GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Security Essentials (Security 401)"
at http://www.giac.org/registration/gsec

NFS Security

By Samuel Sheinin

This paper will address NFS security issues, indicate some measures that can be taken to prevent unwanted intrusions as well as auditing tools, and discuss current efforts to enhance NFS security

NFS Protocol

Network Filesystem [1] protocol was designed and implemented by Sun Microsystems in mid 1980's to provide remote access to shared files. It uses client-server model where client imports file systems from other machines and server exports local filesystem to the other machines. The protocol is stateless which means no client-server state is maintained; hence in an event of failure (client or server) NFS can continue to operate. It was designed to support UNIX filesystem semantics so that protection and access controls are based on the process presenting a UID and set of groups checked against the file's owner, group, and other access modes. The NFS protocol uses Sun's Remote Procedure Call protocol [2] that allows programs running on one computer to call subroutines that are executed on another computer, and external data representation [3] that allows exchange of information between different kinds of computers. It can run over a TCP stream or UDP datagram, typically UDP, which is an unreliable protocol (there are no guarantees that the transmitted packets will be delivered in order or delivered at all). NFS though imposes a requirement for every RPC command to be acknowledged and calls for retransmissions when needed.

On the server side the **portmap**, **mountd**, and **nfsd** daemons must be running. The portmap daemon registers rpc-based services. When an RPC daemon is started, it tells the portmap daemon on which port number it's listening and what RPC services it serves. When a client makes an RPC call to a service it contacts the portmap daemon to determine the port number to which RPC messages should be sent.

Mountd reads the /etc/exportfs file and creates a list of hosts and networks to which each local file system may be exported. Client mount requests are directed to the mountd daemon. After verifying that the client has permission to mount the requested filesystem, mountd returns a file handle for the requested mount point.

The nfsd daemon forks off children, which provide a process context for the NFS RPC daemons.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

NFS Security Issues

NFS like any other unprotected network protocol is vulnerable to two types of attacks: eavesdropping and impostor attack. An eavesdropper can pick up unauthorized data as it goes by on the network. An impostor can gain an unauthorized access to the network.

An NFS server is unable to distinguish falsified file handles from the file handles established by the mountd daemon. A client who manages to snoop the network and steal a file handle, can read and modify any file on the server not owned by root.

NFS exports are controlled locally – each mount point has a list of hosts to which the file system may be exported. This list is enforced by the mountd daemon only, a malicious client can access ask the servers's portmap daemon to forward the request to the mount daemon. When mountd receives the request from portmap, it thinks it's received from a valid client and forwards the file handle to the intruder.

If filesystem is exported without restrictions, an intruder can remotely compromise user or system files, and take over the machine.

By default, the user identity required to access a remote file or directory is specified with the UNIX numeric userid and groupid (AUTH_UNIX). Any user can run a program to generate an NFS request and obtain access to files on behalf of any user.

Basic Measurements for Securing NFS

- Keep up with and install most current NFS security patches
- If possible export file systems read-only
- Export file systems to a restricted set of hosts
- Configure NFS so that requests are only accepted from privileged system programs [4]
- Do not export a file system to the exporting server or to a netgroup which includes the server
- Make sure that there is no localhost reference in /etc/exportfs file and export to fully qualified domain names to diminish spoofing attempts
- Verify that the export lists does not exceed 256 characters (some systems known to ignore the list completely in this case) [5]
- Keep all suid code on one filesystem and export it no root access, mount all other filesystems –nosuid

© SANS Institute 2000 - 2002, Author retains full rights.

- Assure that statmon directories used by statd daemon to control locking services of NFS are not world writable
- Run a portmap (or rpcbind) program that does not forward mount requests [6]
- Make sure that NIS netgroup does not contain empty host fields which are treated as wildcards and will cause mountd daemon to grant access to any host
- Block TCP and UDP ports 2049 (nfs) and 111 (portmap) on the routers and firewalls
- Disable NFS if it's not needed

Auditing Tools

- Use **showmount** command to display filenames exported by a given host
- Use **SATAN** (Security Analysis Tool for Auditing Networks) tool that examines remote hosts by probing NIS, NFS, FTP, and other services. The information gathered includes which network information services are present and which ones are incorrectly configured causing security holes as well as well-known bugs in system or network utilities [7]
- Use **nfswatch** to monitor NFS requests to any given machine or the entire local network [8]
- Use **NFS tracer** to monitor NFS traffic [9]
- Use **nfsbug** to test hosts for well known NFS problems/bugs [10]

