NetBSD, Kerberos & AFS: From Zero to Distributed Filesystem in N Easy Steps

Tracy Di Marco White

Thomas L. Kula

Iowa State University of Science and Technology

AFS & Kerberos Best Practices Workshop 2005

http://kula.public.iastate.edu/talks/afs-bpw-2005/

Version 2006020700

Overview

- Using NetBSD as client and server systems
- Using Heimdal as a KDC
- Using OpenAFS to provide AFS service
- Using Arla to provide AFS clients

Why both OpenAFS and Arla?

- Arla provides a functioning AFS client for NetBSD systems, but some of the functions of `bos`, `vos`, etc, are not implemented, and the server `milko` is really only experimental.

- OpenAFS provides all the functions of `bos`, `vos`, etc. and all of the servers, but does not have a functioning client for NetBSD.

Why NetBSD[1]

- A small yet full-featured core operating system
- Emphasis on clean code and correct design

[1] http://www.netbsd.org/Misc/features.html

Why NetBSD

- 48 stable ports for the current release, in 17 CPU architectures, from a single code tree available via snapshots or anonymous CVS access [1]

- Over 5400 third-party applications in the Packages Collection [2]

[1] `http://www.netbsd.org/Ports/`

[2] `http://www.netbsd.org/Documentation/software/packages.html`

Why Heimdal?

- It is included in the NetBSD base install
- It is also included in the pkgsrc system
- Can also emulate enough of a kaserver to provide authentication to clients that require that
- Included `ktutil` can convert between v5 keytabs and AFS KeyFiles

Prepping NetBSD

- Install the NetBSD kernel source, needed during Arla compilation

  `http://www.netbsd.org/guide/en/chap-kernel.html`

- Install NetBSD pkgsrc, needed to create the OpenAFS package

  `http://www.netbsd.org/Documentation/pkgsrc/getting.html`

Prepping NetBSD

- As both Kerberos and AFS require close time synchronization between clients and servers, you will want to configure NTP

- `http://www.netbsd.org/guide/en/chap-misc.html`

- It is recommended that you set both `ntpdate=YES` and `ntpd=YES` in `/etc/rc.conf`, since if your date is way off, ntpd will give up and not sync it.

Configuring Heimdal as your KDC

- If you already have a KDC, you can use that, as long as you can add various principals to it and extract keys from it

Configuring Heimdal as your KDC

The entire authentication system depends on the trustability of the KDC(s), so anyone who can compromise system security on a KDC system can theoretically compromise the authentication of all users of systems depending on the KDC. Again, no amount of cleverness in the design of the Kerberos system can take the place of solid system administration practices employed in managing the Kerberos KDC(s). [1]

[1] `http://www.oit.duke.edu/~rob/kerberos/kerbasnds.html`

Configuring Heimdal as your KDC

Set up DNS SRV Records

- Doing this allows clients to talk to your KDC without having to have a local krb5.conf file
- `draft-ietf-krb-wg-krb-dns-locate`
- RFC 2782

Configuring Heimdal as your KDC

Set up DNS SRV Records

```
$ORIGIN example.com.
_kerberos._udp IN SRV 0 0 88 kerberos-1.example.com.
_kerberos._udp IN SRV 1 0 88 kerberos-2.example.com.
_kerberos._tcp IN SRV 0 0 88 kerberos-1.example.com.
_kerberos._tcp IN SRV 1 0 88 kerberos-2.example.com.
_kpasswd._udp IN SRV 0 0 464 kerberos-1.example.com.
_kerberos-adm._tcp IN SRV 0 0 749 kerberos-1.example.com.
_kerberos IN TXT "EXAMPLE.COM"
```

Configuring Heimdal as your KDC

Configuring `/etc/krb5.conf`

- NetBSD's default installed kerberos is disabled unless `/etc/krb5.conf` exists

- If you do not have DNS SRV records you will have to configure `/etc/krb5.conf`

Configuring Heimdal as your KDC

Configuring /etc/krb5.conf

```
[libdefaults]
    default_realm = EXAMPLE.COM
[appdefaults]
    afs-use-524 = no
    afslog = yes
[realms]
    EXAMPLE.COM = {
        kdc = kerberos-1.example.com
        kdc = kerberos-2.example.com
        admin_server = kerberos-1.example.com
        kpasswd_server = kerberos-1.example.com
    }
```

Configuring Heimdal as your KDC

Create Master Key

- Used to encrypt the keys in the database
- Do not have to remember this password, stashed in `/var/heimdal/m-key`
- `kstash`

