BUGTRAQ and other security-related mailing lists

☞ Full Disclosure mailing list – very, very busy unmoderated mailing list, hosted by Secunia

(http://www.secunia.org)

☞ http://www.sans.org – The System Administration, Networking, and Security Institute (SANS Institute). Good source of classes and the SANS top twenty vulnerabilities.

Some of these give more information, some give less. CERT has seemed to me over the years to give very little detail; for instance, here is their recent summary on the **sendmail** problem:

```
Sendmail Race Condition Vulnerability
Original release date: March 22, 2006
Last revised: --
```

```
Source: US-CERT
Systems Affected

Sendmail versions prior to 8.13.6.
Overview

A race condition in Sendmail may allow a remote attacker to
execute arbitrary code.

I. Description

Sendmail contains a race condition caused by the improper handling of
asynchronous signals. In particular, by forcing the SMTP server to
have an I/O timeout at exactly the correct instant, an attacker may
be able to execute arbitrary code with the privileges of the
Sendmail process.

Details, including statements from affected vendors are available
in the following Vulnerability Note:
```

```
VU#834865 - Sendmail contains a race condition

A race condition in Sendmail may allow a remote attacker to
execute arbitrary code. (CVE-2006-0058)

Please refer to the Sendmail MTA Security Vulnerability Advisory
and the Sendmail version 8.13.6 release page for more information.

II. Impact

A remote, unauthenticated attacker could execute arbitrary code
with the privileges of the Sendmail process. If Sendmail is
running as root, the attacker could take complete control of an
affected system.

III. Solution

Upgrade Sendmail

Sendmail version 8.13.6 has been released to correct this issue.
```

```
In addition to VU#834865, Sendmail 8.13.6 addresses other
security issues and potential weaknesses in the Sendmail code.

Patches to correct this issue in Sendmail versions 8.12.11 and
8.13.5 are also available.

Appendix A. References

    * US-CERT Vulnerability Note VU#834865 -
          <http://www.kb.cert.org/vuls/id/834865>
    * Sendmail version 8.13.6 -
          <http://www.sendmail.org/8.13.6.html>
    * Sendmail MTA Security Vulnerability Advisory -
          <http://www.sendmail.com/company/advisory>
    * Sendmail version 8.12.11 Patch -
          <ftp://ftp.sendmail.org/pub/sendmail/8.12.11.p0>
    * Sendmail version 8.13.5 Patch -
          <ftp://ftp.sendmail.org/pub/sendmail/8.13.5.p0>
    * CVE-2006-0058 -
          <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-0058>
```

Summary of Security, cont'd

Human factors to consider

- ☞ Disgruntled employees
- ☞ Locking the doors on the racks

Watch out for social engineering (think Kevin Mitnick)

- ☞ Information over the telephone – don't answer questions that can reveal security information to unknown people.
- ☞ Information in the wastebasket – shread sensitive

documents.

☞ Information on old media – Encrypt media that are handled by any third party. Destroy old media with sensitive data such as tapes and cdroms, and wipe any disk drives before disposing of them (preferably destroyed also).

Security and obscurity

☞ Good to record all information such as internal DNS, password file on paper also.

☞ Bad to publish anything.

☞ Don't send out to the world your internal naming schemes.

Best practices for rolling out new security patches

- ☞ Order of patching: (1) development first (2) q/a second (3) production last
- ☞ Patch on the weekend, preferably before the weekly reboot to make sure that reboots still work
- ☞ test, test, test
- ☞ Defense in depth: lots of firewalls: separate with firewalls your production, development, and q/a environments. This can prevent some very serious problems such as applications finding development or

q/a databases rather than production ones; it also separates your developers from rolled-out applications, which is best practices.

Preparation: fingerprint your kernel, binaries, and libraries.

☞ Keep the results on media that cannot be changed such burned to a cdrom

☞ The "safest" way to check machines that already export filesystems is to have the "checker" machine itself in a very tight firewall (*no* inbound TCP connections), have it mount the "checkee" drives remotely, then check against the cd material. (Running locally makes tripwire/whatever the most vulnerable part and if you are already exporting filesystems, it may be worth doing. **However, exporting the root and /usr filesystem is not always good security since it opens up critical items that can be attacked, so you might want to to export those filesystems only during the check window. You might want to use** `cron` **to control the windows for when these file systems are exported.**)

Intrusion detection

☞ SNORT – http://www.snort.org ;

- ☞ Should I pay for IDS services? – even if you are running SNORT, it may well be worth paying a company such as ISS or LURHQ to also monitor your environment.
- ☞ Always look out for odd reboots and odd core files

USAH on pp. 680-681 gives these steps for handling an attack:

- ☞ Don't panic
- ☞ Decide on an appropriate level of response
- ☞ Hoard all available tracking information
- ☞ Assess your degree of exposure
- ☞ Pull the plug

- ☞ Devise a recovery plan
- ☞ Communicate the recovery plan
- ☞ Implement the recovery plan
- ☞ Report the incident to authorities

Recovery: Rebuilding your machines?

- ☞ Rebuilding after a problem (compromise, suspected compromise)
- ☞ Preferably, start rebuild over from media, certainly for the operating system