Back to Index

NAME

auristor_migration - Migrating to the AuriStor File System from OpenAFS

DESCRIPTION

The **AuriStor File System** is an **AFS3**-compatible distributed file system. **AFS3** uses administrative domain units called *cells*. An existing installation of **OpenAFS** can be migrated in place without flag days to a fully functional **AuriStor** cell.

- **AuriStor Cache Managers** are fully compatible with **OpenAFS** services and can be deployed at any time.
- **AuriStor** services are backward compatible with **OpenAFS** and **IBM AFS 3.6** clients.
- **AuriStor File Servers** can be deployed in a cell whose database services are provided by **OpenAFS**. In this configuration, the **AuriStor File Servers**' advanced security features are disabled.
- **AuriStor Database Services** can be deployed as a clone in a cell whose coordinator (or synchronization site) is provided by **OpenAFS**. In this mode all of the **AuriStor** protocol enhancements are disabled.

AuriStor Database Services support **OpenAFS 1.6** compatible formats for the **protection database** and **location database**. As a result, **OpenAFS** database servers can be upgraded in-place.

AuriStor File Servers stores volume data in an incompatible proprietary format. Thus, **OpenAFS** file servers cannot be upgraded in-place. Instead, existing volume data must be migrated onto the **AuriStor File Server** using the **vos** command suite.

TERMINOLOGY

The **AuriStor File System** uses different terminology to describe its services than **OpenAFS** system administrators are used to.

Services not Databases

> The **OpenAFS** documentation discusses various databases such as the *Volume Location Database* and *Protection Database*. In **AuriStor** the architecture is similar but they are referred to as the *Location Service* and the *Protection Service*. That a particular service's data is stored in a database is an implementation detail.

Location Service not Volume Location Service

> In the **AuriStor File System** the *Location Service* manages a broader set of location and key data than simply volumes. As such it has been renamed.

Coordinators not Synchronization Sites

> In the **AuriStor File System** the Ubik distributed database instance that **OpenAFS** documentation referred to as a *synchronization site* is now called the *coordinator*.

PROCEDURE

Start by gathering a list of servers to be updated. The list of **Location Servers** for a cell can be collected with the **bos listhosts** command.

A list of **File Servers** can be collected with the **OpenAFS vos listaddrs -printuuid** command or the **AuriStor vos listfs** command. Any **File servers** that are also **Location Servers** need to be vacated before conversion. Any `RW` volumes can be migrated to another machine with the **vos move** command. For `RO` volumes the **vos remove** command can be used to remove the replication site after other sites for replicas are configured with **vos addsite** and populated with **vos release**.

Organizations are encouraged to begin deploying **AuriStor** by first adding an **AuriStor File Server** to the existing cell and then populate it with volume data. This will provide the organization with a taste of the performance and scalability improvements that **AuriStor** offers above and beyond the capabilities of **OpenAFS**.

It is possible for **AuriStor Location Servers** and **AuriStor Protection Servers** to be added to an existing cell in a read only mode. This permits evaluating organizations to measure the performance benefits of the **AuriStor** services versus their **OpenAFS** counterparts without risking the introduction of **AuriStor** extended data into the cell's databases.

The file */etc/yfs/server/yfs-server.conf* will need to be configured for the cell. Additionally, if the *cellservdb.conf* included in the installed packages does not include configuration for the cell to be upgraded, it will be necessary to provide it in the */etc/yfs/server/yfs-server.conf* file.

Additionally, the */etc/yfs/server/BosConfig* file will need to reflect the paths to the newly-installed server binaries. Note that in **AuriStor**, options to all servers can be configured via the */etc/yfs/server/yfs-server.conf* file. The use of command line switches is discouraged.

Once the organization is prepared to commit to **AuriStor** and take advantage of the enhanced security capabilities and name space limits it is time to upgrade the database services to **AuriStor**.

The safe method of migrating db servers from OpenAFS to AuriStor is to:

1. modify the **OpenAFS** configuration such that all but one of the OpenAFS servers are clones. This ensures that the coordinator is known. All servers will need to be restarted for this to take effect.

2. Convert the **OpenAFS** clones one at a time to **AuriStor**

    - Force each server to operate as a readonly clone with the **clone = yes** option in the **databases** configuration for each **server** entry.

    - Enable the **afscompat** option either in the **[defaults]** stanza or for each service you are converting (e.g. **[vlserver]**, **[ptserver]**, **[buserver]**).

    At this point the cell will be unable to process write operations.

