Global Information Assurance Certification Paper

Unix System Security Audit

SANS GIAC Unix Security

EXECUTIVE SUMMARY

The Audit was conducted to evaluate and identify significant weaknesses in the configuration of the servers ABC (the corporate firewall), server DEF (Corporate Informational Web Server) and sever XYZ (Internal corporate fileserver for users). A number of vulnerabilities were detected. Since the corporate environment at this segment is not perceived as a 'high risk' entity, the level of security required by the network here is not very high. Accordingly, the audit ignores the low vulnerability findings and concentrates on the higher risk issues, which should be addressed as soon as possible.

The Audit involved discussion with the System Administrators and the review of configuration files and settings on each server, and consisted of two major sections:

- Host based analyses: through the evaluation of the configuration of each server.
- Network and Internet Vulnerability analysis: to evaluate what services each server offers, and whether these can be exploited to gain access to the system.

The results of the Audit are summarized below:

- Due to the absence of an overall Corporate Security Policy, the administration of the system is inconsistent and the strength of the security measures is weak and fragmented.
- Users have excessive access to the system. There is no procedure defined for reviewing user access. Further, users have access to system level commands and functions which may be misused.
- Encryption and Authentication methods and standards used to access the system need to be strengthened. Using a password-cracking tool, about 40% of the user passwords were cracked within 10 minutes. Users access the system through inherently weak communication protocols and services like the Unix 'r' services. These should be augmented through the use of software like SSH.
- A number of services and system accounts have been enabled on the servers audited. These should be reviewed, and unnecessary services and accounts should be removed from the server.
- An external scan of the network and the servers was conducted to simulate the actions that would be undertaken by an attack. During this, a large amount of information was found to be available on the network and the servers, which can be used by an intruder to launch a successful attack.
- A number of critical files and directories were found have permissions that allow anyone to access them. This can result in someone tampering with data on the server and the loss of data. Writeable directories also allow intruders to introduce 'Trojan Horses' that can give them complete control over the machine.
- Although a number of logs are being generated, they are not being monitored on a regular basis to identify suspicious activity. This would facilitate preventing attacks or identifying intrusions at an early stage to minimize damage.

Unix System Security Audit

SANS GIAC Unix Security

Audit Methodology

This audit was conducted towards completion of requirements of the SANS Institute's GIAC Level 2 Unix Security Analyst Certification. For reviewing the system security, the system configuration and operation was discussed with the System Administrator, and the following configuration files were requested:

- /etc/passwd
- /etc/sendmail.cf
- at.allow
- at.deny
- /etc/cron.d/cron.allow
- /etc/cron.d/cron.deny
- /etc/shadow
- /etc/system
- /etc/default/login
- /etc/default/passwd
- /etc/profile
- /etc/group
- /etc/shells
- /etc/dfs/dfstab
- /etc/export
- /etc/inetd.conf
- /etc/services
- /etc/hosts.lpd
- /etc/ftpusers
- /etc/hosts.equiv
- /etc/hosts.allow
- /etc/hosts.deny
- /etc/default/su

The following commands were executed on each server and the output analyzed:

- /usr/bin/rpcinfo –p
- /usr/sbin/patchadd –p
- ls –la /dev/*
- ls –la /etc
- ls –la /usr/bin
- ls –la /usr/sbin
- ls –la /var/yp
- ls –la /etc/netgroup
- ls –la /etc/aliases
- ls –la /usr/spool/mqueue
- ls –la /etc/sendmail.cf
- ls –la /etc/hosts.lpd
- ls –la /etc/logindevperm
- ls –la /var/adm/sulog
- ls –la /var/adm/loginlog
- env
- eeprom
- /usr/lib/nis/nisstat
- ypcat passwd
- whereis sudo
- mount
- netstat –an
- xhost
- ps -auxww
- telnet localhost
- telnet localhost 25
- telnet localhost 79
- ftp localhost
- /usr/bin/find / -local -type f -name '.rhosts' -exec ls -al {} \; -exec cat {} \; 2 (.rhosts)
- /usr/bin/find / -local -type f -user root -perm -4000 -exec ls -dal {} \; 2 (SUID files)
- /usr/bin/find / -local -type f -user root -perm -2000 -exec ls -dal {} \; 2 (SGID files)
- find /\(-local –o –prune\) -perm –000002 –print
- find /name .netrc -printfind / -perm –1000

Further to the host-based evaluation, a network-based evaluation was conducted using CyberCop Scanner ™ , WSPingPackPro Scanner ™ and nmap ™.

