NAME

sssd-sudo − Configuring sudo with the SSSD back end

DESCRIPTION

This manual page describes how to configure **sudo**(8) to work with **sssd**(8) and how SSSD caches sudo rules.

CONFIGURING SUDO TO COOPERATE WITH SSSD

To enable SSSD as a source for sudo rules, add *sss* to the *sudoers* entry in **nsswitch.conf**(5).

For example, to configure sudo to first lookup rules in the standard **sudoers**(5) file (which should contain rules that apply to local users) and then in SSSD, the nsswitch.conf file should contain the following line:

```
sudoers: files sss
```

More information about configuring the sudoers search order from the nsswitch.conf file as well as information about the LDAP schema that is used to store sudo rules in the directory can be found in **sudoers.ldap**(5).

Note: in order to use netgroups or IPA hostgroups in sudo rules, you also need to correctly set **nisdomainname**(1) to your NIS domain name (which equals to IPA domain name when using hostgroups).

CONFIGURING SSSD TO FETCH SUDO RULES

All configuration that is needed on SSSD side is to extend the list of *services* with "sudo" in [sssd] section of **sssd.conf**(5). To speed up the LDAP lookups, you can also set search base for sudo rules using *ldap_sudo_search_base* option.

The following example shows how to configure SSSD to download sudo rules from an LDAP server.

```
[sssd]
config_file_version = 2
services = nss, pam, sudo
domains = EXAMPLE

[domain/EXAMPLE]
id_provider = ldap
sudo_provider = ldap
ldap_uri = ldap://example.com
ldap_sudo_search_base = ou=sudoers,dc=example,dc=com
```

When the SSSD is configured to use IPA as the ID provider, the sudo provider is automatically enabled. The sudo search base is configured to use the compat tree (ou=sudoers,$DC).

THE SUDO RULE CACHING MECHANISM

The biggest challenge, when developing sudo support in SSSD, was to ensure that running sudo with SSSD as the data source provides the same user experience and is as fast as sudo but keeps providing the most current set of rules as possible. To satisfy these requirements, SSSD uses three kinds of updates. They are referred to as full refresh, smart refresh and rules refresh.

The *smart refresh* periodically downloads rules that are new or were modified after the last update. Its primary goal is to keep the database growing by fetching only small increments that do not generate large amounts of network traffic.

The *full refresh* simply deletes all sudo rules stored in the cache and replaces them with all rules that are stored on the server. This is used to keep the cache consistent by removing every rule which was deleted from the server. However, full refresh may produce a lot of traffic and thus it should be run only occasionally depending on the size and stability of the sudo rules.

The *rules refresh* ensures that we do not grant the user more permission than defined. It is triggered each time the user runs sudo. Rules refresh will find all rules that apply to this user, check their expiration time and redownload them if expired. In the case that any of these rules are missing on the server, the SSSD will do an out of band full refresh because more rules (that apply to other users) may have been deleted.

If enabled, SSSD will store only rules that can be applied to this machine. This means rules that contain one of the following values in *sudoHost* attribute:

- keyword ALL
- wildcard
- netgroup (in the form "+netgroup")
- hostname or fully qualified domain name of this machine
- one of the IP addresses of this machine
- one of the IP addresses of the network (in the form "address/mask")

There are many configuration options that can be used to adjust the behavior. Please refer to "ldap_sudo_*" in **sssd-ldap**(5) and "sudo_*" in **sssd.conf**(5).

SEE ALSO

sssd(8), **sssd.conf**(5), **sssd-ldap**(5), **sssd-krb5**(5), **sssd-simple**(5), **sssd-ipa**(5), **sssd-ad**(5), **sssd-sudo**(5), **sss_cache**(8), **sss_debuglevel**(8), **sss_groupadd**(8), **sss_groupdel**(8), **sss_groupshow**(8), **sss_groupmod**(8), **sss_useradd**(8), **sss_userdel**(8), **sss_usermod**(8), **sss_obfuscate**(8), **sss_seed**(8), **sssd_krb5_locator_plugin**(8), **sss_ssh_authorizedkeys**(8), **sss_ssh_knownhostsproxy**(8), **sssd-ifp**(5), **pam_sss**(8). **sss_rpcidmapd**(5)

AUTHORS

The SSSD upstream − http://fedorahosted.org/sssd