# Deploying Secure NFS

Nicolas Williams

Staff Engineer

Sun Microsystems, Inc.

nicolas.williams@sun.com

# Secure NFS Background

## A Brief History – Protocol

- In the beginning, no security
  - AUTH_SYS, AUTH_NONE (1984)
- First attempts at security
  - AUTH_DH (1987)
  - AUTH_KERB (1992)
- RPCSEC_GSS (1997)
  - Generic, “pluggable,” extensible

# Secure NFS Background

## A Brief History – Implementations

- SunOS 4.x, AUTH_DH (1987)
- Solaris 2.6, SEAM (2000)
  - RPCSEC_GSS, Kerberos V available as *Sun Enterprise Authentication Mechanism*
- Since then Solaris supporthas improved; Linux, Hummingbird, NetApp, and others have added support for RPCSEC_GSS

# High-Level View

- Step one: key distribution
  - Plan, deploy Kerberos V Realm(s) KDCs
  - Host keying
    - NFS, other services, clients*
  - User keying (password migration)
- Step two: secure the actual shares
    - share -o sec=sys → share -o sec=krb5i

# Deploying Kerberos V

## Planning Realms

- Plan krb5 realms along boundaries of current administrative domains
  - One IT dept. → one realm
- Name realms after DNS domains
  - No need for a realm for each sub-domain
  - Kerberos V has not be internationalized
    - So only ASCII-only realm names work for now!

# Deploying Kerberos V

## KDC Infrastructure

- Plan number of KDCs, topology, replication
  - One or two KDCs per-supported site
  - No need for big iron for KDCs
- Physical security
  - Kerberos V KDCs are trusted third parties that share secret keys w/ all principals
    - KDC theft is a Bad Thing

# Deploying Kerberos V

## Key Distribution

- Key your services
  - nfs/<hostname.fqdn>@<realm>
  - host/<hostname.fqdn>@<realm>
  - Where necessary*, key your clients
    - host/..., root/...*
- Give users Kerberos V principals and passwords
  - <username>@<realm>

# Securing NFS Shares

## NFS Security "Flavors"

- AUTH_* (NONE, SYS, DH)
- RPCSEC_GSS
  - GSS-API mechanism, protection level, QoP
    - `krb5` → Kerberos V, authentication only
    - `krb5i` → Kerberos V, integrity protection
    - `krb5p` → Kerberos V, privacy protection
    - `dh` → MECH_DH, authentication only
    - SPKM, LIPKEY

# Securing NFS Shares

## Throwing the Switch

- Server must be keyed
- Relevant users must be keyed
  - Sometimes clients must be keyed also
- Flip switch per-share
  - Multiple sec flavors OK, but make no sense
    - `sec=krb5i:sys` → as insecure as `sec=sys`
    - `sec=sys:krb5` → fine for testing
  - Mind the server's defaults!

# General Notes

Careful with that Ax Eugene

- Compatibility
- Principal → user mapping
- Credential management
- “Enctypes”
- NFS sec flavor negotiation
- Upkeep

# Compatibility Notes

Not Too Bad

- NFS interoperability is really good
- But, several different KDC admin protocols, tools don't help
  - “ktadd” not very interoperable yet
  - Workaround: create 'keytabs' on compatible client, copy to incompatible target
  - Several different Kerberos V password-changing protocols
  - Most support one particular such protocol

# Principal Mapping

A Server-Side Issue

- Windows 2000 and up uses Kerberos V principals as usernames
  - But mapping may still be needed for principals from non-Windows realms
- Where this is not so (e.g., Solaris, Linux), principal→user mappings are needed

# Principal Mapping

- Linux, Solaris, use gsscred table and/or krb5.conf mappings
- NetApp maps user principals in server's default realm to files, NIS, or LDAP users, as per config
  - root principals mapped to uid 0 per-root exportfs option
- Check your server's docs

# Credential Management

## Yes, Network Credentials Should Expire

- Credentials represent users, clients, services
  - Kerberos V lacks revocation facility, relies on short ticket lifetimes
  - Stolen creds → impersonation
  - Disabling principals
- So creds should have short, finite lifetimes

# Credential Management

Dealing w/ Ticket Expiration

- Platform support can help
  - Auto-renew Kerberos V tickets
    - Auto-re-delegation of tickets
  - Auto-refresh Kerberos V tickets
    - At screen unlock time, say, or on-demand if passwords are cached
- Medium-lived TGTs (say, 7 days), *short*-lived *service* tickets (say, 30 min.)

# "Enctypes"

Get this Right

- Make sure that your host service principals have keys for **only** the enctypes they support
- Make sure that your user principals have keys for the *strongest* enctypes supported by the hosts they log into with passwords

# Secure NFS Negotiation

More on Throwing the Switch

- Multi-user timesharing clients typically mount with one NFS security “flavour,” thus the need for per-share/mount flag days
  - Specify one on mount or let one be negotiated
  - Whichever you get applies to all users on client
  - Details of negotiation may be implementation specific (see later slide on Solaris 10)
  - Be aware of how your clients negotiate NFS sec flavours, if not specifying one on the client-side

# And After Deployment?

## Upkeep

- Key new hosts/services, users
- Revoke old ones
- Install decent password quality policies
  - Even before deploying!
- Mind your KDCs!