Unix System Security Audit
SANS GIAC Unix Security
Technology Summary

The audited network system consists of the following components:

Server ABC:

Operating System: Solaris 2.6
Application: Checkpoint Firewall 1
Purpose: Gateway Firewall between the Internet and the internal corporate networks and mail proxy server
Location: Network Perimeter.

Server DEF:

Operating System: Solaris 2.6
Application: Apache
Purpose: Corporate Informational Web Server
Location: DMZ

Server: XYZ:

Operating System: Solaris 2.6
Application: Fileserver and Administration
Purpose: User Home directories, Application and administrative server
Location: Internal to corporate network

The selection was made from different zones of the corporate network since the needs and security level at each zone is different:
Sever ABC, being the firewall needs to be extremely secure, with a high level of access control and auditing and monitoring. Server DEF is a public information machine and being in the DMZ has a lower security profile. Server XYZ is used by "trusted" employees, and controls must be effectively placed to prevent deliberate or accidental security breaches. However, the degree of public exposure due to a security breach on XYZ is lower than for say the firewall.

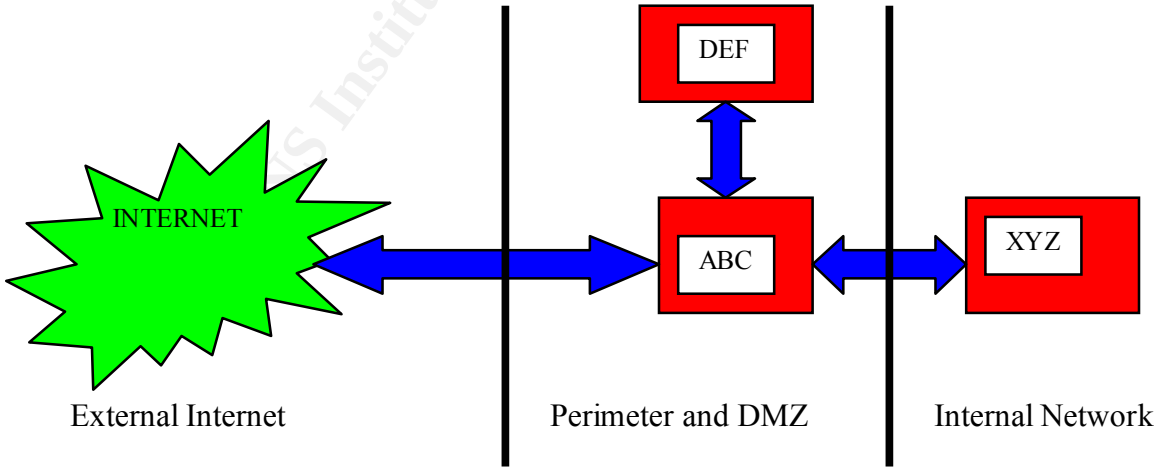

Figure: Network Map

Unix System Security Audit
SANS GIAC Unix Security

AUDIT REPORT

General Administration

Corporate Information Technology Policy
Audit Method
A copy of the existing corporate guidelines for installing, configuring and maintaining Unix systems was requested from the System Administrator. Further, the administration of the Network was discussed.

Finding
Currently, although there are a number of generally followed procedures for various tasks, there is no formal documented and distributed Corporate Security Policy, with specific configuration guidelines for Unix systems and resident applications.

