An Internet Security audit for the fish.com computing network

This report is the result of a security audit of the fish.com computer network on the 8th and 9th of April in 1996. The audit was performed by examining the security policy of the network, by taking a snapshot of the state of systems, and by running a collection of analysis programs.

Dan Farmer & Wietse Venema
Vicious Fishes Security Consulting
1685 Oak Street, #202
San Francisco, CA 94117

vicious-fishes@fish.com
http://www.fish.com/vicious-fishes

Graphic Design and Assistance by Clive Grace
http://www.fish.com/~tanais

Table of contents

How to use this Report

The executive summary provides an overall look at the security of the *fish.com* network, stating highlights and a strategy on how to resolve the issues and problems found. The next five sections (Security Policy & Documentation, Security Architecture & Design, Network Security, Host Security, and Physical Security) cover the five main concepts that make up the larger picture of an organization's network security. All the numerous recommendations are summarized in the Recommendations Redux section. Finally the appendices and glossary cover the detailed material and contain support documentation and pointers to other sources of information.

Executive Summary

Introduction

The *fish.com* network is a small network (between 4-6 hosts at any given time) connected to the Internet by a wireless router. Its primary mission is to provide a stable security research platform, a WWW server and development platform, and finally a place for a moderate number of accounts (about 35) to connect and interact with the Internet. Keeping the computer services running is the highest concern, having a secure network is next, and keeping the users happy is the third primary goal.

Fish.com, being the home of some well-known people on the Internet, is often subject to probes or attacks, has been broken into several times, but never had an official security audit before this time. The general security philosophy is to try and keep intruders off of the *fish.com* hosts; if they would succeed in penetrating the defenses, it is assumed that *root* access was going to be obtained. The audit examined all significant facets of the fish.com network that would affect its security level, including host and network security, physical security, documentation (including the security policy), any special or unusual network daemons, home-grown programs, or network services running, and finally the overall security architecture of the network.

Major Problems

There were significant problems found in several different areas of the *fish.com* network. The most serious will take significant effort to repair:

- A poor security policy and a general lack of security guidelines and written documentation
- Weak host security; file permissions, limited security patches, etc.

One very serious but relatively easy to fix problem was found in a CGI script that is accessible via the WWW - it allowed an arbitrary user to execute an arbitrary program (as user *nobody*) on *flying.fish.com*, the main *fish.com* machine.

Additional Problems

While not critical, these problems should be resolved in the near future:

- The *TCP wrappers* are relied on to restrict access from the Internet. However, *anonymous ftp* from the Internet was enabled, which as a side effect gave anyone access from anywhere to the machine (ftp access is equivalent to shell access, in most cases.)
- IP spoofing and session hijacking, two particularly insiduous attacks currently seen in today's Internet, are not defended against.
- Packet screening isn't currently done.

Conclusion

The fish.com network has inconsistant security and falls below its own modest policy standards. Intruders can gain access to the hosts on the network by exploiting holes and problems that are fairly trivial to exploit, mostly left there by careless or sloppy system administration work, which, in turn, was aggravated by a lack of procedural guidelines, security policies, or documentation.

The problems of the site are not insurmountable, but they are not trivial, either; difficulties such as poor documentation, having numerous small problems on the individual hosts, as well as having some significant network issues indicate a general problem with allocating resources to solve basic security issues. The system staff is generally knowledgeable about security, but needs to spend more time on day to day system administrative tasks as well as achieving more consistency and order into the organization of the systems.

We advise the system staff to concentrate on the immediate problems - fixing the few configuration and WWW problems on the network first, and then methodically fixing **all** of the many host-based problems that were found. After the basic problems are resolved, the cisco router's packet screening abilities should be turned on. Finally, the hard part, that of revamping the security policy, updating the documentation, and putting into place procedures and programs that would prevent all of the many problems found from recurring, must be finished.

Security Policies and Documentation

Introduction

It should be clear that the security polices and written documentation of the organization are without a doubt the single most important components of any organization's overall security strategy. A security policy is a document that describes the responsibilities of the organization and each individual user in maintaining the security of the computers and the computer network or system. It not only educates and teaches employees to be responsible users of an organization's computer facilities and resources, but it also empowers system staff to create and maintain a safe, stable, and usable computing environment. In order for a security policy to be effective, it is imperative that it (or the concepts and ideas contained within it) be disseminated to everyone in the organization. In addition, the policy should have sufficient backing from upper management to ensure that people adhere to it and that the consequences of transgressions from the policy be both clearly worded and fairly meted out to offenders.

Ratings

SECURITY POLICIES	✔ ✔
Who can use resources	✔ ✔ ✔ ✔
Proper use	✔ ✔
Granting access & use	✔ ✔ ✔
SA privileges	✔
User rights & respons.	✔ ✔
Sensitive information	✔ ✔
Sys security configigs	

(These range from 0-5 checkmarks, with 5 being best, 0 worst; for more information on this, see Appendix C.)

