


# Managing Security Setup Issues for SecureNFS

Tom McNeal

TMCN Consulting

trmcneal@comcast.net

# Section 1

# General Overview

# SecureNFS

ONC+2.3 on HP-UX 11.31 Release

- Security service access within NFS
  - Stronger Authentication
  - Data Integrity & Encryption
  - RPCSEC_GSS
- Security designated on both client and server
  - Server exports with “sec=“ option
  - Client mounts with matching “sec=“ option

# Security Modes & Flavors

- AUTH_NONE, AUTH_SYS
  - Original capabilities
- AUTH_DH
  - sec=dh option
- RPCSEC_GSS
  - sec=krb5
  - sec=krb5i
  - sec=krb5p

# Environmental Setup

- NFSTEST_2K test environment
  - Sets up support for specific modes
    - Configuration files (*.conf)
    - Keys, Tickets, directories, files, etc.
  - Stops and starts required services
    - yp* functions for NIS
    - DNS
    - Daemons
  - Verifies capabilities
    - Export, mount, and verify

# Environmental Setup

- Time synchronized with NTP
  - Not required for DH, Kerberos V5 2.0, but…
  - /etc/rc.config.d/netdaemons
    - Defines NTP server
  - /etc/ntp.conf
    - Defines server commands
  - /sbin/init.d/xntpd
    - Startup/Shutdown NTP

Section 2

# Diffie-Hellman review

# Diffie-Hellman (DH)

- Public Key Authentication Method
  - Common key created by Public/Private keys
    - Each platform know the other's public key
    - Messages authorized by conversation key
    - Time synchronized (usually with NTP)
  - Public Key database maintained through NIS (in this case)
- Not a data encryption method
  - Keys and tokens are encrypted; data is not

- Keys defined and kept on NIS Server
  - newkey defines private and public keys
  - NIS handles distribution
- Keys defined locally if not using NIS
  - Public Key database must be built and copied to all users
- Multiple Roles possible for NIS Server
  - NIS Server could also be NFS Server or NFS Client

# NIS Server

- Setup required
  - ypinit, ypbind, nis.server, ypwhich
- Keys defined and stored
  - newkey defines private and public keys
  - keyserv handles private key lookup and transfer
    - Private keys kept in Public Key database
    - keyserv retrieves private key and stores it locally
    - keyserv handles password locally prior to key retrieval

# DH/NIS Client

- Bind to NIS Server for public keys
  - publickey lookup entry in /etc/nsswitch.conf
    - “publickey: nis files”
- Retrieve and store keys
  - keyserv
    - activates lookup in public key database
  - keylogin -r
    - extracts private key from database
    - writes private key into /etc/.rootkey

# Setup Sequence

- Initialize the NIS Master Server
  - ypinit –m #review /etc/rc.config.d/namesvrs
  - Stop and Start YP
    - /sbin/init.d/nis.server stop
    - /sbin/init.d/nis.server start
  - Bind and verify
    - /usr/lib/netsvc/yp/ypbind
    - ypwhich; sleep 10; ypwhich

# Setup (continued)

- Initialize NIS and DH on the client
  - Kill yp* processes
    - /sbin/init.d/nis.client stop
  - Kill keyserv process (if it exists)
  - domainname nishpnfs0xx  #NIS Server domain
  - ypinit –c
- Bind to NIS Server and launch keyserv
  - /usr/lib/netsvc/yp/ypbind
  - /usr/sbin/keyserv; sleep 5; ypwhich

# Setup (continued)

- Create key for Client on NIS Server
  - newkey –h hpnfs0yy –s [ files | nis ]
- Rebuild NIS Server's public key database from anywhere
  - Not needed  if "-s nis" used above
  - cd /var/yp; ./ypmake publickey
  - Rebuild actually happens on NIS Server
- Store private key on NFS client
  - keylogin –r
  - "Wrote secret key in /etc/.rootkey"

