SSSD AD Provider:

Access Control

Pavel Reichl

February 2014

Contents of presentation

1. Need for access control
2. Simple Access Provider
3. LDAP Access Provider
4. Active Directory Access Provider
5. Group nesting example
6. Summary
7. Sources of further information

Need for access control (AC)

- Default configuration of the Active Directory provider enables only checking for an account expiration
- Admins need more power to specify AC, namely:
  - Define access for:
    - Users
    - Groups of users
  - Use a custom filtering mechanism:
    - Restrict permitted values of user's attributes (e.g. user's home directory)
  - Use a combination of multiple conditions

Existing solutions

Simple Access Provider

- **Granting** or **denying** access for objects is based on a content of “**allow” and “deny” lists:**
  - *simple_allow_users/simple_deny_users*
    - list of allowed/denied users
  - *simple_allow_groups/simple_deny_groups*
    - list of allowed/denied groups
  - Simple rules providing access control:
    - If any "allow" list is provided, all users are denied unless they appear in the list
    - If any "deny" list is provided, all users are granted access unless they appear in the list.
    - It is an error to define both simple_allow_users and simple_deny_users

Simple Access Provider: example

- *Setting environment:*

Membership	Cats	Pets
jerry	X	X
spike	X	member
tom	member	member

- *Example SSSD rules:*

```
access_provider = simple
simple_allow_users = jerry, tom
simple_allow_groups = pets
simple_deny_groups = cats
```

- *Results:*
  - *Granted access to: jerry and spike*
  - *Denied access to: tom (**denying rules take precedence over allowing rules**)*

Simple Access Provider: example cont.

- While logging it may be necessary to use Fully Qualified Names
  - e.g.

```
su jerry@ad.domain

Instead of simple

su jerry
```

- This depends on configuration option ***use_fully_qualified_names***
  - By default set to FALSE
  - If AD domain was joined using *realmd* it is explicitly set to TRUE

Simple Access Provider: pros & cons

- Pros:
  - Easy to configure
  - realmd support
- Cons:
  - Account expiration is not checked
  - Limited expressiveness: No way to combine several clauses
  - Does not align with the LDAP structure the Active Directory uses

LDAP Access Provider (deprecated)

- Good things:
  - Allows to base AC on a custom LDAP filter
  - Possible to combine several conditions
  - Conditions are not limited to user names or group membership

LDAP Access Provider (deprecated) cont.

- Bad things:
  - Nontrivial and clumsy configuration (beats the whole purpose of the AD provider)
  - The admin needs to combine AD and LDAP providers
    - Combining different providers can have strange side effects
  - An account expiration check must be configured separately, which is not obvious
  - No support for users from trusted AD domains
  - No realmd integration

New Access Provider

Active Directory Access Provider

- New access filter option to AD access provider
- Greatly **simplified configuration** when compared to the LDAP access control
- The very same expressiveness as *ldap_access_filter*
- More advanced format can be used to restrict the filter to a specific domain or a specific forest
- **Supersedes** LDAP Access Provider

Comparison of configuration

LDAP Access Provider

```
access_provider = ldap
ldap_access_order = filter, expire
ldap_account_expire_policy = ad
ldap_access_filter =
  (&(memberOf=cn=admins,ou=groups,dc=example,dc=com)(unixHomeDirectory=*))
ldap_sasl_mech = GSSAPI
ldap_sasl_authid = CLIENT_SHORTNAME$@EXAMPLE.COM
ldap_schema = ad
```

Active Directory Access Provider

```
access_provider = ad
ad_access_filter =
  (&(memberOf=cn=admins,ou=groups,dc=example,dc=com)(unixHomeDirectory=*))
```

Active Directory Access Provider: pros & cons

- Pros
  - Easy and intuitive configuration. Only one provider type is configured
  - Sane defaults - always checks for expiration
- Cons
  - No realmd integration

Group nesting: example

- Example environment:
  - User *bob* is member of *professors*
  - Group *research_staff* is member of group *staff*
  - Group *professors* is member of group *research_staff*
- Goal
  - Limit access only to members of *staff*
- Simple Access Provider (SAP):

```
access_provider = simple
simple_allow_groups = staff
```

  - Access will be GRANTED for user *bob* because SAP supports indirect membership

Group nesting: example cont.

- Active Directory Access Provider:

```
access_provider = ad
ad_access_filter =
  (memberOf=CN=staff,CN=Users,DC=ad-example,DC=test)
```

  - Access for Bob will be DENIED, because indirect membership is not supported by AD Access Provider
  - To grant access for indirect members the filter must be expanded for all indirect groups

```
 access_provider = ad
 ad_access_filter =
(|(|(memberOf=CN=research_staff,CN=Users,DC=ad-example,DC=test)
    (memberOf=CN=staff,CN=Users,DC=ad-example,DC=test))
  (memberOf=CN=professors,CN=Users,DC=ad-example,DC=test))
```

SSSD AD Provider Access Control: Summary

	Simple Access Provider	LDAP Access Provider	AD Access Provider
Configuration difficulty	Easy	Hard	Medium
Nested group membership	Supported	Not supported	Not supported
Expressiveness	Limited to allowed/denied lists of users and groups	Complex queries	Complex queries
When to use	When allow/deny lists are sufficient	If AD Access Provider is not available	If Simple Access Provider is not enough

Sources for further reading

- Design Docs: Active Directory Access Control
  - https://fedorahosted.org/sssd/wiki/DesignDocs/ActiveDirectoryAccessControl
- How to: Configure SSSD with AD server
  - https://fedorahosted.org/sssd/wiki/Configuring_sssd_with_ad_server
- Manual pages:
  - sssd-ad
  - sssd-ldap
  - sssd-simple


freeIPA
identity | policy | audit