Summary

The *fish.com* network has a very simple security policy that needs revising (referring to reusable passwords when one-time passwords have been in effect for some time, talking about the older, slower network connection, etc.), is probably illegal (it appears to give the system administrators too much power to audit or monitor the system), does not document any host or network security configurations, and finally no security tools or schedules of audits named.

On the positive side, the policy does inform users of their very basic responsibilities (including no sharing of accounts or passwords, admonishing them not to abuse or look at private files of other users, etc.) and how to contact the system staff in case of emergency.

In sum, the policy is useful to the end user and almost completely worthless to the system administration staff.

Recommendations

While the entire policy could be reworked, if the following items were done it would be an acceptable start towards a good security policy. In order of importance:

- Inventory and codify what the various systems do and give each of them a brief security configuration, stating (at least) what is absolutely necessary or what is absolutely not allowed
- List the security tools used to monitor the system, give a schedule of how often they are to be run, and describe who is responsible for resolving their output
- Give sharper definition as to what the system administrators can and cannot do
- Keep the policy updated, as well as inform users of changes to the policy, so they'll be informed as to the current policy
- List all the users on the host and state the reason why they have an account and what hosts they're allowed to log in from
- Document and map out the network

Specific Critique

Who can use resources

This, like the other sections in the policy, is very sketchy. Having a line like "Using common sense and courtesy should be more than enough on the Internet" is almost an invitation to danger. List all the users on the host and state the reason why they have an account and what hosts they're allowed to log in from.

The policy covers the two really basic rules - users should not share accounts and that system administrators are the only ones who can dictate who uses any resources - but not much else.

Proper use

Visitors with portables should be explicitly required to comply with the *fish.com* usage policy, especially with respect to trust and allowing unauthorized users onto the network.

Even when dealing with friends, it should be made clear what is allowed and what is not. For instance, if the machine is difficult to use or nearly unusable because a user runs a password cracker or some other piggish program, the usage policy currently allows this.

Granting access & use

This item is very clear, except for procedural matters; can outsiders (i.e. people not currently on or friendly to *fish.com* users) apply for accounts? What is the criterium?

SA privileges

The policy almost mentions nothing at all about this, except a brief mention that system and keystroke monitoring might be done if an intrusion is suspected.

User rights & responsibilities

This is one of the best sections in the policy (which isn't saying much); while it could be expanded, it still works as it is. It covers what users should and shouldn't do, how they should behave (in loose terms), what some unacceptable behavior is, what to do when a user's *S/Key* password card is lost, etc.

Sensitive information

The sensitive information on fish.com's network is primarily e-mail from the outside. The policy explicitly tells a user to not look at or disclose other user's e-mail or files without their permission.

System security configurations

The security policy provides no informatin on this subject at all. At a bare minimum, it could say, for instance, that the machines "should run COPS with no warning output unresolved." Ideally filtering rules and detailed system configurations should be covered.

Security Architecture & Design

Introduction

The security architecture & design of a site refers to the overall layout, the security and network strategy and design of the organization or system, rather than any single technical detail or concrete issue or security problem. This includes important issues such as network trust, connections to the outside world, and how these connections interact with the site. All of this is of special interest when examining firewalls and networks; even if the raw technical details seem fine, there can be fundamental problems with the design or with the underlying assumptions of the architect or designer.

Ratings

SECURITY ARCHITECTURE & DESIGN	✔ ✔ ✔ ✔
Basic Design	✔ ✔ ✔
Trust	✔ ✔ ✔ ✔ ✔
External Connectivity	✔ ✔ ✔ ✔ ✔

(These range from 0-5 checkmarks, with 5 being best, 0 worst; for more information on this, see Appendix C.)

Summary

Fish.*com*'s security architecture is relatively simple. The network, and how it interacts with the individual hosts dictates the protection scheme used. Almost no trust is allowed - the Berkeley *r-cmds* are disabled, no NFS is used, windowing systems are at a bare minimum, and there are no live modems or other network connections attached to any machine. One-time passwords (*S/Keys*) are required to log in from the Internet.

Angel.fish.com is a potential problem, however, since it is mostly treated internally as a machine off of the Internet. Even though it doesn't know how to route packets to the Internet, it can be attacked via source routing from any remote host or directly from the other hosts on the LAN. *Bi.fish.com* is mostly a relic; it should be decommissioned or stripped bare, and its DNS service should be moved to *flying.fish.com*. This would help centralize the network services and vulnerabilities.

Recommendations

The fish.com network should:

- Install a packet filter between the LAN and Internet
- Treat *angel.fish.com* as a machine on the Internet
- Move DNS to *flying.fish.com*

Specific Critique

Basic Design

The design is reasonably good. As said before, installing packet filtering and moving the DNS server would be good for the security of the site. Angel.fish.com is too easily accessed from the Internet to be treated as a separate machine.

Trust

Excellent. Aside from trusting sgiblab.sgi.com as a secondary nameserver, there is almost no network trust.

External Connectivity

All the connectivity from the outside world comes from the T-1 via a Karlbridge router, which can be easily monitored and controlled.