# Secure NFS Client Availability by Platform

- Linux 2.6, check your distro
  - Fedora core 2
- FreeBSD 5.2, OpenDarwin
- AIX 5.3
- Solaris 2.6 and up
- Windows 2000 and up
  - Hummingbird NFS Maestro 8.0 and up

# Secure NFS Server Availability by Platform

- Linux 2.6, check your distro
  - Fedora core 2
- AIX 5.3
- Solaris 2.6 and up
- Windows 2000 and up
  - Hummingbird NFS Maestro 8.0 and up
- NetApp ONTAP 6.2

# Kerberos V KDC Availability by Platform

- Windows 2000 and up
  - ActiveDirectory
- Cybersafe
  - Runs on Windows, Solaris AIX, HP/UX
- AIX 5.1 and up
- Solaris 2.6 and up
- *cont.*

# Kerberos V KDC Availability by Platform

- Linux distros, *BSDs, open source
  - MIT krb5
  - Heimdal
  - Shishi (GNU)

# NFSv4 Availability by Platform

- Linux 2.6, check distros
  - Fedora core 2 and up
- Windows 2000 and up
  - Hummingbird NFS Maestro 8.0 and up
- Solaris 10
- AIX 5.3
- FreeBSD 5.2 and up

# Secure NFS, Kerberos, on Solaris

- Availability by release
- What's new in Solaris 10
- Client keying requirements in Solaris 10
- Deployment tips and tools
- NFS sec flavor negotiation

# Availability by Solaris Release

- NFSv3
  - Solaris 2.5.1
- RPCSEC_GSS, GSS-API, Kerberos V mechanism
  - Unbundled in 2.6, bundled in Solaris 8
- NFSv4
  - Solaris 10

# Availability by Solaris Release

- Utilities, KDC
  - Unbundled in 2.6, bundled in in Solaris 9
- Kerberized telnet, r-cmds, FTP
  - Unbundled in 2.6, bundled in Solaris 10
- Secure Shell w/ GSS-API support
  - Solaris 10

# What's New in Solaris 10

## With Respect to Kerberos V Support

- Kerberos V improvements
  - New crypto: 3DES, RC4, AES
  - Solaris Cryptographic Framework
  - Resync'ed with MIT krb5 1.2.1 + much of 1.3
    - KDC exchanges over TCP, IPv6 support, much more
  - Better deployment tools

# What's New in Solaris 10

With Respect to NFS Support

- Relaxed host keying reqs for clients
  - No need for “root” principals (except for share -o root=<list> uses)
  - No need for “host” principals on single-user clients; host/<random> also OK for road warriors
- Improved principal to user mapping
- **NFSv4**
- Secure NFS Clustering

# Solaris KDCs

## Planning KDC Infrastructures

- One master, multiple slaves
  - One or two per-supported site
  - Big iron is **not** needed for KDCs
  - Use Incremental Propagation (iprop) for fast synchronization with master KDC
    - Incremental Propagation is new in Solaris 10

# Deployment Tools: kclient

Configuring and Keying Servers, Clients

- kclient(1M)
  - More functional than sysidkrb5(1M)
  - Set up krb5.conf(4) from profiles
  - Keys clients, servers with kadmin(1M)

# PAM Configuration

## Configuring PAM to Use Kerberos V

- Read docs :)
  - pam.conf(4), pam_krb5(5), pam_krb5_migrate (5), AnswerBook
- Design a PAM config for relevant services
  - pam_krb5 required? sufficient? binding?  See examples in pam_krb5(5)
- Deploy pam.conf changes

# Deployment w/ Solaris

## User Password Migration

- Enable automatic user migration in master KDC's kadm5.acl(4)
- Enable automatic user migration in clients' pam.conf(4) by adding pam_krb5_migrate(5)
- Watch users automatically get Kerberos V principals
  - Use kadmin policies to force password aging

# *Notes on Client Keying

It's Easier Now

- Time sharing clients should have "host" principals
  - For user authentication
  - For some per-NFSv4 mount state (clientid)
    - Single-user (home, laptop) systems can do w/o
- "root" principals
  - Required pre-Solaris 10
    - Now required only for root-equivalent access

# Secure NFS Negotiation

Things Worth Knowing

- When mounting w/o sec option, client picks 1st from server offering for which credentials are available
  - [Solaris 10] for the user that triggered the mount
  - [per-Solaris 10] for the client's root principal
- One sec flavor per-mount
  - All users on client are affected; relevant users must have credentials, else they get EACCES

# Secure NFS Negotiation

## A Word About Solaris' nfssec.conf(4)

- nfssec.conf(4) 'default' entry provides default sec flavor for share commands
  - And for WebNFS (v3) mounts
- NFSv3 clients negotiate only sec flavors listed in nfssec.conf
- NFSv4 clients ignore nfssec.conf(4)

# What About MECH_DH

## SecNFS w/ the Sun DH GSS Mech

- Really, don't use DH
  - This slide is here for completeness, and to show similarity with Kerberos V deployment
- Step one: key distribution
  - Deploy LDAP directory
  - Key all hosts and users
- Step two: secure the actual shares
  - share -o sec=sys → share -o sec=dh

# What About MECH_DH

## Deploying w/ AUTH_DH/MECH_DH

- AUTH/MECH_DH issues
  - Authentication only, no transport protection
  - Tiny keys for NIS, files backends
  - Larger keys only for NIS+ (EOLed), LDAP
  - Deployment story is similar to Kerberos V
    - But more difficult in some ways
  - Limited support – only* Sun implements it
- Use Kerberos V instead

# Q/A