Security Implication
Lack of a policy raises the following security risks:

- Inconsistent application of security features and configuration across the network may leave hosts vulnerable to attack.
- The policy is dependent on the System Administrator, and may not be optimal.
- Auditing and monitoring procedures are difficult to design and implement.
- Difficulty in communicating corporate security standards to new employees and administrators.

Risk Level: High

Recommendation

- A comprehensive Security Policy should be developed, with specific sections relating to detailed Unix configuration for each application in the environment.
- A general security policy should be communicated to all existing and newly hired staff.
- Administrators should be provided with configuration guidelines.

System Administration

Operating System Accounts
Audit Method
The */etc/passwd* and */etc/shadow* files were analyzed for active accounts.

Finding
The Solaris default configuration sets up a number of system level accounts, each of which is associated with specific privileges and functionality, e.g. *bin, daemon* etc. On all the servers audited, a number of these accounts were found active. Also, although some of the accounts have been disabled, the method of locking the account is not consistent.

Security Implication
Since these accounts are a standard part of the installation, their names and privilege levels are well known. An attacker would usually try to gain access to the system through these accounts. Further, not all the system accounts are required to be active on each server. An unnecessary account may be exploited without being detected. Also, unnecessary accounts are often associated with outdated binaries and packages that are not upgraded as rigorously since they are not in use.

Risk Level: Medium

Recommendation
- Remove all unnecessary accounts.
- Account removal should be consistent. Preferably, lockout the password, and set the account login shell to *null.*
- Remove disabled accounts from all group memberships.

Root access and Switch User (su) command
Audit Method
System administration procedures were discussed with the Administrator. A copy of the *sulog* was analyzed to determine which users had access to root. The command *whereis sudo* was run to determine if *sudo i*s in use. The *PATH* variable and *env* command output was collected for each server.

Finding
- System Administrators typically log into the system as root.
- Further, privileged users use the *su* command to escalate privilege for administrative purpose using *su.* For this, a number of 'power' users share the root password.
- Output of the *PATH* and *env* command did not have the current working directory as the starting point for a path search for a command (this is not an issue).

Security Implication
- Logging into the system directly as root by a number of people leaves no audit trail. Audit trails are very useful for determining responsibility for actions taken on the system, and also possibly for any legal action.
- When a user gives a command, the system tries to locate the command based on the PATH variable, starting from the current user location in the file system. Using just the *su* command without defining the specific path for the binary, or having a PATH variable with a '.' in the beginning (implying execute from current directory) leaves a system vulnerable to Trojan Horse inserted by an intruder. Such a Trojan horse program may disclose the root password to the intruder, or cause other damage to the system.
- Sharing the root password means that a large number of employees potentially have complete control over the server. This access may be more than what is specifically

required by the user, e.g., an operator may need access to only the backup and restore commands for his job, and does not really need to have complete access.

Risk Level: High

Recommendation

- Users should not login as root. System Administrators should login to normal unprivileged user accounts and then *su* to root.
- The PATH variable for all users should be set globally so that commands are not executed from the present directory by default.
- The root password should be limited to a small number of users. Wherever there is a need for providing users with higher privilege for their job function, utilities like *sudo* should be evaluated to provide limited and controlled access.

User Administration

Audit Method

- Review of */etc/passwd, /etc/default/login, etc/default/passwd,/etc/group, /etc/shadow, ypcat passwd, /etc/profile, /etc/shells*
- Discussion with System Administrator on user set-up and removal procedures.

Finding

- There is no standard for ensuring strong passwords. On server XYZ, using *crack* about 40% of the passwords (162/400) were cracked within 10 minutes.
- There is no defined user name standard across the organization. Further, usernames on all the systems reviewed are inconsistent, e.g. user John Doe-->jdoe; user Jane Doe--> janed; user Joe Blow --> johnblow. Were found on server XYZ.
- On server XYZ, user access levels are defined per the initial access request from the respective Department Supervisor. However, there is no procedure for periodically reviewing the access level and its appropriateness.
- A number of shells like *csh* are available to users, and are defined as their default login shell. However, the valid shells for the servers are not defined in */etc/shells*.