Network Security

Introduction

Interest in network security, and the awareness that currently surrounds it, have been growing quickly since the appearance of the Internet Worm in 1988 and the proliferation of firewalls in the early nineties. Eclipsing host security in importance as more and more computers are linked together, it refers to all of the network services that are run by hosts on the network (such as *sendmail*, *ftp*, *telnet*, etc.), trust between hosts, and any filtering or access controlling that is done either at the application level or at the OS/network level.

Ratings

NETWORK SECURITY	✔ ✔ ✔
Anonymous Services	✔
General Network Services	✔ ✔ ✔ ✔
Mail	✔ ✔ ✔ ✔
Name Services	✔ ✔ ✔
Network Access Control	✔ ✔
Security Patches	✔ ✔ ✔ ✔
Trust	✔ ✔ ✔ ✔ ✔
Windowing Systems	✔ ✔ ✔ ✔

(These range from 0-5 checkmarks, with 5 being best, 0 worst; for more information on this, see Appendix C.)

Summary

The LAN communicates with the Internet via a DES encrypted Karlbridge wireless radio router and utilizes Ethernet (UTP) for local communication. There are four hosts, a cisco router, and three small hubs making up the basic network. *Flying.fish.com* and *bi.fish.com* are Sun SPARCstation UNIX machines and are the general user machine and nameserver, respectively. *Tuna.fish.com* is an Intel 486/66 PC running Windows 95. Angel.fish.com is another SPARCstation, but doesn't know how to route packets to the Internet. Network packets are delivered unfiltered to all systems attached to the local network. Human visitors can (and do) bring along their own machines (typically portable PC's or UNIX boxes) and can plug them directly into the *fish.com* network.

In general, network services are fairly secure. The most dangerous services (NFS, the *r-cmds*, etc.) have been disabled, and most unused services aren't activated. Network logins are allowed only from selected sites (this is accomplished with the *TCP wrappers*) and one-time passwords (S/Key) are required for logins from non-local hosts.

Problems do exist, however. The most serious is that *flying.fish.com*, the WWW server, has a CGI script that passes uncensored data to */usr/ucb/mail*, hence allowing external users to execute arbitrary commands (as user *nobody*) on the system. In addition, packet screening and IP spoofing countermeasures are not taken, even though there are two pieces of hardware (the cisco and the Karlbridge router) in addition to the Suns themselves that could easily perform this task.

Recommendations

Network security would be improved if the *fish.com* network would:

- **Fix the flawed httpd CGI script *dforder.pl***
- **Install packet screening and anti IP spoofing measures**

In addition, if the nameservice problems were taken care of:

- **Remove NIS**
- **Change or upgrade the DNS master's version of BIND**

And the ability for users to *ftp* from anywhere on the network were removed, the network security of *fish.com* would be very good. Further recommendations may be found in the critiques of the various areas in the "specific critique" section.

Specific Critique

Anonymous Services

Fish.com network offers web service; the httpd daemon used is from Netscape, and it runs as user *nobody*, but is not *chroot*'d. Since source is unavailable for the server, it is uncertain if any security problems exist (none are known.) However, given the difficulty of writing secure network code, the size of the program, and the quality of the code coming from Netscape (their browser has become the new security bug champion, passing sendmail as the least secure popular program on the Internet), it is almost certain that serious security bugs exist in the code. Whether or not they will be uncovered is yet unknown, however. The server is configured to run CGI scripts that allow remote users to pass input into programs being run by the server; one of these scripts was found to pass unfiltered input into */usr/ucb/mail* (see the **Detailed Critique** below and **appendix C** for more information on this problem); sendmail should be directly invoked to send any mail from a CGI program). CGI programs should be carefully reviewed before being placed onto the Internet.

Anonymous *ftp* is allowed to *flying.fish.com*. The service allows remote users to deliver files in a mode 0722 incoming directory, runs with a 0277 umask, and disallows the *chmod* command. The root *ftp* directory tree is owned by root, and any files uploaded are unreadable and unusable by other anonymous users. No sensitive files (*.rhosts*, *.forward*, etc.) may be placed in a dangerous area. As innocuous as this seems, however, the *ftp* service essentially allows shell access to local users from anywhere on the Internet (a user can execute arbitrary commands by placing an appropriate *.forward* file). The ability for normal users to use ftp from arbitrary sites should either be removed altogether (by either removing anonymous *ftp* or by putting users in the */etc/ftpusers* file) or curtailed (by modifying the source; this would be tricky.)

A security patched version of INN 1.4 is used as the NNTP server; *nnrp.access* allows connections from only a few sites, but they are listed by DNS entry, and access restrictions could be spoofed.

General Network Services

The *finger* service is accessible from any host on the Internet, because it is not wrapped by the TCP wrappers. By doing a network finger bounce through a trusted host (for example, “finger @fish.com@trust-host.com”, a finger request could easily reach the fish network without the *fish.com* administrators ever knowing that the *finger* request was successfully executed.

