Global Information Assurance Certification Paper

Apache and Solaris Security Considerations
For the SANS GIAC Certification
Version 2.1

Table of Contents

Apache and Solaris Security Considerations

Introduction

This paper can be used to establish policy based on security considerations for running an Apache Web Server on a Solaris (SPARC) platform. Although the operating system and Web Server are specific, the platform hardening concepts can be applied to support other UNIX based platforms and applications. These procedures would apply to all system administrators and Web Server administrators developing secure systems on a DMZ or internal network. This document is broken into two sections: Solaris security and Apache configuration.

Before I start there is a few important points that should be made. Make sure you properly develop and enforce a corporate security user policy. A network is only as good as its policy. If users are allowed access to network resources despite their duties, then logging and event monitoring becomes moot. The user policy must not only grant access to necessary resources, but the minimal access required to complete one's job.

Check your ARP cache. Always check your Web Servers ARP Cache. You should be aware of layer 2 network attacks and watch for repeated address resolution replies, and changing MAC address requests/IP pairs.

Patches and Fixes. Always keep on top of your Patches and Fixes. To reduce your percentage of vulnerabilities associated with your platform and Web Server, you must proactively keep ahead of vendor patches and monitor various security alerts. One of many sites http://securiteam.com is a favorite. Their mailing list is available in two formats: plain text and HTML. While the information is the same, they recommend the HTML list, if you have an HTML capable e-mail client. The average traffic is about 2-3 advisories a day. For your convenience, the advisories contain a special 'tag' in the subject line that identifies the article type. The possible values are:

> [NEWS] - General security news – A resource for new and popular white papers.
> [NT] -Windows specific issues – Covering Microsoft and third party applications.
> [UNIX] -UNIX specific issues – Covering multiple UNIX flavors and apps.
> [EXPL] - Advisories that contain exploit code - Includes virri and whitepapers.
> [TOOL] - Free tool reviews – Some tools created by white and black hat hackers.
> [REVS] - Commercial product reviews – Tested by SecureITeam themselves.

When Apache patches pertain to a minor bug or two, or features which they haven't yet included in a new release, they will place them in their patches subdirectory so people can gain access to it before they roll another complete release. Unofficial patches (things they are not yet sure about including) are in the contributed patches directory.

> Security related information for Sun and Apache products and vendor patches:
>
> Sun http://sunsolve.sun.com/ Apache: http://httpd.apache.org/dist/httpd/patches/

Apache and Solaris Security Considerations

Solaris Security

Help from Sun. Sun computers offer a free tool called The Solaris Security Toolkit, which automatically secures, minimizes, and hardens Solaris Operating Environment systems. Also referred to as the JumpStart Architecture and Security Scripts (JASS) toolkit, it can help simplify Solaris operating system distribution and deployment. It provides a Solaris administrator with an easy method of hardening Solaris systems. It will also help ensure that a Solaris system is secure before an administrator places it on a public network, which is essential to proper deployment. It can be used for installation of identical hardened Solaris OE images to multiple systems, customizing the hardening process to a company's standards. This toolkit may be found at http://www.sun.com/security/jass

Remove unnecessary services. Offer only essential network services and operating system services on the server host machine. Ideally, each network service should be on a dedicated, single-purpose host. Many computers are configured by default to provide a wider set of services and applications than required to provide a particular network service, so you may need to configure the server to eliminate or disable them. The inetd system deamon starts many common network services. The inetd.conf file will help you determine which services are needed and may be commented out in front of the entry. Due to the lower overhead of assigned services you will gain an increase in speed and manageability. A list of suggested Internet services may be found at http://www.cert.org/security-improvement/implementations/i012.01.html

> Actively running services may open ports to that listen for other services that can be running on both the server and the network. **Port scanning tools** can be used to determine which ports are accepting connections on a host. Usually services run on standard ports, so this can be used to determine which services are running also. The attacker simply tries to connect to every possible port on the target system, usually with a script or custom program. Afterwards, the attacker knows which ports are accepting connections, and can use this information to launch further attacks. Sometimes the order is randomized to try to reduce detection.

Remove Unnecessary Ports. Eliminate any unnecessary open network port. Eliminate unnecessary TCP and UDP network ports on which a server process may listen for incoming client connections. This reduces the risk of attack using these ports. Open network ports can be identified using the netstat command on UNIX. The /etc/services file can be used to edit well known ports defined by the Internet Engineering Task Force (IETF). The **Port Numbers** registry can be found at the following URL: http://www.iana.org/assignments/**port-numbers**

One example of the dangers of open service ports and probably the most popular would be **FTP Port Bounce Scanning**. Normally in an FTP connection, a client opens a connection to the control port of the FTP sever on port 21. This connection is used to send commands, file listings, and other information between the server and client. When files are actually transferred, a second connection has to be opened between the server and client, called the data connection, which usually happens on the higher numbered ports. In order to make this second data connection, the client sends a PORT command to the server machine. This command tells the server which IP address to connect to and which port to open at that address, and the server opens that connection. Under normal operation, the IP address that is sent is the client IP address, and the port is a high numbered port. In an FTP bounce scan, the attacker instead sends a series of these PORT commands, using the target system's IP, and a series of port numbers. The attacker can then use the connection acceptation or refusal to determine which ports are open on the target system.

