Final project report for "Account Tidying: AFS/homepages/groupspace" (478)

Project Description

Following on from the "Account Tidying project" (349), we needed to tidy account data not held in home directories.

This includes:

- ACLs: The IDs of deleted accounts needed to be removed from all AFS ACLs.
- homepages: The homepages of any deleted accounts needed to be removed.
- group space: We needed to keep track of group space, and when the user designated the owner* of a group space left, we needed a way to transfer ownership to an active user if the space was still required, or reclaim the space if it wasn't.

*The owner is often (but not always) the original requester of the space.

Project page

Project updates

Implementation

The project updates page explains why certain decisions were made. This document is a summary of the actions taken to complete the project as per the original project description.

ACLs:

Problem:
There wasn't a simple way to find and remove old IDs from AFS ACLs.

Solution:
It was decided to start logging ACLs, then to use these logs to find and remove IDs that no longer exist.

The following scripts were created:

- `/usr/bin/afsvolscan` - Runs on each of the AFS RW servers generating volscan logs and saving them to `/var/services-unit/volscan/`.
- `get_results` - Copies the volscan logs from the AFS servers to a directory "volscan_results" in your current directory so that the logs can be processed by `afsclean`. The script will fail if the directory volscan_results already exists or it's not run as root
- `afsclean` - Searches the **local** volscan logs for ACLs that contain IDs that no longer exist. It then mounts the required volumes and runs the fs cleanacl command on the ACLs of directories that need to be "cleaned".
- `find_id` - Searches the **remote** volscan logs for individual IDs or UUNs (or list of IDs/UUNs within a file). This might prove useful when we need to remove an account from ACLs before the account has officially expired. By default `find_id` will display every ACL the user features in, but there's also an option to display only the names of volumes that contain relevant ACLs. `afsquery` could then be used to see where the volume is mounted, and `fsr setacl /path/ uun none` could be run on each volume to remove the uun.

With the exception of `afsvolscan` (a copy of which is on each AFS RW server), these scripts are held in the Services Unit AFS space `/afs/inf.ed.ac.uk/group/services-unit/afsclean/`. The --help flag or running `perldoc script` for each script will display the various options and example usage.

Homepages:

Problem:
When users leave their informatics homepages need to be removed.

Solution:
We automatically delete empty homepages associated with deleted accounts, any that contain data are moved to one side and deleted at a later date.

See our Homepages archiving/deletion procedure.

Group Space:

Problem:
Whether or not a group space can be deleted when the "owner" leaves can be difficult to determine. Active users may still require the data within the space. We needed a way to flag up when the owner of a space was leaving so that either a new owner could be found, or the space could be deleted.

Solution:
An audit was required, and the information gathered needed to be maintained e.g added to when space is created or changed in some way, removed when deletions occur. It was decided to store this information in a Postgres database, and create scripts for AFS group space creation, as well as searching, updating, deleting and generating reports from the data held in the database.

The following scripts were created (see AFSManagementScripts for a more thorough guide):

- `afscreate` - Creates new AFS group space and adds all the information required to the database. It looks at the AFS partitions page for partitions marked as DEFAULT or DEFAULT NO RO, and figures out the best partitions to create the space on.
- `afsquery` - Searches the database for a specified group space, owner or institute.
- `afsupdate` - Updates the information held about a group space in the database.
- `afsdel` - Deletes a volume from AFS and updates the database. The information will remain in the database for a year. There after it's flagged for deletion.
- `afscheckup` - Reports group spaces owned by accounts that aren't "active", spaces that have passed their review date, spaces where access was removed over a year ago, spaces deleted a year ago, and volumes that exist in AFS but not the database.

The report generated by `afscheckup` (there's a cron running on pergamum) is emailed daily to the services-unit RT guy.