Configuring Heimdal as your KDC

Initialize database

```
kadmin -l
kadmin> init EXAMPLE.COM
Realm max ticket life [unlimited]:
Realm max renewable ticket life [unlimited]:
```

Configuring Heimdal as your KDC

Create KDC host principal

```
kadmin -l
kadmin> add --random-key host/kerberos-1.example.com
Realm max ticket life [unlimited]:
Realm max renewable ticket life [unlimited]:
Attributes []:
kadmin> ext_keytab host/kerberos-1.example.com
```

Configuring Heimdal as your KDC

Add user principals

- Add `alice` (for normal use)
- Add `alice/admin` (for Kerberos administrative tasks)
- Add `alice/afs` (for AFS administrative tasks)

Configuring Heimdal as your KDC

Add user principals

```
kadmin -l
kadmin> add alice
Max ticket life [1 day]:
Max renewable life [1 week]:
Attributes []:
Password:
Verifying password - Password:
```

- Also add `alice/admin` and `alice/afs`

Configuring Heimdal as your KDC

Starting KDC at startup

- To `/etc/rc.conf` add the line `kdc=YES`
- To immediately start kdc, run `/etc/rc.d/kdc start`

Configuring Heimdal as your KDC

Running kadmind and kpasswd

- Edit `/etc/inetd.conf` and uncomment the following lines

```
kerberos-adm stream tcp nowait root /usr/libexec/kadmind kadmind

kpasswdd dgram udp wait root /usr/libexec/kpasswdd kpasswdd
```

- Restart `inetd:` `/etc/rc.d/inetd restart`

Configuring Heimdal as your KDC

Setting administrative ACL

- Edit /var/heimdal/kadmind.acl

  ```
  alice/admin@EXAMPLE.COM all
  ```

Configuring Heimdal as your KDC

Making keytabs for other machines

- `kadmin -p alice/admin`
  `kadmin> add --random-key host/`*`hostname.example.com`*
  `kadmin> ext_keytab -k /tmp/krb5.keytab-`*`hostname`*
  `host/`*`hostname.example.com`*

- Copy `/tmp/krb5.keytab-`*`hostname`* to `/etc/krb5.keytab` on remote machine

- Should be owned by `root:wheel` and mode 700

Configuring Heimdal as your KDC

Making afs principal and `KeyFile`

- `kadmin -p alice/admin`
  `kadmin> add --random-key afs/example.com`
  `kadmin> ext_keytab -k /tmp/afsv5key afs/example.com`

- Create the file `/usr/afs/etc/ThisCell` and put in it one line:
  `example.com`

- `ktutil copy /tmp/afsv5key AFSKEYFILE:/tmp/KeyFile`

- This is used by all AFS servers to authenticate themselves

Installing OpenAFS

- `cd /usr/pkgsrc/net/openafs`
- `make && make package`
- You can copy the binary package in `/usr/pkgsrc/packages/All` to other machines and install by doing `pkg_add openafs`, as long as the destination machines have the same architecture and OS version as the build machine

Installing the initial AFS database server

- Install OpenAFS (lets pretend the machine is called afs-1.example.com)

- Copy `KeyFile` created above to `/usr/pkg/etc/openafs/server/KeyFile`

- Copy over `/etc/krb5.conf` file

- Create and install a keytab containing `host/afs-1.example.com` on this machine

Installing the initial AFS database server

- Configure ntp and ntpdate and set them to start automatically on boot

- Make the file `/usr/afs/etc/ThisCell` contain the line `example.com`

Installing the initial AFS database server

Starting Basic OverSeer Server

- The bosserver is the process that oversees all other AFS server processes

- `/usr/pkg/sbin/bosserver -noauth`

- *Note:* this starts `bosserver` with no authentication at all, which is necessary since the protection database doesn't know about anyone at all

Installing the initial AFS database server

Setting cell name

- `/usr/pkg/bin/bos setcellname afs-1.example.com example.com -noauth`

Installing the initial AFS database server

Create database processes

- `/usr/pkg/bin/bos create afs-1.example.com buserver simple /usr/pkg/libexec/openafs/buserver -noauth`
- `/usr/pkg/bin/bos create afs-1.example.com ptserver simple /usr/pkg/libexec/openafs/ptserver -noauth`
- `/usr/pkg/bin/bos create afs-1.example.com vlserver simple /usr/pkg/libexec/openafs/vlserver -noauth`

