Security Evaluation of the OpenBSD Operating System

Jeffery Hughes
ECE 578 Computer and Network Security
June 3, 2002

Security Evaluation of the OpenBSD Operating System

Table of Contents

Security Evaluation of the OpenBSD Operating System

Abstract

The developers of the OpenBSD operating system claim that it has been designed with security in mind. They claim that their proactive approach to security has allowed them to create a very secure operating system. The objective of this report is to examine the security philosophy in the OpenBSD operating system and how it has been implemented. Also, common areas for vulnerabilities will be examined to see how exposure in these areas has been mitigated.

Security Philosophy of the OpenBSD Operating System

OpenBSD is an open source Unix-like operating system based on the 4.4 BSD operating system. The stated goal of the developers of this operating system is to be “number one in the industry for security” [2]. They are meeting this goal through proactive security measures and integrating cryptographic solutions to security problems. The proactive security measures include “full disclosure of security problems” [2] with the operating system and an active code auditing process. Thus security problems are made public very quickly and of course a fix is provided as soon as possible. The second part of their proactive security approach is their code auditing process.

> “The process we follow to increase security is simply a comprehensive file-by-file analysis of every critical software component. We are not so much looking for security holes, as we are looking for basic software

> bugs, and if years later someone discovers the problem used to be a security issue, and we fixed it because it was just a bug, well, all the better. Flaws have been found in just about every area of the system. Entire new classes of security problems have been found during our audit, and often source code which had been audited earlier needs re-auditing with these new flaws in mind. Code often gets audited multiple times, and by multiple people with different auditing skills." [2]

The OpenBSD operating system also has used cryptography within the operating system itself and in applications provided with the operating system. [2]

Cryptographic Tools and Related Software Applications

Pseudorandom number generators (PRNG) are a very useful part of the cryptographic toolbox within the OpenBSD operating system. If the numbers used to seed cryptographic functions can be guessed then the security of the functions has been greatly compromised. To avoid this problem the OpenBSD operating system has implemented mechanisms to produce higher quality pseudorandom numbers in order to ensure that system security is maintained. To create pseudorandom numbers the operating system starts by creating a randomness pool by collecting measurements on inter-key press intervals, arrival time of packets, and timing of disk access requests. Random numbers are generated by hashing this randomness pool with the MD5 algorithm. [3]

One of the uses of the randomness pool includes using it to seed the arc4random interface. The arc4random interface is the replacement for the standard Unix random interface and provides higher quality pseudorandom numbers. The output from arc4random is used as the seed for the ARC4 stream cipher. The operating system also incorporates a non-repeating pseudorandom number generator that generates 16-bit non-repeating numbers using a random seed from the randomness pool. Also, random numbers are used as data-block padding for cryptographic algorithms, such as IPsec, used within the operating system. The PRNGs provide the salts that are used in the various password algorithms. Finally, the key exchange systems isakmpd and photurisd use the randomness pool to create random Diffie-Hellman values and random nonces. [3]

OpenBSD comes with the IPsec protocol stack. This is to remedy the weakness inherent in the regular IP protocol. The basic IP protocol uses no methods to prevent third parties from viewing transmitted data or prevent tampering with the transmitted data. IPsec provides authentication, integrity, and confidentiality to IP packets transmitted over a network. The IPsec protocol can be used provide end-to-end security between two computers or it can be used on security gateway computers that provide a security tunnel between two networks. The OpenBSD IPsec protocol can use the following algorithms for encryption: DES, 3DES, Cast-128, Blowfish, and Skipjack. It also can use the following hash algorithms: MD5, SHA-1, and RIPEMD-160. It can use either isakmpd implementing the internet key exchange (IKE) protocol or photurisd implementing Photuris for key management. The advantage of using security at the IP layer is that it is transparent to applications and users. [3]