Solaris Enchancements for NFS Security

Solaris includes several tools that are not available for other flavors of UNIX such as:

- There are two authentication services available under NFS. The first, Diffie-Hellman **AUTH_DH**, authenticates clients by using asymmetric encryption; only those who know the decryption key can convert the data to its original form. The second, Kerberos 4 **AUTH_KERB**, uses tickets that provide expanded information about the user and an encrypted DES key that is assigned by a trusted host called the Key Distribution Center (KDC). Both AUTH_DH and AUTH_KERB mechanisms use encryption to protect authentication information
- **ACL**s (Access Control Lists) go beyond traditional UNIX file permissions by allowing administrators to set authorization information for each file
- **ASET** (Automated Security Access Tool) can be run periodically assessing the overall security state of the system checking for existence of new setuid programs, contents of .rhosts, home directory permissions, …
- Besides the use of syslogs, Solaris offers **C2 auditing** which allows an administrator to log any event s/he deems security-relevant

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Future NFS Security Enchancements

NFS Version 4

RFC 2624 NFS Version 4 Design Considerations [11] advocates that the protocol should provide strong authentication, integrity, and privacy. It suggests that RPCSEC_GSS proposal [12] will satisfy these requirements.

Authentication

RPCSEC_GSS provides framework for Simple Public-Key Mechanism (SPKM) [13] and Kerberos V5 [14].

Data Integrity

RPCSEC_GSS provides a cryptographically strong checksum.

Data Privacy

RPCSEC_GSS provides data encryption.

Firewall and Proxy Servers

NFS protocol should allow its use via Internet firewalls and proxy/caching servers and deal appropriately with the issues of authentication, authorization, and cache content validation.

© SANS Institute 2000 - 2002, Author retains full rights.

References

[1] RFC1813 - NFS Version 3 Protocol Specification
http://www.ietf.org/rfc/rfc1813.txt

[2] RFC1831 - RPC: Remote Procedure Call Protocol Specification Version 2

http://www.ietf.org/rfc/rfc1831.txt

[3] RFC1832 - XDR: External Data Representation Standard
http://www.ietf.org/rfc/rfc1832.txt

[4] Unprivileged NFS access
http://www.cerias.purdue.edu/coast/satan-html/tutorials/vulnerability/NFS_export_to_unprivileged_programs.html

[5] CERT(*) Advisory CA-94:02
ftp://ftp.cert.org/pub/cert_advisories/CA-94:02.REVISED.SunOS.rpc.mountd.vulnerability

[6] CERT(*) Advisory CA-94:15
ftp://ftp.cert.org/pub/cert_advisories/CA-94:15.NFS.Vulnerabilities

[7] Improving the Security of Your Site by Breaking Into it
http://www.cerias.purdue.edu/coast/satan-html/docs/admin_guide_to_cracking.html

[8] nfswatch
http://www.madness.net/software/SGI_freeware/SGI_freeware_CD.2.0/relnotes/nfswatch.html

[9] NFS tracer
ftp://coast.cs.purdue.edu/pub/doc/network/passive-network-monitoring.ps.Z

[10] nfsbug
ftp://coast.cs.purdue.edu/pub/tools/unix/nfsbug

[11] NFS Version 4 Design Considerations
http://www.ietf.org/rfc/rfc2624.txt

[12] RPCSEC_GSS Protocol Specification
http://www.ietf.org/rfc/rfc2203.txt

[13] The Simple Public-Key GSS-API Mechanism (SPKM)
http://www.ietf.org/rfc/rfc2025.txt

[14] The Kerberos Network Authentication Service (V5)
http://www.ietf.org/rfc/rfc1510.txt

[15] NFS Security
http://www.sun.com/software/white-papers/wp-nfs/nfs_15.html

© SANS Institute 2000 - 2002, Author retains full rights.

[16] Sun Solaris Security
http://www.sun.com/software/white-papers/wp-security/

© SANS Institute 2000 - 2002, Author retains full rights.