Contents of the presentation

1. Overview of Active Directory related improvements
2. Range attributes support
3. Mapping Active Directory SIDs onto UNIX IDs
4. The Active Directory provider
5. Hands-on example – joining a SSSD 1.9 client to an Active Directory domain
Active Directory client improvements in SSSD 1.9

• Support of range attributes
  • Enables resolving large groups from AD
• Mapping of Windows SIDs to UNIX IDs
  • Removes the requirement for AD users to contain POSIX attributes, in particular UIDs and GIDs
• A new Active Directory provider
  • Simplifies SSSD config file and uses AD specific defaults
  • Takes advantage of several AD specific features to improve performance
Range attribute parsing

• By default, AD limits the number of multivalued attributes returned in a single search
  • Typically an issue with the member attribute when large groups are present in AD
• If the number of values is over the “single page” limit, the attributes are returned in the form “attribute;range=low-high”
  • Example: member;range=99-499
• The SSSD 1.9 is able to parse and process these attributes
  • Support on by default in the LDAP provider, no configuration needed
Mapping AD SIDs to UNIX IDs

- Windows use Security Identifiers to identify users and groups
  - Contains identifier of the domain and relative identifier of the object
- In SSSD 1.9, the sssd is able to automatically map these SIDs to IDs
- The SSSD automatically selects the proper range for mapping SIDs to IDS preventing overlaps and conflicts between different domains
- In LDAP provider, set `ldap_id_mapping = true`
  - Off by default in LDAP provider, on by default in AD provider
The Active Directory provider

- It was possible for client to use identities from an Active Directory server prior to SSSD 1.9
- The SSSD would treat Active Directory as a generic LDAP server for identities and Kerberos server for authentication
- So why bother with a brand new AD provider?
  - POSIX attributes were required on the AD side
  - Non trivial configuration of the SSSD
  - Did not use AD-specific features the client could benefit from, such as tokenGroups
Benefits over using the LDAP provider

- Simplified configuration
  - The AD provider already contains the correct defaults for attribute names as used on the AD side

- Secure by default
  - The AD provider defaults to using GSSAPI for encrypting identity lookups

- Faster logins
  - Using the tokenGroups attribute speeds up the initgroups operation

- Support for ID mapping
  - The Windows Security Identifiers (SIDs) are automatically converted into UNIX IDs
AD provider configuration example

- **sssd.conf with LDAP provider**
  ```
  [domain/ad.example.com]
  id_provider = ldap
  auth_provider = krb5
  ldap_schema = rfc2307bis
  ldap_sasl_mech = GSSAPI
  ldap_user_object_class = user
  ldap_group_object_class = group
  ldap_user_home_directory = unixHomeDirectory
  ldap_user_principal = userPrincipalName
  ldap_force_upper_case = true
  ```

- **SSSD with AD provider**
  ```
  [domain/ad.example.com]
  id_provider = ad
  #uncomment if autodiscovery is not required
  #ad_server = ad.example.com
  ```

- ..and more that wouldn't fit on the slide..
Performance enhancements in AD provider

- Many users were not happy with slow logins
- Usually the slowest part of login is initgroups
  - The initgroups operation is performed on each login
  - Collects the groups a user is a member of
  - Benchmark: `id -G $username`
- The LDAP provider would look up all the groups the user is a member of with LDAP searches
  - Could be several searches per single login, at least one search per nesting level
- The AD provider uses tokenGroups to improve performance
tokenGroups

- With AD provider it is possible to grab the list of groups the user is a member of along with the user entry
- The AD specific tokenGroups attribute contains a list of all the Security Identifiers (SIDs) the user is a member of
- The client must be able to map the SIDs to UNIX IDs
  - Only works when ID mapping is enabled
Comparison with Winbind and the LDAP provider

<table>
<thead>
<tr>
<th>Feature</th>
<th>SSSD with LDAP/KB5 providers</th>
<th>SSSD with AD provider</th>
<th>Winbind</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requires SFU/IMU</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Supports ID mapping</td>
<td>None</td>
<td>One method</td>
<td>Multiple methods</td>
</tr>
<tr>
<td>AD specific call to retrieve initgroups</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Handles mounting CIFS shares</td>
<td>No</td>
<td>No (planned for 1.10)</td>
<td>Yes</td>
</tr>
<tr>
<td>DNS site support</td>
<td>No</td>
<td>No (planned for 1.10)</td>
<td>Yes</td>
</tr>
<tr>
<td>DNS dynamic updates</td>
<td>No</td>
<td>No (planned for 1.10)</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Q&A: Migration from the LDAP provider