OpenBSD uses the Bcrypt as its default user password scheme. This improves security over the traditional Unix crypt password scheme. Crypt uses passwords with a maximum length of 8 characters. It uses the password with a 12-bit salt as the 56-bit key to the DES algorithm which then encrypts a 64-bit string of zeros 25 times. The encrypted value with the salt is stored as the password value. The OpenBSD password scheme uses the following techniques to increase password security: passwords can have a length from 6 to 128 characters long, the characters can be alpha-numeric or special characters, and the passwords can age out. It uses Bcrypt to create the stored password value. Bcrypt takes the password, a 128-bit salt, and a cost value and uses the Blowfish encryption algorithm to encrypt a 192-bit magic value 64 times in ECB mode to generate the stored password value. The password is the key for the algorithm, the cost value determines how long it takes to produce the key schedule, and the 128-bit salt modifies the key schedule. The improvement in security that this password scheme provides consists of increasing the time necessary for someone to perform a brute-force dictionary attack on the password file. The time to perform an encryption is intentionally slowed down by the time necessary to produce the key schedule, but not so slow as to be noticed by a user. The 128-bit salt is to prevent computing and storing a list of values to simply compare to the password file, since the storage requirements are very large. The cost is a value that can be changed to increase the time necessary for key scheduling and thus for computation as computer hardware gets faster. [4]

The OpenBSD operating system includes the Kerberos V system. This system enables a key distribution center to distribute session keys to clients. These keys will then provide confidentiality between the two computers communicating across a network by using encryption. The Kerberos system uses the DES encryption algorithm to encrypt messages containing the session key that are sent to the client. The Kerberos system also can be used for authentication of users on a local workstation since the login facilities have been enabled to work with Kerberos. [3]

A one-time password system supplied with the OpenBSD operating system is S/Key. This uses a pass-phrase from the user and a one-way hash function to generate a list of one-time passwords. The choice of hash function used can be MD4, MD5, SHA1, or RIPEMD-160. This is a useful feature if there are not sufficient security mechanisms implemented to ensure the protection of the transmitted password. [3]

The OpenBSD operating system includes OpenSSH which is a secure shell implementation that is compatible with SSH version 1.3, 1.5, 2.0. OpenSSH is a replacement for the r-services, such as rlogin used by Telnet, that do not provide security for user names and passwords as they are transmitted from client to server. The user name and passwords are encrypted thus providing more security by preventing an attacker from learning passwords by sniffing the communications channel. [5] [3]

OpenBSD also supports the use of Secure Sockets Layer (SSL). This is implemented in the HTTP server and provides data encryption between a server and client so that sensitive data can be transmitted over the web. [3]

Common Vulnerabilities and the OpenBSD Solutions

One area of common security problems is that many services, such as HTTP servers and Telnet, are enabled and active after the installation of the operating system whether or not the user will use them. This allows unneeded services to be running which could enable an attack on the computer. To mitigate this common vulnerability the OpenBSD operating system installs in a “Secure by Default mode. All non-essential services are disabled” [2] in the default installation. This may create some inconvenience for the user that will have to turn on the services before they can use them, but this practice eliminates one of the most common areas for vulnerabilities. [2]

A widespread vulnerability exists in the TCP protocol that can lead to spoofing attacks. Using random numbers helps prevent spoofing attacks against TCP.

> "The predictablility of TCP initial send sequence values has been known to be a security problem for many years. Typical systems added either 32k, 64k, or 128k to that value at various different times. Instead, our new algorithm adds a fixed amount plus a random amount, signficantly decreasing the chances of an attacker guessing the value and thus being able to spoof connection contents." [6]

Again, in an attempt to reduce the chance that an attacker will gain knowledge about a computer on the network that is sending out packets the non-repeating PRNG is used to assign the 16-bit identifier for each IP packet.

> "Another issue was avoiding disclosure of information when using IPsec in tunneling mode. A naive implementation might create a new IP header with an ID one more than the ID in the existing IP header. This could lead to known-plaintext attacks against IPsec." [6]

Another area where randomness is useful is in the allocation of ports by services. Most services allocate ports in a predictable fashion, such as bind that allocates ports incrementing from 1024. OpenBSD replaces the old code in these services that caused this predictable behavior with code that will choose random port numbers in the specified range of the service.