Security Implication

A consistent user naming convention makes it easier to locate a given user's access and facilitates auditing.

User access often changes with time, as people change their job functions. If not reviewed periodically, a user may have legacy access that is excessive or not relevant or necessary to their job function. This access may be exploited by a user to gain access beyond what they are currently required to do, leading to a system compromise.

Weak passwords, especially in globally visible authentication databases like NIS make it trivial for a user to 'crack' other user's passwords and gain unauthorized access. Also, weak passwords are extremely easy for external intruders to crack using brute force attacks.

If all the valid system shells are not defined in */etc/passwd* any user on the system can use any file or application as a valid shell. This new 'shell' can give the user or intruder

access to commands that would normally be denied, e.g., an intruder could gain root access by executing a special 'shell' tailored to exploit the system.

Risk Level: High

Recommendation

- Users should be educated on the need for using strong passwords and ways of generating them, e.g., the first letter of each word in a phrase they can easily remember. Passwords should be between 6-8 characters long, have a 90 day limit, and should not either be re-used on the same or other systems. The System Administrator should run a password cracker like *crack* periodically to verify the strength of user passwords, and enforce the change of weak passwords.
- Develop a consistent username format across the organization, e.g., first character of first name, and first 7 characters of the last name.
- All valid shells should be listed in */etc/shells*.
- User access should be reviewed on a periodic basis and updated. Department supervisors should inform the System Administrator of all changes to their organization. User access should be limited to the applications they need access to, using restricted shells or a *chroot* () functionality.

World-Writeable and SUID/SGID Files

Audit Method

Find commands were executed on the servers ABC, DEF and XYZ to locate all files with world-writeable permissions and SUID/SGID permissions. The output was redirected to appropriate files for later analysis.

Finding

A large number of world-writeable and SUID/SGID files were found on the server XYZ. Further, a number of files in the /usr, /opt and /var directories allow all users to have write permission.

Security Implication

- World-writeable files allow any user or an intruder to change the contents of a file, effecting information integrity. Also, for executable files, an intruder may replace the file with a trojan horse that can damage the system and its integrity.
- SUID/SGID files execute with the privilege of the owner/group. These can be subverted by an unauthorized user or intruder to escalate their privilege to those of the owner/group of the SUID/SGID file.

Risk Level: High

Recommendation

- Review all world-writeable and SUID/SGID files on the system. Using freeware tools like fix-modes or YASSP can facilitate identifying and correcting the permissions on files. After the review, create a list of all the remaining "approved" World-writeable and SUID/SGID files on the system and store in a secure place.

Periodically, check the system against this list to identify changes and ensure that such changes are approved.

- NFS shared files, especially files in /usr, /opt and /var should be exported 'read-only to specific hosts. Further, through */etc/vfstab,* the exported file systems (except special cases like */tmp, /dev* and /) should be mounted with the *nosuid* option to prevent the inadvertent granting of SUID privilege on NFS mounted files.

Job Scheduling

Audit Method

Access permissions to the *crontab* command and permissions over the */etc/cron.d* directory and its contents were reviewed. The configuration settings in the files */etc/cron.d/cron.allow, /etc/cron.d/cron.deny, /etc/cron.d/at.allow* and */etc/cron.d/at.deny* were reviewed for access control settings.

Finding

- On the server ABC, DEF and XYZ, users have access to the *crontab* command and the ability to schedule jobs.
- The allow or deny files have been not created on any server to define access control settings.

Security Implication

The *crontab* command and the *at* and *cron* are an integral part of the Solaris operating system for job scheduling. Access to these is restricted through file permissions on the *crontab* command or through the allow and deny configuration files listed above. In the absence of restrictions, any user can schedule jobs. These jobs may be unauthorized, leading to network disruption and unauthorized or excessive access to system resources, to the extent of data loss and system shutdown. An intruder can also exploit this for unauthorized activity.