3. Convert the coordinator from **OpenAFS** to **AuriStor** without the **afscompat** option.

4. Remove the **afscompat** option from the clones.

    At this point the cell can once again process write operations.

5. Remove the clone designation from servers that are not clones.

The examples below assumes a cell named `your-cell-name.com` is to be upgraded to **AuriStor**.

CREATING THE SERVER CONFIG FILE

OpenAFS Servers use the *CellServDB*, *ThisCell*, *NetRestrict*, *NetInfo*, *krb.excl* and *krb.conf* files to configure parameters which are set in */etc/yfs/server/yfs-server.conf* on **AuriStor Servers**. Additionally, all command line parameters to **AuriStor** daemons should be set in the configuration file.

The **[defaults]** section of the server configuration file holds all of the information previously obtained from these files:

NetInfo

The contents of *NetInfo* should be specified as a list of addresses in the **netinfo** stanza.

NetRestrict

The contents of *NetRestrict* should be specified as a list of addresses in the **netrestrict** stanza.

ThisCell

The contents of *ThisCell* should be specified in the **thiscell** stanza.

CellServDB

The *CellServDB* data for the cell this server is a member of should be specified in the **databases** stanza. The **databases** stanza should contain a **servers** stanza that lists lines for the address, and optionally, clone status, for each server.

```
databases = {
    servers = {
        db1.your-cell-name.com = {
            address = 10.0.0.1
        }
        db2.your-cell-name.com = {
            address = 172.16.150.1
        }
        db3.your-cell-name.com = {
            address = 192.168.5.2
        }
        db4.your-cell-name.com = {
            address = 192.168.5.1
            clone = yes
        }
    }
}
```

Additionally, the **fileserver** and **volserver** processes, like any **AuriStor** client process, do not use the database settings, so a **[cells]** section must also be provided for the local cell.

```
[cells]
    your-cell-name.com = {
        description = "My test cell"
        servers = {
            db1.your-cell-name.com = {
                address = 10.0.0.1
            }
            db2.your-cell-name.com = {
                address = 172.16.150.1
            }
            db3.your-cell-name.com = {
                address = 192.168.5.2
            }
            db4.your-cell-name.com = {
                address = 192.168.5.1
            }
        }
    }
```

The **[cells] your-cell-name.com** block differs from the **databases servers** block as follows:

- The **databases servers** block details must include only the addresses which are used by *Ubik* for database management operations; and the server name is used for descriptive purposes only.
- The **[cells] your-cell-name.com** block can include any reachable address and the server names should be valid DNS host names.

If the **[cells] your-cell-name.com** block is not provided the **fileserver** and **volserver** will use DNS SRV record lookups when other services within the cell must be contacted. The *yfs-server.conf* has further information.

The **[kerberos]** section of the server configuration file holds all of the information previously obtained from these files:

krb.conf

The contents of *krb.conf* should be specified as a list of realms in the **local_realms** stanza.

krb.excl

The contents of *krb.excl* should be specified as a list of principals in the **foreign_principals** stanza.

Each service also can have options specified in an appropriately-named section, e.g. the `bosserver` can have options specified in the **[bosserver]** section. Examples might include running in nofork mode, where the line **nofork = yes** would be included. The **databases** stanza may also be configured per-service.

It is important that the */etc/yfs/server/yfs-server.conf* file and the directory containing it be owned by the user the **AuriStor** server processes will run as. Typically this user is named `yfsserver` and will be created when the **AuriStor** binary packages are installed if your platform supports running **AuriStor** daemons as non-root users.

CREATING THE KEYFILE

The existing **afs** key must be installed on the **AuriStor** servers. **AuriStor** servers store the **afs** key data in two files:

KeyFile

The *KeyFile* is compatible with **IBM AFS** and **OpenAFS** cells. Prior to OpenAFS 1.6.5 the **afs** key was always a 56-bit parity checked key compatible with the deprecated U.S. Data Encryption Standard (DES). Keys compatible with the DES encryption types, *des-cbc-crc*, *des-cbc-md4* and *des-cbc-md5*, are referred to as *rxkad* keys. The *KeyFile* file stores *rxkad* keys.

KeyFileExt

Beginning with the OpenAFS 1.6.5 release the **afs** key could be any encryption type accepted by the Kerberos v5 standard including *aes256-cts-hmac-sha1-96*, *aes128-cts-hmac-sha1-96*, *des3-cbc-sha1*, and *arcfour-hmac*. All of the non-DES **afs** keys are referred to as *rxkad_krb5* keys.

The **AuriStor File System** also adds a new key type known as *yfs-rxgk*. The *KeyFileExt* file stores both *rxkad_krb5* and *yfs-rxgk* keys.