> "There are a number of poorly designed protocols (e.g., rsh, ftp) which are affected by predictable port allocation; we believe that our approach is making it harder for attackers to gain an edge." [6]

Also, process ID's are randomized in OpenBSD simply because there is a lot of code that is written to use the PIDs as a random number, but since it has poor randomness qualities this can compromise security by allowing an attacker to have knowledge about the program that is using the PIDs as random numbers. This is true of the remote procedure call transaction (RPC) IDs and the NFS RPC IDs, which both use the PIDs for their IDs and then increment from there. This could allow an attacker to possibly guess the IDs. The RPC and NFS RPC IDs are randomized using the arc4random interface. [3]

Similarly to the TCP spoofing vulnerability, PRNGs are used to prevent spoofing attacks against DNS.

> "DNS query IDs typically start at 1 and increment for each subsequent query. An attacker can cause a DNS lookup, e.g., by telneting to the target host, and spoof the reply, since the content of the query and the ID are known or easily predictable." "To avoid this issue, we have modified our in-tree copy of bind and our libc resolver to make use of the non-repeating PRNG." [7]

Scanning OpenBSD for Common Vulnerabilities

There are many readily available software packages that will perform security scans of an operating system and check for common vulnerabilities. Three packages were used to scan the OpenBSD operating system after the default installation. These three packages included the Computer Oracle and Password system (COPS), Strobe, and Network Map (Nmap). COPS performs a broad-based scan for security issues. The areas that COPS scans includes:

> -File, directory, and device permissions/modes.
> -Poor passwords.
> -Content, format, and security of password and group files.
> -The programs and files run in /etc/rc* and cron(tab) files.
> -Existance of root-SUID files, their writeability, and whether or not they are shell scripts.
> -A CRC check against important binaries or key files to report any changes therein.
> -Writability of users home directories and startup files (.profile, .cshrc, etc.)
> -Anonymous ftp setup.
> -Unrestricted tftp, decode alias in sendmail, SUID uudecode problems, hidden shells inside inetd.conf, rexd running in inetd.conf.
> -Miscellaneous root checks -- current directory in the search path, a "+" in /etc/host.equiv, unrestricted NFS mounts, ensuring root is in /etc/ftpusers, etc.
> -Dates of CERT advisories vs. key files. This checks the dates that various bugs and security holes were reported by CERT against the actual date on the file in question.
> -The Kuang expert system. This takes a set of rules and tries to determine if your system can be compromised. [8]

Strobe is port scanner that scans for open TCP ports on a computer system. Nmap is a more sophisticated port scanner that scans for open TCP ports, open UDP ports, and also does system fingerprinting. [9] [10]

Here is a summary of the results of the COPS scan. After COPS performed the scan it produced only two warnings about the system. The first was that there are no restrictions on who can mount the file system on the computer. This may or may not be a problem, it all depends on the particular situation the system is used in. The second was that the etc/security file could be read by any user. This is a problem since it allows anyone to view the file and gain knowledge about some security settings. Similarly the etc/passwd file can be read by any user thus giving information, such as user names. Access should be restricted to only privileged users for these files. Interestingly the COPS scan did not record the etc/passwd problem the second time it was run and thus is not included in the results in Appendix A. The complete output results of the COPS scan can be found in Appendix A.

The Strobe scan showed that only five TCP ports were open; port 13(daytime), port 22(ssh), port 37(time), port 111(sunrpc), port 113(auth). The notable services that are missing from this list are services like Telnet, SNMP, and HTTP. By having these services shutdown in the default installation reduces a lot of common vulnerabilities. The fact that there are few ports open and thus few services running makes the system more secure because it gives attackers less possible options with which to attack the system. Complete results for the Strobe scan can be found in Appendix B.