Installing the initial AFS database server

Create initial pts entries

- `/usr/pkg/bin/pts createuser -name alice -cell example.com -id` *somenumber* `-noauth`

- `/usr/pkg/bin/pts createuser -name alice.afs -cell example.com -id` *anothernumber* `-noauth`

- You can leave out `-id` *somenumber* and `-id` *anothernumber* if you don't care what the user's pts id number is

Installing the initial AFS database server

Adding afs principals to the `system:administrators` list

- `/usr/pkg/bin/pts adduser alice.afs system:administrators -cell example.com -noauth`

- *Note:* Even though you created the v5 principal `alice/afs` and will be using only v5 kerberos tickets to get tokens, the afs side still knows this user as `alice.afs`

Installng the initial AFS database server

Create SUsers

- Each AFS server has a list of users who can perform privileged operations on it

- `/usr/pkg/bin/bos adduser afs-1.example.com alice.afs -cell example.com -noauth`

Installing the inital AFS database server

Restart bosserver with authentication

- `/usr/pkg/bin/bos shutdown afs-1.example.com -noauth`
- `ps ax | grep bosserver`
- `kill -HUP` *pid-of-bosserver*
- `/usr/pkg/sbin/bosserver`

Installing the initial AFS database server

Automatically starting bosserver

- `cp /usr/pkg/share/examples/rc.d/bosserver /etc/rc.d/bosserver`
- `chmod 555 /etc/rc.d/bosserver`
- Add `bosserver=YES` to `/etc/rc.conf`

Installing the initial AFS file server

- The initial file server can be on the same machine as the initial database server

Installing the initial AFS file server

Preliminary Setup

- For the sake of discussion, we are going to set up afs-2.example.com as our first AFS file server

- If your first AFS file server is on your initial database server, you can skip all of the preliminary setup steps

Installing the initial AFS file server

Preliminary Setup

- Install OpenAFS
- Copy same `KeyFile` used on afs-1 to `/usr/pkg/etc/openafs/server/KeyFile`
- Copy over `/etc/krb5.conf`
- Configure and set ntp and ntpdate to start automatically on boot

Installing the initial AFS file server

Preliminary Setup

- Create and install a keytab containing `host/afs-2.example.com` on this machine

- Copy `CellServDB` and `ThisCell` from `/usr/pkg/etc/openafs/server` on `afs-1.example.com` to the same location on this machine

Installing the initial AFS file server

Preliminary Setup

- Make the file `/usr/afs/etc/ThisCell` contain the line `example.com`

- Configure bosserver to start automatically on boot, and start it

Installing the initial AFS file server

Create file server processes

- You will need to have at least one partition to store afs files in. These partitions must be mounted at / and be called `/vicepa` through `/vicepzz` (although you can only have up to 255 partitions)

Installing the initial AFS file server

Create file server processes

- `/usr/pkg/bin/bos create afs-2.example.com fs fs`
  `/usr/pkg/libexec/openafs/fileserver`
  `/usr/pkg/libexec/openafs/volserver`
  `/usr/pkg/libexec/openafs/salvager`
  `-cell example.com -localauth`

Installing the initial AFS file server

Create root.afs

- The volume `root.afs` represents what is in `/afs`
- `/usr/pkg/sbin/vos create afs-2.example.com /vicepa root.afs -localauth`

Installing the initial AFS file server

Create root.cell

- The volume `root.cell` represents what is in the top level of your cell (i.e. `/afs/example.com` in this example)

- `/usr/pkg/sbin/vos create afs-2.example.com /vicepa root.cell -localauth`

Installing the Arla client

- Currently, the arla client in NetBSD pkgsrc is a little old
- And since the arla client is very dependent on which kernel you are running, you want to build it on each machine you have

Installing the Arla client

Compiling Arla

- Download `arla-0.39` from the Arla Project Homepage: `http://www.stacken.kth.se/projekt/arla/`

Installing the Arla client

Compiling Arla

- `./configure --prefix=/usr/local`
  `--with-krb4-lib=/usr/lib`
  `--with-krb4-include=/usr/include/kerberosIV`
  `--with-krb5-lib=/usr/lib`
  `--with-krb5-include=/usr/include/krb5`
  `--with-sys=/usr/src/sys`

- make && make install

Installing the Arla Client

Configuring Arla

- Add the lines in your database server's CellServDB file into `/usr/local/etc/CellServDB`
- Make `/usr/local/etc/ThisCell` read `example.com`
- Change items in `/usr/local/etc/arla.conf` as you see fit