If the cell uses an *rxkad* key it is strongly recommended that the cell be upgraded to use one or more *rxkad_krb5* keys before converting the cell to the **AuriStor File System**. The *rxkad* key can be broken by brute force in under a day using publicly available cloud services. The **How To Rekey** guide provides instructions to convert an **OpenAFS** cell to use the more secure *rxkad_krb5* keys. This should be done before continuing to generate the *KeyFileExt*.

If the cell cannot be upgraded to use *rxkad_krb5* keys, then the **OpenAFS** *KeyFile* should be copied to */etc/yfs/server/KeyFile* on each **AuriStor** server.

If *rxkad_krb5* keys are used by the cell, they must be installed into the */etc/yfs/server/KeyFileExt* file using the **AuriStor** version of **asetkey**. Most OpenAFS cells running version 1.6.5 or later will be using a krb5 keytab file. Some installations might have an **OpenAFS** *KeyFileExt* file. If neither an **OpenAFS** *KeyFileExt* file nor a krb5 keytab with the existing key is available, a krb5 keytab must be generated. How this is done depends on which Kerberos implementation is in use.

MIT Kerberos

To avoid changing the key when using MIT Kerberos, you will need to reconstruct the keytab from the existing key. Use **asetkey list** to get the key for the newest kvno:

```
# asetkey list
kvno    4: key is: d34dbeefd34dbeef
kvno    3: key is: baddc4febaddc4fe
All done.
```

Then, use ktutil to create a keytab:

```
# ktutil
ktutil:  add_entry -key -p afs/your-cell-name.com@EXAMPLE.ORG \
         -k 4 -e des-cbc-crc
Key for your-cell-name.com@EXAMPLE.ORG (hex): d34dbeefd34dbeef
ktutil: write_kt /etc/yfs/server/rxkad.keytab
ktutil: exit
```

Heimdal

This can be done via the Heimdal **ext_keytab** command.

```
# ext_keytab -k /etc/yfs/server/rxkad.keytab \
        afs/your-cell-name.com@EXAMPLE.ORG
```

As the desired result is a *KeyFileExt* file, you will need to convert your keytab. First, generate a list of all the key types it contains via *ktutil*.

MIT Kerberos

```
# ktutil
ktutil:  rkt /etc/yfs/server/rxkad.keytab
 ktutil:  list -e
slot KVNO Principal
---- ---- ---------------------------------------------------------------------
   1    4 afs/your-cell-name.com@EXAMPLE.ORG (aes256-cts-hmac-sha1-96)
   2    4 afs/your-cell-name.com@EXAMPLE.ORG (aes128-cts-hmac-sha1-96)
   3    4 afs/your-cell-name.com@EXAMPLE.ORG (des3-cbc-sha1)
   4    4 afs/your-cell-name.com@EXAMPLE.ORG (arcfour-hmac)
```

Heimdal

```
# ktutil -k /etc/yfs/server/rxkad.keytab  list
/etc/yfs/server/rxkad.keytab:

Vno  Type                    Principal                                      Aliases
  4  aes256-cts-hmac-sha1-96  afs/your-cell-name.com@EXAMPLE.ORG
  4  aes128-cts-hmac-sha1-96  afs/your-cell-name.com@EXAMPLE.ORG
  4  des3-cbc-sha1            afs/your-cell-name.com@EXAMPLE.ORG
  4  arcfour-hmac-md5         afs/your-cell-name.com@EXAMPLE.ORG
```

All keys will need to be inserted with **asetkey**. The `all` key type will iterate over all supported keys and add them. For example, the above keys have a key version of `4`, so for each key, a kvno parameter of `4` would need to be specified to the **asetkey add** command.

```
# asetkey add rxkad_krb5 4 all \
  /etc/yfs/server/rxkad.keytab afs/your-cell-name.com@EXAMPLE.ORG
```

In a cell running an AuriStor Location Service, a cell-wide **yfs-rxgk** key is required in order to support AES256 encryption, protection of cache manager callback connections, and many other enhanced security features. This additional key is created with **asetkey add**. The **yfs-rxgk** key should be key version 1, and future keys can increment from there. **Do not** create the **yfs-rxgk** key when adding an AuriStor Server to a cell running an OpenAFS Location Service.

```
# asetkey add yfs-rxgk 1 aes256-cts-hmac-sha1-96 random
```

This *KeyFileExt* can then be distributed to all **location servers** (and to any server which will have the **AuriStor** server software installed).