The Nmap scan produced the same results as Strobe for the TCP scan. Nmap also revealed that six UDP ports were open; port 68(dhcpclient), port 111(sunrpc), port 512(biff), port 514(syslog), port 748(ris-cm), port 1011(unknown). Once again, the fewer ports open, the tighter the security. The fingerprint scan resulted in guesses for the operating system being either Mac O/S 8.5 or OpenBSD 3.0. This is an area that some improvement could be made to minimize the amount of information that an attacker can gain about the system. Another Nmap scan revealed the types of protocols that are active on the system; ICMP, IGMP, IP, TCP, UDP, IPv6, GRE, ESP, AH, Mobile, etherIP, and IPcomp. This is another area where an attacker could gain information about the system but in this case it is difficult or impossible to keep from revealing this information since protocols are standards that work in a set fashion. The fingerprint scans that produce information about the computer should be detected and be logged as a warning for system administrators. Complete results for the Nmap scan can be found in Appendix C.

Recently Discovered Vulnerabilites

Over the past six months a few vulnerabilities on the OpenBSD operating system have come to light. These vulnerabilities are posted on the OpenBSD website and are mostly comprised of vulnerabilities in specific software packages, such as SSH or Sendmail, but also cover vulnerabilities in the kernel and other areas of the system. The fixes for these vulnerabilities are also posted on the OpenBSD website. These vulnerabilities are listed in Appendix D.

Conclusion

Cryptography has been integrated into the OpenBSD system and implemented in such a way as to solve the security issues that exist in many other operating systems. The results of several security scans have shown that the design of the system is secure with a few possible areas of improvement. The proactive security approach in the design of the OpenBSD operating system has succeeded in producing a secure operating system.

References

[1] OpenBSD website home page, http://www.openbsd.org/.

[2] OpenBSD website “Security” page, http://www.openbsd.org/security.html.

[3] T. de Raadt, N. Hallqvist, A. Grabowski, A. Keromytis, and N. Provos. Cryptography in OpenBSD: An Overview. Obtained from http://www.openbsd.org/crypto.html.

[4] Neils Provos and David Mazieres. A Future-Adaptable Password Scheme. In *Proceedings of the Annual USENIX Technical Conference*, 1999. Obtained from http://www.openbsd.org/crypto.html.

[5] OpenBSD website “Cryptography” page, http://www.openbsd.org/crypto.html.

[6] T. de Raadt, N. Hallqvist, A. Grabowski, A. Keromytis, and N. Provos. Cryptography in OpenBSD: An Overview. Section 3.3.1.

[7] Ibid. Section 3.3.2.

[8] COPS software package, version 1.04. Obtained from ftp://ftp.jaring.my/pub/cert/tools/cops/.

[9] Strobe software package, version 1.03. Obtained from http://www.deter.com/unix/software/strobe103.tar.gz.

[10] Nmap software package, version 2.54BETA34. Obtained from http://www.insecure.org/nmap/.

Appendix A

Results of the COPS Security Scan

```
ATTENTION:
Security Report for Sat Jun 1 12:31:06 PDT 2002
from host ece578


**** root.chk ****
**** dev.chk ****
Warning!  NFS file system  exported with no restrictions!
Warning!  NFS file system  exported with no restrictions!
Warning!  NFS file system  exported with no restrictions!
Warning!  NFS file system  exported with no restrictions!
Warning!  NFS file system  exported with no restrictions!
**** is_able.chk ****
Warning!  /etc/security is _World_ readable!
**** rc.chk ****
**** cron.chk ****
**** group.chk ****
**** home.chk ****
**** passwd.chk ****
**** user.chk ****
**** misc.chk ****
**** ftp.chk ****
**** pass.chk ****
**** kuang ****
**** bug.chk ****
```

Appendix B

Results of the Strobe Port Scan

```
192.168.1.105                  daytime             13/tcp Daytime
[93,JBP]
192.168.1.105                  unknown             22/tcp unassigned
192.168.1.105                  time                37/tcp Time
[108,JBP]
192.168.1.105                  sunrpc             111/tcp SUN Remote
Procedure Call [DXG]
192.168.1.105                  auth               113/tcp
Authentication Service [130,MCSJ]
```