Protect Checksum Binaries. Protect your servers Checksum Binaries with IDS. Whenever possible, programs such as Tripwire should be utilized to maintain at least the "best industry standard" of IDS protection.

File Permissions. Always utilize the most restrictive file permissions. Whenever possible the he most restrictive level of file permissions should be used to protect root access and system directories. Backed by an appropriate Security Policy, root and file permission should be properly allocated to the appropriate system administrators and back-up operators.

Good Password Standards. Password groups should be maintained in accordance with your Issue-specific password management system/policy. Group accounts should not be used for authentication and each user/group account should have a unique username and password to sign on. Shadow password should be utilized whenever possible.

Programs such as John the Ripper and Lophtcrack utilize what is called "dictionary attack" methods when scrying passwords from a system. The best way to protect against them is with a strong password policy. A good policy, backed with preventive software, would not authorize passwords that:

- Contain less than six characters
- Consist of words found in a dictionary (English or foreign)
- Are of common usage such as: Names of family, pets, friends, co-workers, etc.
- Consists of computer terms and names, commands, sites, companies, hardware, software.
- Include names of cities and places as in "newyork", "southbeach" or any derivation.
- Include personal information such as birthdays, addresses and phone numbers.

- Include word or number patterns like bbbaaa, wysiwyg, jhlafgu, 098890, etc.
- Include any of the above spelled backwards.
- Include any of the above preceded or followed by a digit (e.g., passwd1, 1passwd)

Remove unnecessary accounts. The /etc/passwd file contains files called userdel uucp; userdel nuucp that you may want to delete for centralized account administration. Also consider changing the default null shell for the daemon, bin, sys, adm, lp, smtp, listen, noaccess, nobody4 and nobody accounts to /bin/false. Cert.org suggests this command 'usermod -s /bin/false daemon' can be used as an example.

> Remember the more accounts you have to administer; the more chances they can be cracked. The administration of system and user accounts can play a debilitating role in network security if not managed properly. In some cases it only takes a disgruntled system administrator or even a back-up operator to capitalize on holes within your policies.

Apache Security Issues:

Compiling Apache consists of three steps. First detail the responsibility that your Web Server will perform, and then define the steps you will need to get there. Selecting which Apache modules you want to include into the server is very important at this point. Next, you will want to create a configuration for your operating system. Then compile the executable. Compiling the minimum necessary modules needed for the required function of the Web Server will help develop an efficient machine that's also easy to administrate and troubleshoot. Not doing so could result in unnecessary services that may be difficult to remove after installation and add additional resources to CPU utilization.

Once again, review the services installed by the default and the removal of unnecessary services within the Internet daemon services or Inetd. Network services are primarily generated from the Internet daemon and the run control scripts as the system goes through its run levels during boot-up. Many of these services have inherent vulnerabilities associated with them and are normally not required for the operation of a Web Server. Located in the /etc/inetd.conf file, the Internet daemon intercepts the request and hands it to the appropriate server when a service is requested from another host. Comment out unnecessary services within the inetd.conf in front of the service to disable them.

Apache and Solaris Security Considerations

Apache Web Served Access: The .htaccess and http.conf file for host-based access/authentication files should be removed from the root directory (and possibly a different drive with separate security acl's). Consider configuring the .htaccess file which resides within the particular directory you are limiting access to or the httpd.conf file. **General Security Guidelines for an Apache Web Server on Solaris by** Scott Tieman suggests the following recommendations for consideration since, each have their unique pros and con's.

> **Htaccess - Pro**: Server does not have to be re-started after updating the file. Also, if you need to relocate directories within your server, your .htaccess file containing your access controls will move with them. **Con**: No centralized location. Might have to access multiple directories

The ability to update and manage files is a strong consideration for utilizing this method. Workarounds and fixes for a vulnerable server or for just updating file access can be done without interruption to an "in use" production server. However, for large installations that span multiple directories across multiple servers, a non-centrally managed access configuration can make access/authentication management very difficult.

> **Httpd.conf - Pro**: All access controls reside in centralized location. **Con**: Each time the file is updated, the server must be re-started before it takes effect.