All non-local network logins onto *flying.fish.com* require an one-time password (using *S/Key*) to access the site. While either of the two accounts with reusable password may login to *bi.fish.com* from anywhere, the TCP wrappers disallow most non-local access.

Most of the really dangerous network services (NFS and the r-cmds) are disabled.

Counter and *telnetx* are unused services on *bi.fish.com* that should be removed.

Multicast is enabled on *flying.fish.com*. While no known security problems exist with the current version, the implementation makes significant changes to the kernel and problems could easily arise. Any newsgroups or mailing lists that discuss this implementation of the protocol should be followed, and any fixes applied as they appear.

Installing the capability to conduct encrypted network sessions to avoid session hijacking and command insertion would be a wise idea (SSH and other programs can do this).

Mail

Only *flying.fish.com*, the mail host, has *sendmail* listening to the SMTP port, and it is the most recent version (8.7.4.1) as of this writing, which is the safest version of this program known (this isn't saying much.)

The latest (again, as of this writing), security patched version of *majordomo* mailing list software is in use; this appears to be a moderately safe mailing list program.

Bi.fish.com has many mail aliases that are unused (the mail serviced migrated to *flying.fish.com*); these aliases should all be removed.

Name Services

NIS is used on all of the user workstations, but it contains no password maps, and the NIS alias map is not used by *sendmail*. Unless necessary, NIS should be removed altogether. Mail and host alias maps do exist and could be compromised; they should be removed if NIS is not. In addition:

- *Ypbind* on *bi.fish.com* and *flying.fish.com* talk to servers anywhere on the Internet, as well as ones talking from unprivileged ports; this should be fixed.

- The *portmapper* and other RPC services allow access from anywhere. This also should be fixed.

As of this writing, the most current version of *BIND* is being used on *flying.fish.com*, which is a caching DNS server. For some reason, however, a slightly older version of DNS is being used on *bi.fish.com*, the primary nameserver. It should be upgraded or *flying.fish.com* should become the new primary server.

There are two flaws in the DNS setup that should be repaired:

- Localhost PTR record in zone 97.174.140.IN-ADDR.ARPA does not belong.
- There is a bad record "fish.com. MX 10 flying.fish.com."; this should be removed.

Network Access Control

No packet filtering is done. This is a fairly serious mistake, and should be enabled. See *ftp://ftp.cert.org/pub/tech_tips/packet_filtering* for more on this, including some suggested ports to filter.

Network logins are disallowed (by the *TCP wrappers)* by default, but enabled on many different Internet sites. This set of sites should be checked to see if all of them are really required to service the users of this site.

Root logins (including via *ftp*) are disallowed over the network.

Security Patches

The only network program that needs security patching is *BIND* on *bi.fish.com*; however, it is a fairly recent version already, and the *fish.com* network relies on DNS very little for security.

Interestingly, there are older Netscape browser binaries (in */usr/local/bin*) that have known security holes in them. They should be either removed or *chmod*'d to 000, because users could execute these and while browsing on the WWW have flying.fish.com compromised because of a bug in the browser. This is also covered in the **Host Security** section.

Trust

The fish.com network doesn't explicitly trust (allow shell access because of any hostname or IP address) any remote host. However, there are two types of trust that do exist, with time and with DNS:

The */etc/resolv.conf* file (used for name resolution) has *ns1.berkeley.edu* in it, and *sgiblab.sgi.com* is a secondary nameserver. Neither of these are of immediate or vital concern, but the less trust the better - if these can be eliminated, they should be.

The fish.com network uses *ntp*; time is synchronized from *apple.com*. The NTP protocol relies on other sites for time - this is in the design. The *fish.com* network should use at least one additional site for time peering.

Windowing Systems

X is the fundamental core of the Sun windowing system. This abomination (as far as security goes) should be deleted from any sensible system, but since flying.fish.com is a user machine, and there are no reasonable alternatives, this will presumably stay. However, be aware that there are significant and tangible risks to running any known complex windowing system.

Host Security

Introduction

Host security refers to the problems that involve individual user authentication and file permissions, the filesystem of the host, system start-up files, system configuration files, and bugs & vulnerabilities that could be exploited by intruders who are actually logged onto the host. While most important when the intruder has access to your system, bugs and vulnerabilities can have unforeseen side effects when combined with network problems (e.g. if the CGI problem that was discussed in the network section were allowed to exploit a bug on the operating system, *root* access would occur), and should be quickly resolved. In most cases (with the exception of operating system bugs) the problem can be corrected with just a few keystrokes.

Ratings

HOST SECURITY	✔
Auditing and Logs	✔ ✔
Authentication	✔ ✔ ✔ ✔ ✔
Filesystem	
Security Patches	
User Security	✔ ✔ ✔ ✔

(These range from 0-5 checkmarks, with 5 being best, 0 worst; for more information on this, see Appendix C.)

Summary

The *fish.com* UNIX hosts (the PC wasn't tested) have inconsistant and fairly poor host security. There are many important files that have poor file permission modes on them, and several security patches should be installed.