It is important that the *KeyFileExt* file and the directory containing it be owned by the user the **AuriStor** server processes will run as. Typically this user is named `yfsserver` and will be created when the **AuriStor** binary packages are installed if your platform supports running **AuriStor** as non-root users.

BOSSERVER SETUP

The **OpenAFS bosserver**shares the same cell-wide keys as the **AFS** cell. The **AuriStor bosserver** is an independent service that is used to manage a single machine. As such, each **AuriStor bosserver** requires its own Kerberos service principal and keytab. This keytab should be installed in **/etc/yfs/server/bos.keytab**.

MIT Kerberos

This example creates a **bos** key for server `server.your-cell-name.com`:

```
# kadmin
kadmin: add_principal -randkey afs3-bos/server.your-cell-name.com@EXAMPLE.ORG
kadmin: ktadd -k /etc/yfs/server/bos.keytab \
        afs3-bos/server.your-cell-name.com@EXAMPLE.ORG
```

Heimdal

This example creates a **bos** key for server `server.your-cell-name.com`:

```
# kadmin
kadmin> add -r afs3-bos/server.your-cell-name.com@EXAMPLE.ORG
kadmin> ext_keytab -k /etc/yfs/server/bos.keytab \
        afs3-bos/server.your-cell-name.com@EXAMPLE.ORG
```

It is important that the **/etc/yfs/server/bos.keytab** file and the directory containing it be owned by the user the **AuriStor** server processes will run as. Typically this user is named `yfsserver` and will be created when the **AuriStor** binary packages are installed if your platform supports running **AuriStor** as non-root users.

LOCATION SERVERS

The **location servers** use a Kerberos key to authenticate clients, both users and machines. Authentication tokens for other servers are provided using the **yfs-rxgk** cell key.

RXGK KERBEROS KEY

A key for **yfs-rxgk** needs to be inserted in the Kerberos KDC. In order that services are able to be fully configured, the key should be created disabled.

Install as */etc/yfs/server/vl.keytab* on all the **location server**s.

MIT Kerberos

```
# kadmin
kadmin: add_principal -randkey -allow_tix
          yfs-rxgk/_afs.your-cell-name.com@EXAMPLE.ORG
kadmin: ktadd -k /etc/yfs/server/vl.keytab yfs-rxgk/_afs.your-cell-name.com
```

Heimdal

```
# kadmin
kadmin> add --attributes=+disallow-all-tix -r
          yfs-rxgk/_afs.your-cell-name.com@EXAMPLE.ORG
[...]
kadmin> ext_keytab --keytab=/etc/yfs/server/vl.keytab \
                     yfs-rxgk/_afs.your-cell-name.com
```

It is important that the */etc/yfs/server/vl.keytab* file and the directory containing it be owned by the user the **AuriStor** server processes will run as. Typically this user is named `yfsserver` and will be created when the **AuriStor** binary packages are installed if your platform supports running **AuriStor** as non-root users.

CREATING THE SERVER SUPERUSER LISTS

The **OpenAFS** *UserList* file will need to be replaced with a *UserListExt* file, which should be installed in */etc/yfs/server/UserListExt*. That file should contain the same list as the **OpenAFS** *UserList*, except with the Kerberos realm appended to the end of each user, after an `@` sign. For example,

```
alice.admin
bob.admin
bob
```

would become

```
alice.admin@YOUR-CELL-NAME.COM
bob.admin@YOUR-CELL-NAME.COM
bob@YOUR-CELL-NAME.COM
```

Alternately, because the **bosserver** has a key installed, it is possible to use local superuser authentication to configure the superuser list with the **AuriStor bos adduser** command. To add users `alice.admin` and `bob.admin` on server `server.your-cell-name.com`:

```
# bos adduser -server server.your-cell-name.com -user alice.admin@YOUR-CELL-NAME.COM \
  -user bob.admin@YOUR-CELL-NAME.COM -localauth
```

Regardless of which manner is chosen, this procedure will need to be done on **location servers** as well as on **file servers**.

Unlike **OpenAFS**, **AuriStor** restricts access to cell metadata that is not necessary for proper operation of clients. This includes **pts** memberships, volume statistics, and **bosserver** process information. By default the **restricted_query** option, supported by each **AuriStor** service, is set to *admin* which prevents users who are not on either the **UserListExt** or **ReaderList** from viewing metadata. Members of the **ReaderList** are granted read-only access to restricted metadata. This file can be configured using **bos adduser** by adding the **-type reader** specifier.

```
# bos adduser -server server.your-cell-name.com \
  -user mallory.admin@YOUR-CELL-NAME.COM -localauth -type reader
```

When **restricted_query** is set to *admin* automated processes such as monitoring tools will need to operate with access to valid authentication tokens. Alternatively, adding **restricted_query = anyuser** to the **[defaults]** or service specific configuration section will restore the **OpenAFS** behavior.