- A client already uses SSSD with AD using the LDAP provider. Can I simply switch to using the AD provider?
  - Not simply. The UIDs and GIDs of users and groups would change when the client switches from using the POSIX attributes to using ID mapping.
- Can I disable ID mapping and just use the faster initgroups feature?
  - No, the tokenGroups support only works in conjunction with ID mapping.
Q&A: Migration from Winbind

• A client already uses Winbind for his setup. Can I simply switch to using SSSD with ID mapping?
  • Not easily. You would have to carefully set the default domain SID (ldap_idmap_default_domain_sid) and the range start (ldap_idmap_range_min). But in general, such migration is not recommended.
Joining a Linux client to Active Directory

- Example – enroll a client linux.example.com into an AD server at ad.example.com that is administering the domain EXAMPLE.COM

- Pre-requisites
  - Functional host name resolution
    - Including SRV records if auto discovery is needed
    - Our examples will even use the AD server as DNS server
  - Synchronized time for Kerberos
  - Packages installed
    - yum install sssd krb5-workstation samba-common authconfig
The general steps to join a Linux client to AD

- Enroll the client into Active Directory
  - Configure krb5.conf
  - Configure smb.conf
  - Obtain the keytab using the net utility
- Configure the system to use SSSD for looking up identity information and performing authentication
- Configure the SSSD
Enrolling a client – configure Kerberos

- Start by configuring /etc/krb5.conf
- Define the [realms] and [domain_realm] sections if autodiscovery doesn't work

```
[logging]
    default = FILE:/var/log/krb5libs.log

[libdefaults]
    default_realm = EXAMPLE.COM
dns_lookup_realm = true
dns_lookup_kdc = true
ticket_lifetime = 24h
renew_lifetime = 7d
rdns = false
forwardable = yes
```
Enrolling a client – configure Samba

- Edit /etc/samba/smb.conf

```
[global]
  workgroup = EXAMPLE
  client signing = yes
  client use spnego = yes
  kerberos method = secrets and keytab
  log file = /var/log/samba/%m.log
  password server = AD.EXAMPLE.COM
  realm = EXAMPLE.COM
  security = ads
```
Enrolling a client – joining the domain

- Obtain the Kerberos ticket of a user to enroll as
  - `kinit Administrator`
  - Can be any user with sufficient rights to join a machine to domain

- Join the machine
  - `net ads join -k`
  - Should print “Joined 'linux' to dns domain 'example.com'” on success
  - A new file `/etc/krb5.keytab` should be created
Enrolling a client – check the keytab

- Check if the keytab contains the expected principal
  - `klist -k`
  - Should print several entries that contain both the full and the short host name of the client and the domain

- Try to `kinit` using the keytab
  - `kinit -k`
Configure the system to use the SSSD

- Currently authconfig can't configure the SSSD with the AD provider on its own
- We'll use authconfig to set up the system to use the SSSD
  - authconfig --enablesssdauth --enablesssd --update
  - nsswitch.conf to look up identity information with the SSSD
  - PAM stack to perform authentication using the SSSD
- ..and then configure the SSSD manually
Configure the SSSD

- Configure /etc/sssd.conf
- Use ad_server to specify the AD server if autodiscovery is not working
- Example sssd.conf using autodiscovery:

```ini
[sssd]
services = nss, pam
config_file_version = 2
domains = EXAMPLE.COM

[domain/EXAMPLE.COM]
id_provider = ad
```
Enrolling a client – test the setup

- Start the SSSD service
  - `service sssd start`
- Test if identity information can be obtained
  - `getent passwd aduser`
- Test if authentication works
  - Some services (notably sshd) must be restarted to re-read the new PAM config
  - `ssh aduser@linux.example.com`
Troubleshooting the SSSD

• Generic checklist
  • Check if time is synchronized
  • Check if the keytab `/etc/krb5.keytab` contains

• What if identity information can't be obtained
  • Raise the `debug_level` in the `[nss]` and `[domain]` sections of `sssd`, restart the SSSD and attach the log files in `/var/log/sssd`

• What if logins do not work
  • All of the above and debug logs from the `[pam]` section
  • `/var/log/secure`