# Setup (continued)

- Enable NFS usage of DH
  - Uncomment dh entry in /etc/nfssec.conf
- Export from Server with desired security flavor
  - share –o sec=dh  /usr
- Mount with same security flavor
  - mount  –o sec=dh  hpnfs999:/usr  /wherever

Section 3

# Kerberos V5 Review

# Kerberos V5

RFCs 1510(krb5), 1964(gssapi)

- Developed originally at MIT
  - http://web.mit.edu/kerberos/www/
- Kerberos Server provides 3d Party Authentication via tickets
  - All Kerberos Clients assume Server is trusted
- NFS Servers & NFS Clients are all Kerberos Clients
  - Clients use encrypted keys and tickets

- Kerberos modes available
  - krb5 provides authentication
  - krb5i adds integrity via checksums to krb5
  - krb5p adds data encryption to krb5i
- Encryption services used
  - DES-MAC MD5 Integrity service
  - DES Encryption service

# Kerberos V5 Server

- Product Number T1417AA
  - www.software.hp.com
  - /opt/krb5/sbin/krbsetup
- Kerberos V5 Server version 2.0
  - NTP synchronization not required, but it helps
- Kerberos V5 Server utilities
  - /opt/krb5/bin/kadmin - text tool
  - /opt/krb5/admin/kadminl_ui - X tool

# Kerberos V5 Client

- krb5.conf configuration file needed
  - /etc/krb5.conf on all clients
  - Sample krb5.conf file available from http://docs.hp.com/hpux/onlinedocs/T1417-90006/T1417-90006.html
- Time Synchronized
  - NTP synchronization helpful
- Cannot also be a Kerberos Server

# V5 Client (Continued)

- Key Table needed
  - /etc/krb5.keytab default on all Kerberos Clients
  - Keys extracted with kadmin on Kerberos Server
- kinit command
  - Builds tickets with time limits (< 24 hours)
  - The first ticket is the TGT – the Ticket Granting ticket
- klist command
  - Gives tickets currently active on client

# V5 Client (Continued)

- Service enabling files
  - /etc/services and /etc/nfssec.conf
- Domain defined by DNS
  - /etc/resolv.conf
  - passwd entry in /etc/nsswitch.conf set to dns
- GSS Setup
  - /usr/lib/netsvc/gss/gssd (/etc/inetd.conf)
  - /etc/gss/gsscred.conf (set to “files”)
  - “gsscred –m krb5_mech –a”  builds table

# Setup Sequence

- Define four principal keys on Kerberos Server
  - root, host, nfs, sample
    - add host/hpnfs190.cup.hp.com (for example)
- Extract keys for Kerberos clients
  - kadmin extract command for client keys
  - Copy to /opt/krb5.keytab on all Kerberos Clients
- Have DNS or NIS defined on clients
  - /etc/resolv.conf for DNS
  - ypinit for NIS

# Setup (continued)

- Acquire & verify initial ticket (TGT)
  - kinit root #will ask for passwd
  - klist #verify tickets
- Verify gssd daemon & configuration
  - /usr/lib/netsvc/gss/gssd & /etc/inetd.conf
- Enable NFS usage of Kerberos
  - Uncomment krb entries in /etc/nfssec.conf

# Setup (continued)

- Export with desired security mode
  - share –o sec=krb5 /usr
    - krb5i
    - Krb5p
- Mount with correct security mode
  - mount –o sec=krb5 hpnfs0xx:/usr /wherever
    - krb5i
    - krb5p

# GSSAPI Sample Program

- Uses ‘sample’ key defined for clients
  - sample/hpnfs0xx.cup.hp.com
  - sample/hpnfs0yy.cup.hp.com
- 1st Kerberos Client looks for message
  - /usr/contrib/gssapi/sample/gss-server sample
    - hpnfs0xx displays note and waits for message
- Generate message from 2nd client
  - /hpsample/gss-client hpnfs0xx sample@hpnfs0xx “hi”
- 1st client receives message

# Managing Security Setup Issues for SecureNFS

Questions? Contact me via email, or cell phone

Tom McNeal

trmcneal@comcast.net

(650) 906-0761