Global Information Assurance Certification Paper

The Process of Hardening Linux
Chris Koutras

Introduction

The classic project cycle consisting of a study period followed by the acquisition period and ending with the operations period can be adapted or tailored to just about any task. The process can be quite involved for major projects such as building the space shuttle. However, the process can easily be tailored to smaller projects such as building a secure Linux server. By following a methodical process of evaluating the user needs, using these needs to determine the system configuration, documenting the system and regularly maintaining the system, we can create a secure product in a minimum of time. The Linux operating system has numerous settings and permissions which provide high degree of customization but this same feature also makes it a challenge to secure properly. Add to this the nature of the open source software environment where anybody can create program for this operating system and you end up with an infinite number of possible configurations. The goal is to use these properties to create a secure system that meets the users needs. There are a number of excellent resources available guides to securing the system and test for vulnerabilities. We will incorporate these system security resources into our process and leave system vulnerabilities testing, which can be a whole paper in and of themselves, for another time.

There are a number scenarios which can lead to not securing a system properly such as; "The head of a large department wants that new web server up ASAP", "It is only a test system - there is no valuable data on it", or an old favorite, "The IP address is not published, so no one will know it is on the Internet". Each of these scenarios can be managed in a way that creates the time required to secure the system properly. Remind the department head that lost proprietary or customer data from a compromised system can do irreparable damage to the companies image and bottom line. Bear in mind that test system has a connection to the Internet, spare CPU cycles and little or no security - it would make a great zombie for a distributed denial of service attack. It has been proven time and again, security through obscurity does not work on the Internet - script kiddies use automated tools to scan large blocks of IP addresses looking for vulnerable machines. If machine is vulnerable and connected to the Internet, it is only a matter of time before someone finds and exploits it. It seems the old adage "There is always time to do something twice but never time to do it right the first time" also applies to Linux security, "There is always time to rebuild a compromised system but never time to secure it properly the first time".

There are a number of vulnerabilities in most Linux systems and it seems the vulnerability reports will continue to increase. A review of the Bugtraq [1] vulnerability database confirms an alarming trend in the reporting of Linux vulnerabilities. From 1997 to 1998 the number of reported vulnerabilities doubled from 12 to 24. The following year, 1999, they more than quadrupled to 99. The first eleven months of 2000 showed an increase of 31 to 130. A review of the vulnerabilities reported to Internet Security Systems X-Force [2] database shows incomplete data for 1997 and 1998. However the data for 1999 reveals 98 reported vulnerabilities and 149 for 2000. These numbers closely track each other and the increase in popularity of Linux as a server operating system. Nineteen ninety-nine was a banner year for Linux as several Linux oriented ventures went public and there was an all out public relations campaign to extol the virtues of this free open source operating system. This increase in public exposure and use of Linux appears to have also brought about an increase in vulnerability reports. The Linux community and distribution vendors have begun to take security seriously and this is paying off in terms of increased frequency of security related patches and reduced vulnerability discoveries.

One of the benefits of open source software is the source code is available for bug fixes and improvements. Unfortunately, this can also work to the hackers advantage since they can also study the source code and look for vulnerabilities. Another area to pay particular attention to is the acquisition of your Linux binaries. They should be downloaded from the web site of or purchased on CD-Rom from a reputable supplier such as www.redhat.com , www.mandrakesoft.com or www.linuxcentral.com. The MD-5 or PGP signatures should always be verified if they are available. Downloading or purchasing binaries from unknown sites could lead to installing software with a trojan horse, backdoor or virus. Downloading the source code and compiling the binaries yourself is not always a guarantee of getting unmodified code, you should take the same precautions when acquiring source code as when acquiring binaries. Some Linux proponents argue since the source code is freely available for examination this makes open source more secure. This is true if you have the time and expertise to examine and interpret hundreds, if not thousands, of lines of code to ensure the it does not contain dangerous routines or exploitable flaws. The time and effort required for this is not practical in most production environments, therefore, acquiring binaries from a trusted source and checking the MD-5 or PGP signatures is the most efficient procedure for acquiring clean binaries.