On the positive side, the user level authentication methods are excellent, and the individual users' account security is very good.

Recommendations

There are a number of repairs that must be done to the *fish.com* systems to make it even passably secure:

On both *flying.fish.com* and *bi.fish.com*:

- Patch any known security holes (see **Appendix B**)
- Remove or *chmod 000* the old, vulnerable versions of netscape (in */usr/local/bin*) and sendmail (in */usr/lib/sendmail** and */usr/local/bin/mailq*)

On *flying.fish.com:*

- The directory */usr/local/WWW/usage* is owned by user *muffy*, but a cron job writes to a file in this directory as root. The cron job should run as user *muffy*
- *umask* should be set to 022 in the /etc/rc* files, before any programs run
- Several files maintained by the majordomo software package (in */usr/local/majordomo*) are world writable; they should be *chmod*'d to 644
- Many SUID binaries are either insecure or unused; they should be deleted (see the **Specific Critique** section "**Filesystem**" for more details).

Bi.fish.com, the dedicated name server, has so many problems that they can't be concisely listed. Many major security fixes (detailed in the **Specific Critique** section "**Filesystem**") must be installed.

Specific Critique

Auditing and Logs

While reasonably good logs are kept (via the *TCP wrappers* and *sendmail* primarily), they are only rarely examined. A program or a system administrator should examine the audit logs at least once a week, if not more frequently.

Authentication

All non-subnet logins must go through the (excellent) *logaemon* (based on *S/KEY*) one-time password program. Only three accounts (including *root*) even have reusable passwords.

Filesystem

<u>Problems with both *flying.fish.com* and *bi.fish.com*:</u>

The old, vulnerable versions of sendmail still SUID and executable on the *flying.fish.com* and *bi.fish.com* systems (in */usr/lib/sendmail** and */usr/local/bin/mailq*) should be either removed or *chmod'*d to disallow execution (mode 0000 is fine for this); even though they aren't running in daemon mode, they allow a local user to execute them and get root privileges by exploiting their flaws.

<u>On *flying.fish.com:*</u>

Flying.fish.com uses a cron job to maintain WWW statistics. The *cron* job to do this runs as *root* but writes to a file in a user owned directory (*/usr/local/WWW/usage*). Using symbolic links the user can case *root* to overwrite any file on the host. The statistics directory should be writable only to the user that owns it and the cron job should be run as that user.

The umask should be set to 022 in the */etc/rc** files, before any programs run. Several files on the system have bad file permissions set because the default *umask* causes world writable files to be created.

Many files maintained by the *majordomo* software package (in */usr/local/majordomo*) are world writable. Local users can both break into the *majordomo* account and execute commands as *daemon* by exploiting this problem. The files should be *chmod'*d to 644; the insecure file mode is a side-effect of the insecure *umask* setting, as discussed above.

Nnmaster should be SUID *news*, not SUID root! The ownership of the database files should be *chown*'d to *news* as well.

Mail2news runs as SUID majordom; the SUID bit isn't needed and should be removed.

Autoreply is SUID root, but is not being used (it was last used August of 1995) - delete it or remove the SUID bit.

Ntop is SUID root; it doesn't even run on this platform! Delete it.

There are older Netscape browser binaries (in */usr/local/bin*) that have known security holes in them. They should be either removed or *chmod*'d to 000.

<u>On *bi.fish.com:*</u>

Old and vulnerable (CERT advisory VB:95-04) logdaemon binaries should be replaced; the vulnerable version gives users write access to the */etc/skeykeys* database.

Global write access to */etc/utmp* should be removed; a number of bugs (*comsat*, *dump*, *write*, *syslogd*, etc.) can be exploited using this problem.

Updatedb runs as root and reveals hidden file names.

/dev/nit should be mode 600, not 640.

The *unused /usr/games/lib/nethackdir/nethack* is SUID root; it should not be.

/usr/etc/mount and */usr/etc/umount* are SUID root; they can be used by unprivileged users to execute arbitrary shell commands as *root* and should not be SUID anything.

Security Patches

Several security patches should be installed on all of the systems - Appendix B has a complete list. In addition, however, binaries of old, potentially dangerous executables (sendmail, netscape, etc.) have been left on the system, rather than remove or *chmod* them to 000; leaving the programs executable makes the patch effectively useless, and care should be taken to insure that this doesn't continue to happen.

User Security

There are no accounts without a password on any of the UNIX systems, all accounts have non-trivial passwords (most require *S/Key*), and unused accounts are blocked (a "*" password and a non-executable shell) or removed altogether (including system accounts). The only significant problem is that there are too many accounts (there are thirty-seven total) on the systems - many are either completely dormant or almost never utilized; the number of accounts should be reduced.