CREATING THE BOS CONFIGURATION

Because the **bosserver** has a key installed, it is possible to use local superuser authentication to configure the **bosserver** with the **bos create** command. This is the safest way to make sure the *BosConfig* file has the proper format and ownership. Any options should already be configured in the *yfs-server.conf*. file.

This would create the **location server** and **protection server** processes on server `server.your-cell-name.com`:

```
# bos create -server server.your-cell-name.com -instance vlserver \
             -type simple -cmd "/usr/libexec/yfs/vlserver" \
             -localauth
# bos create -server server.your-cell-name.com -instance ptserver \
             -type simple -cmd "/usr/libexec/yfs/ptserver" \
             -localauth
```

The **location server** and **protection server** processes should all be running. The **file server** should be upgraded as explained in the "FILE SERVERS" section before any data is moved to the server.

ENABLING RXGK

Once all the database servers have been upgraded to **AuriStor**, enable the rxgk key in the Kerberos KDC.

MIT Kerberos

```
# kadmin
kadmin: modprinc +allow_tix
          yfs-rxgk/_afs.your-cell-name.com@YOUR-CELL-NAME.COM
```

Heimdal

```
# kadmin
kadmin> modify --attributes=-disallow-all-tix
          yfs-rxgk/_afs.your-cell-name.com@YOUR-CELL-NAME.COM
```

FILE SERVERS

A list of **file server**s can be discovered with the **OpenAFS** command `vos listaddrs -printuuid` or the **AuriStor** command **vos listfs**. Each server will need a **/etc/yfs/server/bos.keytab** as described in the "BOSSERVER SETUP" section, as well as a copy of the **KeyFileExt** file.

Servers can be upgraded one at a time, but all volume data must be removed from a file server before the upgrade is performed. See **vos_move**.

Existing vice partitions can be reused, but

- must support POSIX extended attributes. On some Linux systems, enabling POSIX extended attributes requires mounting the partitions with the `user_xattr` option. See **fstab** and **mount** for additional details. The **mount** command will display the options a filesystem was mounted with.

- the ownership must be changed from `root` to `yfsserver`

  ```
  chown -R yfsserver:yfsserver /vicep*
  ```

- all files except for an optional *AlwaysAttach* must be removed

See **fileserver** for more details.

MAPPING THE LOCAL REALM TO THE CELL

For the simplest and most common case, the cell name and Kerberos realm name will match, and no configuration will need to be done.

In cases where the cell and realm names do not match, the **local_realms** configuration stanza can be used to map all principals in one (or more) realms to the same username in the cell.

```
[kerberos]
    local_realms = YOUR-CELL-NAME.COM
```

To disallow some principals, the **foreign_principals** stanza may be used. For instance, you may not want a particular admin principal to be able to authenticate to AuriStor.

```
[kerberos]
    foreign_principals = admin@YOUR-CELL-NAME.COM
```

CREATING THE BOS CONFIGURATION

Because the **bosserver** has a key installed, it is possible to use local superuser authentication to configure the **bosserver** with the **bos create** command. This is the safest way to make sure the *BosConfig* file has the proper format and ownership. Any options should already be configured in the *yfs-server.conf*. file.

This would create the fileserver process on server `fs1.your-cell-name.com`:

```
# bos create -server fs1.your-cell-name.com -instance dafs -type dafs \
             -cmd "/usr/libexec/yfs/fileserver" \
             /usr/libexec/yfs/volserver \
             /usr/libexec/yfs/salvageserver \
             /usr/libexec/yfs/salvager -localauth
```

The fileserver processes should all be started. Data can now be migrated onto the fileserver.

PRIVILEGE REQUIRED

Typically, local superuser `root` is required to install new packages. Unlike **OpenAFS**, **AuriStor** can be configured to run server daemons as non-superusers.

SEE ALSO

asetkey(8), BosConfig(5), bos_adduser(8), bos_create(8), bos_listhosts(8), bos.keytab(5), bosserver(8), fileserver(8), fstab(5), KeyFileExt(5), mount(8), ReaderList(5) UserListExt(5), vl.keytab(5), vos(1), vos_addsite(1), vos_listfs(1), vos_release(1), vos_remove(1), vos_move(1), yfs-server.conf(5), How To Rekey

COPYRIGHT

ACKNOWLEDGEMENTS

Back to Index