Planning

Planning the build of a new system can be time consuming but taking the time to plan will pay great dividends in creating a secure system which

meets the users needs. The first step is to determine the purpose of the system; web server, mail server, file server, etc. The ultimate purpose of the system should be one of the guiding factors in determining what software will be installed on the system. Interviewing the users, customer or department manager for whom the system is being built will help ensure the end product will meet their needs. They should be asked in detail the following; who, what, where, when and how. Who is such things as who will use the system, who will administer the system and who will be responsible for day to day maintenance. What is details such as what applications does the user need or what type of information will be stored on the server. Where means determining where will the system be located, both physically and on the network. When, which is usually yesterday, sets the timeline for building and installing the system. However, a realistic date the server needs to put in service should be negotiated to allow sufficient time to properly secure the system. How can refer to many things such as how many users will access the system and access profiles or how much money does the customer/department head have budgeted for this system. An analysis of the answers to these questions and others which may apply to your organization will yield a set of criteria to help you select the software packages for the new system. Do not install any software which does not fulfill a requirement based on your analysis of the users needs. Reducing the number of installed software packages is a good way to reduce the number of possible vulnerabilities. Research the available packages which will meet the users needs and check for exploits at web sites such as www.securityfocus.com/bugtraq or www.linuxsecurity.com. Packages selected by analyzing the user needs and that are not eliminated because of serious or unpatched exploits should be the best candidates. The most secure system is of little value to an organization if it does not meet the users needs in terms of features, price and performance.

Configuring

The work we have done up to this point has given us a solid understanding of how the system will function. We will use this information to help select the appropriate software for the server we begin the process of actually installing and configuring the software. This section makes reference to a number of excellent papers on securing Linux so the exact steps will not be repeated here, rather we will hit the high spots of several of these papers. In his white paper, Armoring Linux [3], the author recommends using the "Custom" option during RedHat Linux installation. Using the custom option will allow you to install only the packages which have been selected during the planning phase. By not installing unnecessary software we will make the job of configuring the server that much easier. This option is also available in Mandrake Linux and most other mainstream Linux distributions. In this same paper the

author also recommends a separate partition for /var since this is where the system logs and email will reside. If we placed system logs and email in the root partition, an attacker could attempt to fill the space with logs and bogus email resulting in a denial of service since Linux becomes very unhappy when the root partition be comes full. Once the required packages have been installed the task of configuring the system begins. If the system is running RedHat or Mandrake distribution, the hardening scripts available from the Bastille Linux [4] project can help tighten the security of these systems in an automated fashion. The script based approach has several benefits such as; ensuring steps in hardening the system are not skipped, ability to undo changes, log of the changes made and configuration consistency between systems. The scripts provide a way for busy administrator to tighten security with a minimum of effort. After running the scripts the system must still be tested to ensure their are no other vulnerabilities which need attention. Running the command "netstat -vat" will bring up a listing of all the currently running services on the system and tell their current state. Check this list against the services you planned to run to ensure services you need are not missing and services you do not need, which may contain a vulnerability, are not running. Another option which many new Linux distributions offer is the use of PAM (Pluggable Authentication Modules) which allows the use of advanced password checking, password file encryption and longer passwords. The last area to pay particular attention to is system logging. Linux offers a robust logging capability which should be taken advantage of to provide a means to check for unauthorized use of the system or intrusion. If possible, the logs should be stored on or backed up regularly to a dedicated logging server. When a hacker compromises a system, the logs are one of their prime targets for covering their activities. The last area which is often overlooked is physical security of the machine. If an attacker can gain physical access to the machine they have a number of additional avenues for attacking the server. If the machine is destined for a hosting facility than the use of BIOS passwords, case locks and floppy locks should be considered.