Security Risk: Medium

Recommendation

- The number of users allowed to schedule jobs should be restricted and minimal. Jobs should be scheduled through the System Administrators after appropriate approval from the Department Supervisor.
- Access control over job scheduling should be consistently implemented through the allow and deny files, with a "default deny" stance.

For configuring the allow and deny files:

- Set ALL: ALL in the deny file, thereby denying access to all users.
- Then add the necessary users and specific machines to the allow file, e.g. allow joe1 machine2

Unix System Security Audit
SANS GIAC Unix Security

Unix Service Administration

General Server Services

Audit Method

- Review of */etc/inetd.conf* for enabled services
- Review of */etc/sendmail.cf* for sendmail configuration
- Review of *ps-auxww* for active service daemons and processes
- Review of */etc/ftpusers* for users with ftp access
- Manual review of the servers for dial-up/RAS devices
- CyberCop, nmap and WSPingPackPro reports

Finding

- The server XYZ allows administrators to dial-up through a modem for System Administration. This modem is usually left on, although the dial-up number and access is limited to 3 administrators only.
- On server XYZ, a number of services were found active and enabled, viz. *ftp, telnet, finger, uucp, sendmail* and *smtp*. On server DEF, *ftp, finger and telnet* were found active.
- The */etc/ftpusers* file has not been created on any of the servers ABC, DEF and XYZ.

Security Implication

Each of these services is vulnerable to being exploited by an intruder or unauthorized person to gain/escalate access to the system. Finger service on the externally visible server DEF allows an external person to query the system to provide user name and login information. *ftp* and *telnet* are services that allow people to login to the system and have a number of known vulnerabilities. *uucp* is a legacy protocol for transferring files over serial lines, which can be exploited through buffer overflows as well as to provide a means for an intruder to access the system. Sendmail is the system mail routing program. It runs with root privileges and successful attacks on this service usually gives the intruder complete access to the system as root. An intruder who is able to exploit any of these services can now access the system and cause considerable damage in terms of loss of data and confidentiality, and system shutdown.

Risk Level

Server ABC: *High*
Server DEF: *High*
Server XYZ: *Medium* (Internal server, protected from external attack by the firewall)

Recommendation

- The need for each service should be reviewed on each server, and all unnecessary services should either be disabled by one of the following:
  - Commenting out the service in */etc/inetd.conf* so that the service is not started at boot
  - Removing the unnecessary system binaries

- In case it is necessary to have the services:
  - Use a port-based access control method by implementing *tcp-wrappers* to specify which users and machines have access to a given service on a given machine
  - For sendmail, if the server is not a mail server, disable the service. The sendmail service may be required by some applications, and running sendmail periodically through cron and then shutting down the service can satisfy this need.
- Modem dial-up access should be limited. Preferably, the modem should be activated locally each time it is needed and then shutdown. Further, all remote access, especially for administrative purposes should use some end-to-end encryption and strong authentication technology like SSH.
- *ftp* access should be reviewed and denied to privileged users like root, by specifying the denied users in */etc/ftpusers*.

Unix 'r' Services

Audit Method

- Review of */etc/inetd.conf* for enabled services
- Review of *find* command for *.rhost, .xhost* and *.netrc* files
- Review of *ps -auxww* for active service daemons and processes
- Review of */etc/hosts.equiv, /etc/hosts.allow* and */etc/hosts.deny* files
- CyberCop, nmap and WSPingPackPro reports

Finding

- On server XYZ, 'r' services like *rlogin* and *rsh* were found enabled and active.
- The */etc/hosts.allow* and */etc/hosts.allow* files have not been defined on any of the servers ABC, DEF or XYZ.
- An explicit "Trust relationship" has been defined between servers XYZ and the development server XYZDev through */etc/hosts.equiv* for all users on XYZDev.
- A number of *.rhosts* and *.netrc* files have been defined for trust relationships for users with other machines in the internal network.