Installing the Arla Client

Configuring LKM

- Arla uses a Loadable Kernel Module to provide the interface between the arla client and the kernel

- Add the following on one line to your `/etc/lkm.conf`:
  `/usr/local/bin/nnpfs_mod.o - nnpfs_mod /usr/local/sbin/nnpfs_makedev /var/db/nnpfs_sym BEFOREMOUNT`

Installing the Arla client

```
#!/bin/sh
#
# PROVIDE: arlad
# REQUIRE: beforemountlkm
.  /etc/rc.subr
name="arlad"
rcvar=$name
command="/usr/local/libexec/$name"
command_args="-z /dev/nnpfs0"
start_precmd="/usr/local/sbin/mount_nnpfs /dev/nnpfs0 /afs"
stop_postcmd="/sbin/umount /afs"
required_files="/dev/nnpfs0"
required_dirs="/afs"
load_rc_config $name
run_rc_command "$1"
```

- Install this as `/etc/rc.d/arlad`
- `chmod 555 /etc/rc.d/arlad`

Installing the Arla Client

Configuring LKM

- In `/etc/rc.conf` add the lines `lkm=YES` and `arlad=YES`
- If your `/usr` directory is on a separate partition, add the following to `/etc/rc.conf`:
  `critical_filesytems_local="/var /usr"`

Installing the Arla client

- `mknod /dev/nnpfs0 c 165 0`
- `mkdir /afs`
- Restart

Configuring the top-level of AFS

Set permissions for /afs

- Get `alice/afs` tickets/tokens (`kinit alice/afs`)
- `/usr/pkg/bin/fs setacl /afs system:administrators rlidwka`
- `/usr/pkg/bin/fs setacl /afs system:anyuser rl`

Configuring the top-level of AFS

Create mountpoint for root.cell

- `/usr/pkg/bin/fs mkmount /afs/example.com root.cell`
- `/usr/pkg/bin/fs setacl /afs/example.com system:administrators rlidwka`
- `/usr/pkg/bin/fs setacl /afs/example.com system:anyuser rl`

Configuring the top-level of AFS

Create mountpoint for root.cell

- You can create read-only copies of volumes in AFS, and replicate them on different afs file servers

- By default, AFS will chose a read-only version of a volume, so if you need to make changes to a replicated volume, you need some way of getting to the read-write version of the volume

Configuring the top-level of AFS

Create mountpoints for root.cell

- `/usr/pkg/bin/fs mkmount /afs/.example.com root.cell -rw`

- Allows you to explicitly access the read-write version of replicated volumes by going through `/afs/.example.com`

Installing the initial AFS file server

Replicating root.afs and root.cell

- Adding a read-only copy of volumes on the server that contains the read-write copy of the volume costs nothing
- If you add other file servers you will want to have read-only volumes replicated on at least another machine
- In order to get to any volume, you need to be able to get to any volumes above its mount-point

Installing the initial AFS file server

Replicating root.afs and root.cell

- `/usr/pkg/sbin/vos addsite afs-2.example.com /vicepa root.cell`
- `/usr/pkg/sbin/vos addsite afs-2.example.com /vicepa root.afs`
- `/usr/pkg/sbin/vos release root.afs`
- `/usr/pkg/sbin/vos release root.cell`

Other things

Having ssh get afs tokens on login

- While the OpenSSH that comes by default does not currently support this, the OpenSSH in pkgsrc does

- Installing that and then adding the following to your `/usr/pkg/etc/ssh/sshd_config` file will get you afs tokens on login:

```
KerberosAuthentication yes
KerberosGetAFSToken yes
```

Other Considerations

- Adding a slave KDC
- Adding additional database or file servers
- Dealing with clients that expect to talk to a kaserver

Acknowledgements

Thanks to:

- The NetBSD, Heimdal, Arla and OpenAFS projects for producing high-quality software

- Ty Sarna for presentation testing

Acknowledgements

Thomas would like to thank

- Stomping Grounds in Ames, IA, for providing free wireless access and high-quality caffeine

Acknowledgements

Tracy would like to thank

- Thomas for producing the giant PDF file

NetBSD, Kerberos & AFS: From Zero to Distributed Filesystem in N Easy Steps

Tracy Di Marco White

Thomas L. Kula

Iowa State University of Science and Technology

AFS & Kerberos Best Practices Workshop 2005

http://kula.public.iastate.edu/talks/afs-bpw-2005/