Physical Security

Physical security is one of the most overlooked aspects when considering the overall computer security of an organization - nothing is typically done beyond installing a simple badging system or having cheap locks that provide little to no security. However, a physical security audit should not merely cover locks and other physical access controls, but it should also examine contingency plans and equipment for dealing with physical disasters (fire, flood, electrical storms or failures, etc.), the physical layout and construction of the organization's buildings and machine rooms, the positioning and access to network cabling, access and potential for abuse of ventilation ducts, floor and ceiling tiles, etc. The construction, installation, and even auditing of all of these factors can be very costly; it is typically only done for extremely high security sites such as found in military or government systems. However, physical security is an important component of the overall computer security of an organization (especially considering that most computer security problems happen on-site by employees) and time should be allocated to audit it.

Ratings

PHYSICAL SECURITY	✔ ✔
Physical Access	✔ ✔
Disaster Resistance & Recovery	✔ ✔
Physical Authentication	✔ ✔ ✔ ✔
Backups	✔

(These range from 0-5 checkmarks, with 5 being best, 0 worst; for more information on this, see Appendix C.)

Summary:

Fish.com resides on the second floor of a building in a fairly typical San Franciscan condominium. A large number of people (visiting friends and acquaintances), both regular users and not, can get physical access to the machines, as can a house cleaning person who comes every other week. It would be fairly simple to gain access to the machines by approaching via the backyard (up the fire escape), the roof (the building has two adjacent buildings that give roof access), or from one of the neighboring condominiums; the back sliding glass doors are often left unlocked. The machine rooms (i.e. bedrooms and living room) all have heat activated water sprinklers, and a dry chemical ABC fire extinguisher is easily accessible from any room. Simple surge protectors are on all the computers, but there is no UPS. There is a large amount of fur (from the four cats) and dust on and in all of the machines. Backups are done sporadically, not organized very well, and are kept on shelves scattered about the house.

Recommendations:

Since this is a home, steps that might be normally suggested to improve physical security can't be practically suggested. However, there are several fairly simple measures that could be taken that would increase the physical security situation without changing the atmosphere to something oppressive or unlivable:

- **All the computers should run a screen locking program**
- **Keep the backdoors locked**
- **Put backups in a locked fireproof box**
- **Save old backups off-site**
- **Keep the cats from sleeping on the monitors**

Specific Critique:

Physical Access

Screen locking. All the computers should run a program that locks access to the computer when the computer is idle for more than a few minutes.

Keep the backdoors locked. Although breaking into a house is fairly simple to do, forcing an intruder to physically break something would be a useful deterrent.

Disaster Resistance & Recovery

Fire is the main concern (it is difficult to safeguard the network against earthquakes); the sprinkling system and fire extinguisher is adequate defense against most small fires. However, the lack of off-site backups could be a serious problem if the computers were physically damaged or destroyed. Lightning, hair driers, and vacuum cleaners cause occasional power outages; a UPS would certainly be of use, but fairly expensive for all of the computers for the benefits it would impart. The cat fur and dust has no simple solution; the cats seem determined to use the warm computers for resting places despite draconian disciplinary measures. Keeping the computers in a closed room is impractical, and training the cats seems hopeless. Frequent cleaning of the computers or shaving of the cats is recommended.

Physical Authentication

Even though formal methods, such as badges or card readers aren't used, no one enters the home unless they are known to the owners or are vouched to by trusted friends (except the odd pizza deliver person, who doesn't get a chance to access the machines.) This is an effective method in most circumstances, but when parties are held, many people are often there of unknown character or quality.

Backups

A regularly scheduled backup regime is really necessary. In addition, putting the backups in a locked fireproof box and saving the old backups off-site (even at someone else's home or at work) would be a big plus.

Recommendations Redux

This is a summary of all of the recommendations in the previous sections of the report; the most serious are listed first.

Serious problems to be resolved (by section)

Policies & Documentation

- Inventory and codify what the various systems do and give each of them a brief security configuration, stating (at least) what is absolutely necessary or what is absolutely not allowed
- Both list the security tools used to monitor the system and give a schedule of how often they are to be run and who is responsible for resolving their output
- Give sharper definition as to what the system administrators can and cannot do
- Keep the policy updated, as well as inform users of changes to the policy, so they'll be informed as to the current policy
- List all the users on the host and state the reason why they have an account and what hosts they're allowed to log in from
- Document and map out the network

Architecture

- Install a packet filter between the LAN and Internet
- Treat *angel.fish.com* as a machine on the Internet
- Move DNS to *flying.fish.com*

Networking

- Fix the flawed httpd CGI script *dforder.pl*
- Install packet screening and anti IP spoofing measures
- Remove NIS
- Change or upgrade the DNS master's version of BIND
- Remove arbitrary ftp access from anywhere for *fish.com* users

<u>Host</u>

On both *flying.fish.com* and *bi.fish.com*:

- Install the patches listed in **Appendix B**
- Remove or *chmod 000* the old, vulnerable versions of *sendmail* and *netscape*

On *flying.fish.com:*

- The cron job that runs the WWW statistics packages should be run by user muffy instead of *root*, since */usr/local/WWW/usage* is owned by user *muffy*.
- *umask* should be set to 022 in the /etc/rc* files, before any programs run
- Several files maintained by the majordomo software package (in */usr/local/majordomo*) should be *chmod*'d to 644
- Many SUID binaries are either insecure or unused; they should be deleted (see the Host **Specific Critique** section "**Filesystem**" for more details)