- If a group's owner isn't "active" this is flagged up in the report. We then attempt to find an alternative owner or establish whether the space can be deleted. If it appears that no one is using the space, and a new owner can't be found, we remove access to the group space by altering the ACLs (or file permissions in the case of NFS space), and update the access_removed date in the database using the `afsupdate` script.
- If a space has passed its review date in the database it's flagged up. We contact the owner to see whether the space is still required or an extension is needed.
  - If the owner doesn't reply, and it appears that no one is using the space, access can be removed using the steps above.
- If we remove access to a space, and record the date in the database, it will appear in the report again after a year has passed. The longer a space is inaccessible without anyone anyone noticing, the stronger the case for deletion.
- When we delete a space we leave the related data in the database for a year. This is the period the space will still be retrievable from backups after its deletion. Once a year has passed it's flagged up in the report again so the data can be removed from the database.
- There will be times when group space is created manually (i.e not using the afscreate script) and so doesn't appear in the database. Any AFS group space not found in the database will be flagged in the report so it can be added to the database manually.

These scripts are held in the co utils area. The `--help` flag or running `perldoc script` for each script will display the various options and example usage. Also see AFSManagementScripts.

The scripts read and write to the "spacedb" database held on pgresearch. Like all databases on pgresearch, it's backed up nightly (and kept for a week), and weekly backups are kept for a month.

As well as AFS space (which is usually prepended with `group.` or `sweb.`), the database also holds information about NFS space, prepended with "`nfs.`".

Results Summary

So far:

- 3,843,779 ACLs have been "cleaned".
- 5.75TB of space has been reclaimed.

Effort Spent

Approximately 252.5 hours spread over 19 months.

Maintenance

When a new AFS server is created, the `/usr/bin/afsvolscan` script needs to be copied on to the new server, and the following needs to be added to the machine's profile:

```
!cron.additions      mADD(volscan1 volscan2 volscan3)
!cron.add_volscan1   mSET(MAILTO=rarmstr1@inf.ed.ac.uk)
!cron.add_volscan2   mSETQ("0 0 * * fri,tue /usr/bin/afsvolscan > /dev/null 2>&1")
!cron.add_volscan3   mSET(MAILTO=root@inf.ed.ac.uk)
!cron.owner_volscan1 mSET(root)
!cron.owner_volscan2 mSET(root)
!cron.owner_volscan3 mSET(root)
```

The new server also needs to be added to the `get_results`, `afsclean` and `find_id` scripts.

If an AFS server is being decommissioned then remove the above.

If the new server has any partitions that are to be used as default partitions for new group space, AFSPartitions needs to be updated with the strings "DEFAULT" or "DEFAULT NO RO" added to those partitions.

A year after its deletion a space will be flagged up in the `afscheckup` report, and should be deleted from the database. Connect to the db with: `psql -h pgresearch spacedb spacedb_user`. The password can be found in any of the scripts, `afscreate` for example. To ensure you're deleting the correct group, delete using the UUID, which can be found with:

```
SELECT * from space_table WHERE space_name = 'group.testing' AND owner_uun = 'rarmstr1';
               space_id               | space_size | space_name | institute | owner_uun
--------------------------------------+------------+------------+-----------+-----------
 c52e7fe8-ad36-4dd1-93a3-259e750355f9 |    1000000 | group.testing  |           | rarmstr1
```

Once you're sure you've got the right group, delete it using the space_id:

```
DELETE FROM space_table WHERE space_id = 'c52e7fe8-ad36-4dd1-93a3-259e750355f9';
```

Future Considerations

It might be nice to create a cosigned page that shows a user what group space they have allocated to them. The link could be included in "account ending" email.

The database could supply data to PiP (the Personalised Information Portal), so that members of staff can easily the information we hold on any space they "own".

Recent changes to the way we create entities means that more of the sweb creation process could be automated as part of the `afscreate` script.

The find_id script finds all the ACLs that a specified ID features in. It might be useful to have a --delete flag, so that the ID is removed from the ACLs.

-- RossArmstrong - 15 Jul 2021

I	Attachment	Action	Size	Date	Who	Comment
🖼	spacedb_ER_diagram.png	manage	52.1 K	15 Jul 2021 - 11:05	RossArmstrong	The entity relationship diagram for spacedb.

Topic revision: r34 - 22 Sep 2021 - 22:05:33 - RossArmstrong