In large Enterprise environments, this would be the suggested method of management. It may be true that you will have to restart the server prior to affect a change. This is where change management protocols come into place. This will allowing for proper departments to prepare critical systems to halt development and updates until changes are made to production equipment. This is necessary to keep conflicts from arising during the development and maintenance cycle.

Note: Regardless of the location, the syntax/content remains the same. Host-Based access controls are normally performed through container directives. The following are the most popular ones utilized which control access to for specific HTTP methods, directory and file access that resides within your HTML pages.

<Limit> Restricts directives that are contained within a particular HTTP method such as "get" or "put". These methods are contained within the HTTP request header from the client (browser) to the server (Web Server). Get, informs the server that the client is requesting to retrieve information, such as a document or a HTML page. Put, informs the server to store something, such as a file.

Host Based Authentication: General Security Guidelines for an Apache Web Server on Solaris by Scott Tieman says authentication refers to login name and password entries to gain access to a specific location on the server. Apache utilizes the <AuthType>, <AuthName>, and <AuthUserFile> directives.

> <AuthType> Defines the authentication mechanism the server will use. Basic = Utilized the most. However, information is passed in the clear and is susceptible to sniffing unless utilized with SSL.

Digest = Uses a Message Digest (MD5) for authentication.

<AuthName> Defines the label that is passed to the Authentication Directive. Also appears on the login prompt as well.

<AuthUserFile> Defines the absolute path where the authorized users and their passwords are located.

Apache Event Logging: Apache generates monitorable event logs which are valuable assets for tracking server access and potential server problems. By default, Apache generates two main log files. The **General Security Guidelines for an Apache Web Server on Solaris by** Scott Tieman tips describes them as:

Access_log: Tracks all HTTP connections by IP and provides a time stamp. The server access log records all requests processed by the server.

Error_log – Tracks server related events such as re-starts or general server problems.

The server error log, whose name and location is set by the Errorlog directive, is the most important log file. This is the place where Apache httpd will send diagnostic information and record any errors that it encounters in processing requests. It is the first place to look when a problem occurs with starting the server or with the operation of the server, since it will often contain details of what went wrong and how to fix it.

Users with access to the same directory that Apache is writing it's log files to can almost certainly gain access to the uid that the service is started as, which is usually root. Do not give users write access to where the directory logs are stored in without understanding the consequences.

In addition, log files may contain information supplied directly by the client, without escaping. Therefore, it is possible for malicious clients to insert control-characters in the log files, so care must be taken in dealing with raw logs.

It's important to understand the importance of log files in the ongoing attempt to stop malicious network activity. Try to determine the level of security within your own environment and back-up log files according to industry standards and enterprise policies. There are many different ways to maintain a level of non-repudiation, such as outputting logs to a separate off server directory, a printer, or a third party SQL database application.

Permissions on Server Root Directories are the first consideration on securing any Web Server after a default configuration. In typical operation, Apache is started by the root user, and it switches to the user defined by the User directive to serve hits. As is the case with any command that root executes, you must take care that it is protected from modification by non-root users. Not only must the files themselves be writeable only by root, but so must the directories, and parents of all directories.

If you allow non-root users to modify any files that root either executes or writes on then you open your system to root compromises. For example, someone could replace the httpd binary so that the next time you start it, it will execute some arbitrary code. If the logs directory is writeable (by a non-root user), someone could replace a log file with a symlink to some other system file and then root might overwrite that file with arbitrary data. Caution must be exercised, if these log files are writeable (by a non-root user), then someone may be able to overwrite the log itself with false data.

One aspect of any configured Web Server, which is occasionally misunderstood, is the feature of default access. That is, unless you take steps to change it, if the server can find its way to a file through normal URL mapping rules, it can serve it to clients. When configuring your server pay particular attention to the interactions of `<Location>` and `<Directory>` directives; for instance, even if `<Directory />` denies access, a `<Location />` directive might overturn it.

Also be wary of miss-configurations with the http://httpd.apache.org/docs/mod/userdir directive; setting it to something like `"./"` would have the same effect as root.

> Another UserDir issue concerns an exploit that was posted by SecureITeam on September 15, 2001 called **Apache UserDir Information Disclosure.** It seems an information leak can occur on Apache based Web Servers. The leak occurs whenever the UserDir module is enabled, and it would allow an external attacker to enumerate existing user accounts by trying to access their home directory and monitoring the response. Note that users do not have to have public_html directories for this attack to work. One of the workarounds involved changing "UserDir public_html" to "UserDir disabled". More information can be found at http://www.securiteam.com/unixfocus/5WP0C1F5FI.html

Server Side Includes (SSI) may present a server administrator with several potential security risks. The first risk is the increased load on the server. All SSI-enabled files have to be parsed by Apache, whether or not there are any SSI directives included within the files. While this load increase is minor, in a shared server environment it can become significant.