Appendix C

Results of the Nmap Port Scan

```
# nmap (V. 2.54BETA34) scan initiated Sun Jun  2 15:55:13 2002 as: nmap
-vv -sU -O -oN results_nmap1 192.168.1.105
Warning:  OS detection will be MUCH less reliable because we did not
find at least 1 open and 1 closed TCP port
Interesting ports on  (192.168.1.105):
(The 1453 ports scanned but not shown below are in state: closed)
Port       State       Service
68/udp     open        dhcpclient
111/udp    open        sunrpc
512/udp    open        biff
514/udp    open        syslog
748/udp    open        ris-cm
1011/udp   open        unknown
Remote OS guesses: Mac OS 8.5, OpenBSD 3.0 (x86 or SPARC)
OS Fingerprint:
T5(Resp=Y%DF=Y%W=0%ACK=S++%Flags=AR%Ops=)
T6(Resp=Y%DF=Y%W=0%ACK=O%Flags=R%Ops=)
T7(Resp=Y%DF=Y%W=0%ACK=S%Flags=AR%Ops=)
PU(Resp=Y%DF=N%TOS=0%IPLEN=38%RIPTL=134%RID=E%RIPCK=F%UCK=E%ULEN=134%DA
T=E)


# Nmap run completed at Sun Jun  2 15:55:39 2002 -- 1 IP address (1
host up) scanned in 26 seconds


# nmap (V. 2.54BETA34) scan initiated Sun Jun  2 15:52:22 2002 as: nmap
-vv -oN results_nmap 192.168.1.105
Interesting ports on  (192.168.1.105):
(The 1551 ports scanned but not shown below are in state: closed)
Port       State       Service
13/tcp     open        daytime
22/tcp     open        ssh
37/tcp     open        time
111/tcp    open        sunrpc
113/tcp    open        auth

# Nmap run completed at Sun Jun  2 15:52:47 2002 -- 1 IP address (1
host up) scanned in 25 seconds


# nmap (V. 2.54BETA34) scan initiated Sun Jun  2 15:59:21 2002 as: nmap
-vv -sO -oN results_nmap2 192.168.1.105
Interesting protocols on  (192.168.1.105):
(The 243 protocols scanned but not shown below are in state: closed)
Protocol   State       Name
1          open        icmp
2          open        igmp
4          open        ip
```

```
6          open        tcp
17         open        udp
41         open        ipv6
47         open        gre
50         open        esp
51         open        ah
55         open        mobile
97         open        etherip
108        open        ipcomp

# Nmap run completed at Sun Jun  2 16:41:32 2002 -- 1 IP address (1
host up) scanned in 2531 seconds
```

Appendix D

OpenBSD vulnerabilities from the last six months posted on the OpenBSD website.

May 8, 2002: A race condition exists that could defeat the kernel's protection of fd slots 0-2 for setuid processes.

April 25, 2002: A bug in sudo may allow an attacker to corrupt the heap.

April 22, 2002: A local user can gain super-user privileges due to a buffer overflow in sshd(8) if AFS has been configured on the system or if KerberosTgtPassing or AFSTokenPassing has been enabled in the sshd_config file.

April 11, 2002: The mail(1) was interpreting tilde escapes even when invoked in non-interactive mode. As mail(1) is called as root from cron, this can lead to a local root compromise.

March 19, 2002: Under certain conditions, on systems using YP with netgroups in the password database, it is possible for the rexecd(8) and rshd(8) daemons to execute a shell from a password database entry for a different user. Similarly, atrun(8) may change to the wrong home directory when running jobs.

March 13, 2002: A potential double free() exists in the zlib library; this is not exploitable on OpenBSD. The kernel also contains a copy of zlib; it is not currently known if the kernel zlib is exploitable.

March 8, 2002: An off-by-one check in OpenSSH's channel forwarding code may allow a local user to gain super-user privileges.

January 21, 2002: A race condition between the ptrace(2) and execve(2) system calls allows an attacker to modify the memory contents of suid/sgid processes which could lead to compromise of the super-user account.

January 17, 2002: There is a security hole in sudo(8) that can be exploited when the Postfix sendmail replacement is installed that may allow an attacker on the local host to gain root privileges.