On *Bi.fish.com*:

This host has so many problems that they can't be concisely listed here; look at the details in the Host section, **Specific Critique,** under "**Filesystem**")

<u>Physical Security</u>

- All the computers should run a screen lock program
- Keep the backdoors locked
- Put backups in a locked fireproof box
- Save old backups off-site
- Keep the cats from sleeping on the monitors

Non-critical security issues to be resolved (by section)

<u>Policy Issues</u>

- Visitor's with portables should be explicitly required to comply with the *fish.com* usage policy
- Even when dealing with friends, it should be made very clear what is permissible behavior and what is not
- Clarify account availability issues; can outsiders apply for accounts? What are the criteria?

Networking

- Remove *Counter* and *telnetx* services on *bi.fish.com*
- Monitor multicast news and security patches
- Remove mail aliases on *bi.fish.com*
- Make *Ypbind* and the *portmapper* on *bi.fish.com* and *flying.fish.com* only talk to servers from the local net and using privileged ports
- Upgrade the version of BIND on *bi.fish.com*
- Localhost PTR record in zone 97.174.140.IN-ADDR.ARPA does not belong.
- There is a bad DNS record "fish.com. MX 10 flying.fish.com."; this should be removed.
- Audit the hosts that are allowed to connect via the *TCP wrappers*

Host

- A program should be written to examine the logs automatically, or a person designated to do this regularly

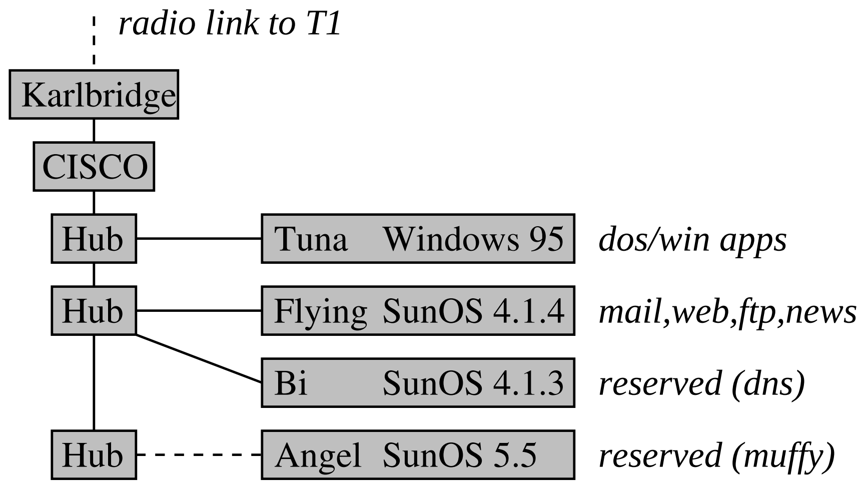
On *flying.fish.com:*

- *Nnmaster* should be SUID news, not SUID root
- Remove the SUID bit on *Mail2news*
- Delete *autoreply* or remove the SUID bit on it
- Delete *Ntop*
- Delete the old Netscape browser binaries

On *bi.fish.com:*

- Remove the old logdaemon binaries
- Chmod 644 */etc/utmp*
- Chmod 600 */dev/nit*
- Remove the SUID bits on */usr/etc/mount* and */usr/etc/umount*
- Make */usr/games/lib/nethackdir/nethack* SUID games, not root
- Delete dormant or unused user accounts

Appendix A
Map of the fish.com Network



Portables frequently get plugged into the ethernet, but the following hosts are permanently on the ethernet:

- Bladerunner.fish.com. Cisco 4000 router, running 10.3. Not currently doing any filtering of traffic.
- Tuna.fish.com. An Intel 486/66 running Windows 95.
- Flying.fish.com. The main user machine. SPARCstation 20, dual CPU's, running SunOS 4.1.4 with multicasting support.
- Bi.fish.com. DNS master. SPARCstation 2 running SunOS 4.1.3.
- Angel.fish.com. User machine not directly on the Internet. SPARCstation 5 running Solaris 2.5.

Appendix B
Security Patches Needed

The security tool *Fuck* (alpha release) was used to determine that the following vendor patches should be installed on *bi.fish.com*:

Flying.fish.com should install the following SunOS patches (listed by patch ID number):

- 100448-03
- 100452-72
- 100478-01
- 102436-02
- 102545-04

Bi.fish.com should install:

- 100448-03

Angel.fish.com should install:

- 103279-01
- 103300-02

Appendix C
Audit Methods, Tools & Data

<u>Checkmarks:</u>

The checkmarks that are used to rate subsections range from zero to five checkmarks, with five being the best rating you can get and zero checkmarks meaning that the system totally failed the test. An aggregate rating for the section is calculated by summing up all the points in the subsections and dividing that total by the number of subsections. Point values are:

(zero checks)	-4
✔	1
✔ ✔	2
✔ ✔ ✔	3
✔ ✔ ✔ ✔	4
✔ ✔ ✔ ✔ ✔	5

<u>CGI script vulnerablity details:</u>

The CGI script *cgi-bin/dforder.pl*, called from *cgi-bin/dftotal.pl* pass unfiltered data into */usr/ucb/mail*, and gives remote users the ability to execute arbitrary shell commands as user *nobody*. To exploit this (say, to send the flying.fish.com password file to you), send queries with parameter values that contain:

%OA (newline)

Followed by "*~!cat</etc/passwd*" (or whatever command you'd like to execute.)