Documentation

Producing documentation for a newly configured system is one of the areas that is often overlooked in the rush to bring the system on line. Thoroughly documenting a system will pay dividends in several areas; day to day maintenance, forensic analysis in the event of a compromise and easy duplication of a system with similar capabilities. The day to day maintenance of any production system should consist of checking reputable sources such as Bugtraq for reported vulnerabilities and your distributions site for security upgrades and patches. A good system software log will allow you to quickly check the version and patch level of software on your system to determine if it is vulnerable or requires an

upgrade. A spreadsheet with a tab for each server and headings for such details such as; Date, Software Name, Distribution Name, Version Number, Patch Level and comments is a good start for keeping track of what software is on your servers. In Securing Linux, Part 2 [5], the author suggests making a CD-ROM snapshot of a new system before placing it into service. This is an excellent idea as it provides an unalterable baseline for comparing the system to if a compromise is suspected. Using a command such as "LS -alR > FileList.txt" from the root directory will create a text file of the machines directory structure, file permissions, filenames and sizes. This file should also be placed on the CD-ROM and can be used to quickly search for a particular file. Keep in mind certain directories such as /etc will be constantly changing so comparing them to a CD-ROM snapshot will not provide reliable information of a compromise. Good documentation will allow you to quickly build another system to either expand your network or replace a failed server without having to reinvent the wheel each time. Additionally, maintaining servers with an identical baseline becomes easier since you will not have to remember as many unique configurations. Though no one like to think about it, in the event of personnel turn over, the new person responsible for the system will be able to come up to speed much quicker with good documentation.

Summary

Properly securing a Linux system no more difficult than any other system if you follow a process. This paper summarizes a process which can be applied to most any system you will be required to build, configure and maintain. The bibliography contains the sources cited in the paper as well a several others I came across while researching Linux security for this paper. Linux and the threats a system faces are constantly changing, so remember good system security is not a state that is achieved. Rather it is a constant process of researching vulnerabilities which may affect your system, determining if they do in fact affect your system and promptly taking corrective action to eliminate the vulnerability. Review the system logs for anomalous activity and investigate the activity to determine if it is hostile or not. By being proactive about security you will ensure your organizations information, reputation and in many cases survival are protected.

Bibliography

Cited works

1. Levy, Elias. "BugTraq Vulnerability Statistics." "BugTraq Vulnerability Database"
http://www.securityfocus.com/vdb/stats.html (3 January 2001)

2. Internet Security Systems. "X-Force Database" http://xforce.iss.net (4 January 2001)

3. Spitzner, Lance. "Armoring Linux", 19 September 2000 http://www.enteract.com/~lspitz/linux.html (1 January 2001)

4. Beale, Jay. "Bastille Linux: A Walkthrough", 6 June 2000 http://www.securityfocus.com/focus/linux/articles/linux-bastille.html (1 January 2001)

5. Warfield, Michael. "Securing Linux, Part 2, Advanced Linux Security", "Securing Linux", July 1999 http://linuxworld.com/linuxworld/lw-1999-07/lw-07-ramparts_p.html (1 January 2001)

Other works not specifically cited

Warfield, Michael. "Securing Linux, Part 1, Elementary security for your Linux box", "Securing Linux", May 1999 http://linuxworld.com/linuxworld/lw-1999-05/lw-05-ramparts_p.html (1 January 2001)

Boran, Sean. "Hardening Red Hat Linux with Bastille" http://www.securityportal.com/cover/coverstory20000501.html (4 January 2001)

Andrews, James. "Linux Network Security", 29 May 1999 http://www.linuxplanet.com/linuxplanet/tutorials/211/1/ (4 January 2001)

Giannocavo, Patrick. "Hardening Linux Machines for Web Services", 8 August 2000 http://www.themestream.com/articles/72754.html (4 January 2001)

Ozancin, Craig. "Securing the Linux Environment Part One: Installation Issues", "Securing the Linux Environment" http://209.134.33.130/business_models/article.asp?ArticleID=1763 (4 January 2001)