> In general SSI files pose the same associated risks that are found with CGI scripts. Apache.org explains that using the "exec cmd" element, SSI-enabled files can execute any CGI script or program under the permissions of the user and group Apache runs as, as configured in httpd.conf.
>
> Caution should be exercised when enabling SSI for files with .html or .htm extensions. The risks are maximized within shared or high traffic, server environments. Consider enabling SSI-enabled files with a separate extension, such as the conventional shtml. This helps keep server load at a minimum and allows for better management of associated risk.

Many system administrators simply disable the ability to run scripts and programs from SSI pages. To do this, Apache.org suggests replacing Includes with IncludesNOEXEC in the Options directive. Note that users may still use <--#include virtual="..." --> to execute CGI scripts if these scripts are in directories designated by a ScriptAlias directive. More information can be garnered at http://httpd.apache.org/docs/misc/security_tips.html#protectserverfiles.

Essentially, that there must be a level of trust between the servers' security and the CGI programmers. You and your CGI script/program writers need the ability to spot potential security holes in CGI scripts and programs both deliberate and accidental. This also includes third party applications and scripts. Remember, all CGI scripts will run as the user, so they have the potential to conflict with other scripts. One program which can be used to allow scripts to run as different users is suEXEC which is included with Apache as of 1.2 and is called from special hooks within the Apache server code which can be found at http://httpd.apache.org/docs/suexec.html. Consider using CGIWrap for your applications. It can be found at http://wwwcgi.umr.edu/~cgiwrap/.

Hack your server. Before you consider a server to be properly configured is to test it. The best way to test the security of your configurations is to "test" your configurations. Since passwords and ports are the most vulnerable use programs like Nessus, Crack, Netstat, and NMAP can help identify configuration errors before they become security vulnerabilities.

Nessus is a remote security scanner that will remotely audit a given network, and determine whether it is vulnerable to common security threats. This free product is updated twice a month and it's modular architecture is very fast and has an exhaustive list of security checks. It can be downloaded from http://www.nessus.org

Crack is another freeware program that is used for UNIX systems, used especially for breaking encrypted passwords. Although, it is by hackers for remotely undermining passwords on UNIX systems, it can also be used to determine the password strength of server accounts. It is an invaluable tool for testing password policy. It can be found at http://www.itprodownloads.com/ using it's free search engine.

Netstat is a network utility that comes standard with many UNIX and Windows NT installations and is used by most administrators to review open ports and services on any given installation. The command netstat –a will display all TCP/UDP connections running with their status as either "listen" or "idle".

NMAP is an extremely versatile remote port scanner with many options and may be downloaded from http://www.nmap.org. The current Beta Version 2.54 includes uptime scanning; TCP timestamp and IP.ID sequence scanning (for advanced attacks and OS detection). At a minimum you should run the following commands**. General Security Guidelines for an Apache Web Server on Solaris by** Scott Tieman suggests that at minimum these commands should be used to test your system.

./nmap –sT <IP> This command will scan TCP ports.

./nmap –sU <IP> This command will scan UDP ports.

./nmap –sR <IP> This command will scan for rpc ports.

Finally…

Everything should begin with proper policy. A network with the proper security policy foundation would set the framework for a standard server build describing the steps described above to help ensure proper security. System and Issue Policies for server and user administration, acceptable usage and back-up policies, help ensure that the work completed is managed by those with the appropriate work instructions, and level of authority. A free list of security policy example templates may be found at http://www.sans.org/newlook/resources/policies/policies.htm. This site as sans.org puts it: "…offers a primer for those new to policy development and specific guidance on policies related to legal requirements such as the HIPAA guidelines." What's nice about this site is that it is a "work in progress and the policy templates will be living documents", which means as changes in technology changes so will the templates. This is much better than many pay sites that charge for dated documents.

References

1. The Sun Dot.Com Builder web site:

http://dcb.sun.com/practices/howtos/securesite.jsp

2. The Apache Web Server Security Tips:

http://httpd.apache.org/docs/misc/security_tips.html - serverroot

3. Apache Web Server Patches

http://httpd.apache.org/dist/httpd/patches/

4. Carnegie Mellon CERT Co-ordination Center Web site:

http://www.cert.org/security-improvement/implementations/i012.01.html

5. Scott Tieman for the SANS Institute

General Security Guidelines for an Apache Web Server on Solaris

http://rr.sans.org/unix/unix_list.php

6. SANS Security Policy Templates

http://www.sans.org/newlook/resources/policies/policies.htm

7. Tool research and update web site

http://www.securiteam.com/

8. NMAP program & help documentation

http://www.insecure.org/nmap/

9. Crack program download

http://www.itprodownloads.com/

10. Nessus Security Scanner and documentation

http://www.nessus.org

11. Tripwire

http://www.tripwire.com