<u>List of auditing programs used:</u>

These were the tools that we used to audit the *fish.com* network (in addition to the tools given by SunOS):

- Crack
- Tiger
- COPS
- Fuck
- SATAN

Appendix D
Selected Bibliography

Books and Articles:

Chapman, Brent, *Network (in)security through IP packet filtering*. In Proceedings of the Third Usenix UNIX Security Symposium, pages 63-76, Baltimore, MD, September 1992. Available from *ftp://ftp.greatcircle.com/pub/firewalls/pkt_filtering.ps.Z*.

Cheswick, William R., and Bellovin, Steven M., *Firewalls and Internet Security*. Addison-Wesley, Reading, MA, 1994.

D. Farmer &W. Venema, *How to Improve the Security of Your Site by Breaking Into It*. Posted to netnews in 1993. Available from *ftp://ftp.win.tue.nl/pub/security/admin-guide-to-cracking.101.Z*.

Holbrook, Paul, and Reynolds, Joyce, editors. *Site Security Handbook*. RFC 1244 July 1991.

Simson Garfinkle & Eugene Spafford, *Practical UNIX Security*. O'Reilly & Associates, Sebastopol, CA, 1991.

URL's of Interest:

Don't go to any of these sites if you want to see anything visually appealing or stimulating (most security people seem to lack in artistic imagination), but these are some of the definitive repositories for security information on the Internet:

ftp://cert.org. Staid, dull, and behind the times, this is the home of CERT/CC, who nonetheless put out the definitive security advisory bulletin; they have archives of many tools and advisories, but should get a life.

http://coast.cs.purdue.edu. COAST is the research laboratory/playground of Gene Spafford of Purdue. COAST has an extensive collection of security papers and tools on-line.

ftp://ftp.win.tue.nl/pub/security. Wietse Venema's personal site. Not as comprehensive as the others, it nonetheless is important since it contains the latest versions of all of his papers and tools.

Mailing lists:

Firewalls. Send mail to *firewalls-request@greatcircle.com* (*firewalls* uses the *majordomo* mailing list program).

CERT. Send mail to *cert-request@cert.org* to receive the CERT advisories.

CIAC. Send mail to *ciac-listproc@llnl.gov* to receive the CIAC advisories.

Best of Security. Send mail to *majordomo@suburbia.net* to receive this mailing list.

8LGM. Send mail to *majordomo@8lgm.org* to receive the (sporadic) 8LGM advisories.

Newsgroups to watch:

Netnews is typically dreadful in quality, but sometimes you see very useful things that you can't easily find anywhere else. These are all worth scanning from time to time:

alt.security
comp.security.UNIX
comp.security.firewalls
comp.security.misc
sci.crypt

In addition, the main source code *newsgroups (comp.source.UNIX, comp.source.misc, and alt.sources*) should be watched.

Archive locations for important auditing software:

COPS	*http://www.fish.com/security/cops*
Crack	*ftp://ftp.win.tue.nl/pub/security/*
ipsend (part of the IP Filter package)	*ftp://coombs.an.edu.au/pub/net/kernel/ip-fil3.0.4.tar.gz*
lsof	*ftp://coast.cs.purdue.edu/pub/tools/lsof*
SATAN	*ftp://ftp.win.tue.nl/pub/security/satan.tar.Z*
Tiger	*ftp://net.tamu.edu/pub/security/TAMU*

Glossary

CGI. Common Gateway Interface. Used by Web servers to allow users to place poorly written services on the Internet. A real security hazard if used; httpd daemons should be in *chroot*'d jails if CGI programs are allowed.

Firewall. Simply speaking, a network device that sits between two potentially hostile networks, in an effort to prevent dangerous traffic from passing from one side to the other.

IP spoofing and hijacking sessions. Both are attacks gaining popularity on the Internet. The former allows someone to impersonate an IP address by spoofing sequence numbers so that network trust can be abused. The latter is a method of actually taking over or injecting commands into an existing TCP connection. Both are fairly trivial operations, given the availability of tools.

Packet screen. A method of blocking network packets, based on protocol, source address, destination address, or port numbers.

r-cmds. The supremely abusable Berkeley UNIX commands, which include rsh, rcp, rlogin, and rcp. They all use name services (DNS, NIS, etc.) for authentication, can allow users to log in without supplying a password, and are almost always bad news.

Trust. When a host allows another host to remotely utilize a service without providing authentication.