Security Implication

The 'r' services provide access to the system with weak authentication. The services have historically been vulnerable to a number of well-known exploits like IP Spoofing, password sniffing, traffic interception etc. that are used to give an intruder complete access and control over the system. Trust relationships through *.rhosts* and *.netrc* allow a user to access other machines without explicit authentication, and may contain the user's password in clear-text in the configuration file. A person, on a given machine, to gain unauthorized access to other machines in the network can use these relationships. Trust relationships are also a prime target for intruders to escalate access to sensitive machines and resources on the network.

Risk Level: *High*

Recommendation

- The need for the 'r' services should be reviewed, and disabled wherever not necessary. Alternate and more secure methods of providing the same functions to a user should be incorporated (e.g., *telnet* with SSH instead of *rlogin*).
- In case the 'r' services are absolutely necessary, then they should be strengthened through the use of encryption and authentication applications like SSH or SecureRPC (using UnixAuth etc.).
- Users should not be allowed to establish trust relationships. The system should be swept for *.rhosts* and *.netrc* files daily, and users should be educated about the security risks associated with their use. Global trust relationships defined through */etc/hosts.equiv* should be qualified and restricted to specific hosts as follows:
  - Deny access to all hosts through */etc/hosts.deny* and then,
  - Define specific hosts and users that can use the trust relationship in */etc/hosts.allow*

Network Information Service (NIS) And Network File Service (NFS)

Audit Method

- For NFS, review */etc/dfs/dfstab, /etc/export*
- For NIS, review of *ypcat passwd, /etc/netgroup, /var/yp, /usr/lib/nis/nisstat*
- Discussion with the System Administrator.

Finding

NIS is used in the organization for resource and domain control. NFS is used to provide transparent file access to users over the network.

- Root level passwords are defined in the NIS map.
- The *ypbind* daemon is not configured to run in secure mode.
- A number of the exported NFS files allow any user read and write access.

Security Implication

- The NIS password file is not shadowed. Due to this, the encrypted password hashes are globally visible, and anyone can download a copy of the same. Using commonly available password cracking tools like *crack*, this password file can be decrypted to reveal the password of every user defined on the system in the NIS password file, including root.
- Further, *ypbind* is the client based service that lets a NIS client bind to the *ypserv* service on a NIS server, for information. In the secure mode, a client can only bind to an NIS server on a reserved, trusted port. In the absence of this security, an intruder can start an NIS server on the network, and clients that bind to it will get false information, leading to a loss of integrity and thence a potential compromise of the network.
- Exporting NFS file systems with global access permissions allows any user on the system to access and make changes to the file, leading to loss of data and the confidentiality of sensitive corporate information.

Risk Level: High

Recommendation

- NIS should be replaced with NIS+ which provides better security features like shadowing
- In case NIS is necessary for the organization and cannot be replaced, NIS should be run in secure mode by configuring with the *–s* option
- Root level passwords should be hard-coded locally on each client and should not be part of the NIS map
- NFS shared files, especially files in /usr, /opt and /var should be exported 'read-only to specific hosts. Further, through */etc/vfstab,* the exported file systems (except special cases like */tmp, /dev* and */*) should be mounted with the *nosuid* option to prevent the inadvertent granting of SUID privilege on NFS mounted files.

Network Services and Access Management

Audit Method

- WSPingPackPro scan of the servers to determine visible ports and services over the Internet
- CyberCop Scan of the servers under review
- Nmap scan of the servers
- Review of *telnet* and *ftp* commands (refer command list on page 2) on different ports on each server

Finding

- The firewall configuration on ABC allows an intruder to scan the internal network and gather information using ICMP pings and scans. Further, the DNS Zone transfer and IQUERY functionality is not secure.
- Users at the off-site location access the server XYZ through the firewall using the Internet. Access is attained by the user dialing up to a local ISP, from where a 'pseudo VPN' application provides secure access. Access is provided through services like *telnet, ftp* and the 'r' services.
- Server ABC and XYZ display identifying banners on ports 21, 23 and 25, e.g., banner on machine XYZ identifies the machines a Solaris 2.6 server running sendmail version 8.8.3.

Security Implication

- Gathering information on a network is the first step in initiating a successful attack, since it helps an attacker understand the network, identify IP addresses and hostnames, operating systems on each server and services offered and identify sensitive targets for an attack.
- Although a pseudo VPN is used to provide a secure connection between the user at a remote location and the server XYZ , the traffic is not encrypted till it reaches the VPN channel at the ISP. This phase of the traffic therefore is open to interception, hijacking and spoofing etc.
- Identifying banners provide information on the server operating system, service version etc. These help an attacker identify the system and focus on exploits that can

be used against the specific system, making it much easier to successfully break into the system very quickly.

Risk Level: High

Recommendation

- The network and firewall should be configured to provide minimal information. The lack of information makes it more difficult and therefore a deterrent for an attacker.

Some measures that should be implemented are:

  - Disallow all ICMP traffic leaving the internal network, to disable ICMP ping sweeps
  - Use a split-horizon DNS to provide minimal information on internal servers to the external world.

- Instead of a pseudo VPN, the organization should evaluate the use of an end-to-end encryption method like SSH. This provides greater security, and will also be cheaper since the organization will no longer need to pay for the services of the pseudo VPN service.
- Identifying banners should be removed and replaced with generic banners, which warn intruders against unauthorized access.

Auditing, Monitoring and Disaster Recovery

Audit Method

Discussion with the System Administrators regarding procedures for auditing and monitoring of the network, as well as any disaster recovery plans.

Finding

- Although a number of logs are generated by each server locally, there is no formal process for reviewing the logs periodically
- System backups are taken on tape every night, and are stored off-site. The tapes are rotated per a well-defined procedure every 6 weeks.
- An alternate disaster recovery site has been set up at a branch office of the organization in a neighboring city about 50 miles away. Per leading practices, the disaster recovery plan is adequate, and this is therefore not an issue.

Security Implication

In the absence of a periodic review of the logs, unauthorized activity may go undetected for a very long time until substantial damage has already been done to the system.

Risk Level: Medium

Recommendation

- The following logs should be created and maintained on each server:
  - /var/log/wtmp : login/logout history

- /var/log/btmp: unsuccessful login history
- /var/log/messages: syslog messages
- /var/log/secure: access and authentication log
- /var/adm/loginlog
- /var/adm/sulog : su attempts log

♦ The audit logs should be reviewed periodically, preferably every day. To facilitate the task, various freely available tools like *swatch* or *logcheck* can be used to identify traffic and access patterns, and flag suspicious activity before it escalates into an intrusion.

♦ Further, server logs should be forwarded to a centralized logging server where the logs should be maintained securely.

Unix System Security Audit
SANS GIAC Unix Security
References

- Unix System Administration Handbook,
  Nemeth et al, Prentice Hall, ISBN 0-13-151051-7
- Building Internet Firewall,
  Chapman and Zwicky, O'Reilly and Associates, ISBN 1-56592-124-0
- Nmap Documentation,
  http://www.insecure.org
- Hacking Exposed- Network Security Secrets and Solutions,
  Stuart McClure et al, Osborne, ISBN 0-07-212127-0
- Solaris System Administrator's Guide,
  Janice Winsor, Ziff-Davis Press, ISBN 1-56276-080-7
- Solaris Advanced System Administrator's Guide,
  Janice Winsor, Ziff-Davis Press, ISBN 1-56276-131-5
- Practical Unix and Internet Security,
  Garfinkel and Spafford, O'Reilly and Associates, ISBN 1-56592-148-8
- Unix Unleashed,
  Robin Burk and Salim M Douba, MacMillan Publishing Company,
  ISBN: 0672314118
- Managing NFS and NIS,
  Hal Stern and Mike Loukidaes, O'Reilly & Associates, ISBN: 0937175757
- SANS GIAC Unix Security Course Material