External Authentication Guide

Technical Note

Table of Contents

1 Introduction

This guide demonstrates the configuration of SRE for authenticating GUI users through external authentication servers. Support is provided for LDAP and OpenID protocols.

2 Authentication vs. Authorization

Authentication is solely managed by the external identity provider, focusing on verifying user credentials. However, it is crucial to note that user accounts must be locally created for authorization purposes. It is imperative to configure users within the SRE framework prior to attempting authentication with external servers, ensuring a seamless and secure integration of authentication and authorization processes.

> **Note**
>
> The only exception is by using ldap_role_mapping setting described below. If used, it's not necessary to create user on SRE, only roles are needed.

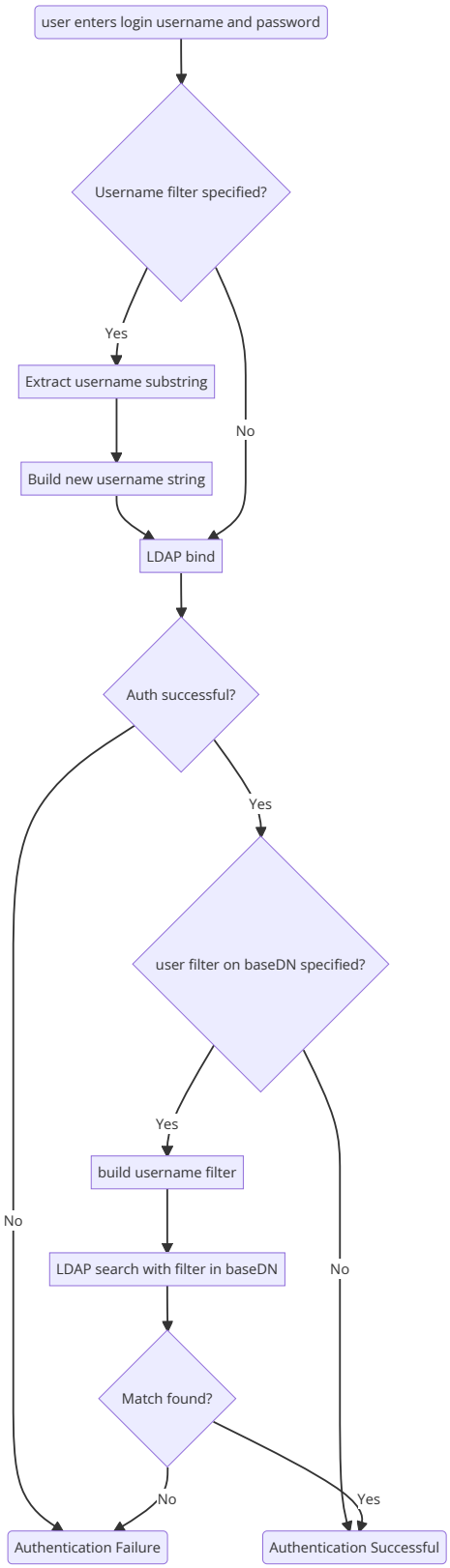
3 LDAP

LDAP settings are configured in */opt/sre/etc/sre.cfg* configuration file on element managers in the **[authentication]** section. When specified, LDAP authentication overrides the built-in authentication mechanisms.

Below are the supported options:

Parameter	Value	Mandatory	Default value if not specified	Description
protocol	local, ldap	**N**	local	
ldap_url	<ldap|ldaps>://<hostname or ip>[:port]	**Y** (if protocol is ldap)	N/A	URL of LDAP server used for authentication
ldap_username_regex	Regex with at least one capture group	**N**	Original username as entered in login page	used for extracting a substring from the original username
ldap_username_substitution	Substitution containing replacement expression (\<number>)	**Y** (if ldap_username_match is specified else **N**)	No substitution	Used to construct a new username string with substring extracted with ldap_username_match regular expression
ldap_username_filter_base_dn	LDAP distinguished name	**N**		Base DN used for LDAP search
ldap_username_filter_regex	Regex with at least one capture group	**Y** (if ldap_username_filter_base_dn is specified else **N**)		Used for extracting a substring from the original username
ldap_username_filter_substitution	String containing replacement expression (\<number>)	**Y** (if ldap_username_filter_base_dn is specified else **N**)		Used to construct a new username string with substring extracted with ldap_username_match regular expression
ldap_role_mapping	<LdapGroup DN>:<SRE role> for each line	**N**	No ldap group mapping, users must exist on SRE	Mapping of ldap groups to sre roles

This is a state diagram for the LDAP authentication procedure:


user enters login username and password
Username filter specified?
Yes
No
Extract username substring
Build new username string
LDAP bind
Auth successful?
Yes
No
user filter on baseDN specified?
Yes
No
build username filter
LDAP search with filter in baseDN
Match found?
No
Yes
Authentication Failure
Authentication Successful

Example configuration section:

```
[authentication]
protocol=ldap
ldap_url=ldaps://<ldap server>
ldap_username_match=(.*)@example.com
ldap_username_substitution=uid=\1,ou=People,dc=example,dc=com
ldap_username_filter_base_dn=ou=People,dc=example,dc=com
ldap_username_filter_match=(.*)
ldap_username_filter_substitution=(&(uid=\1)(memberOf=cn=Acme-Admin,ou=Groups,
    ↪ dc=example,dc=com))
ldap_role_mapping=cn=sre_expert,cn=groups,dc=bxl,dc=netaxis,dc=be:admin
```

In this example, for a user trying to login with *firstname.lastname@example.com*, the system initiates an LDAP connection utilizing the username *uid=firstname.lastname@,ou=People,dc=example,dc=com*, along with the provided password. Following a successful login, a filter operation ensues with the filter *(&(uid=firstname.lastname@example.com)(memberOf=cn=Acme-Admin,ou=Groups,dc=example,dc=com))* starting from the base DN *ou=People,dc=example,dc=com*. The flexibility inherent in the regular expression parameters facilitates diverse manipulations of the username during both authentication and user filtering operations.

After modifying the sre.cfg configuration, you can test authentication using the following command:

```
/opt/sre/bin/sre-admin users test-authentication
Please enter your username: testuser
Please enter your password: *******
```

The output will display *Successful* or *Failed* depending on the result of the authentication process.

4 OpenID

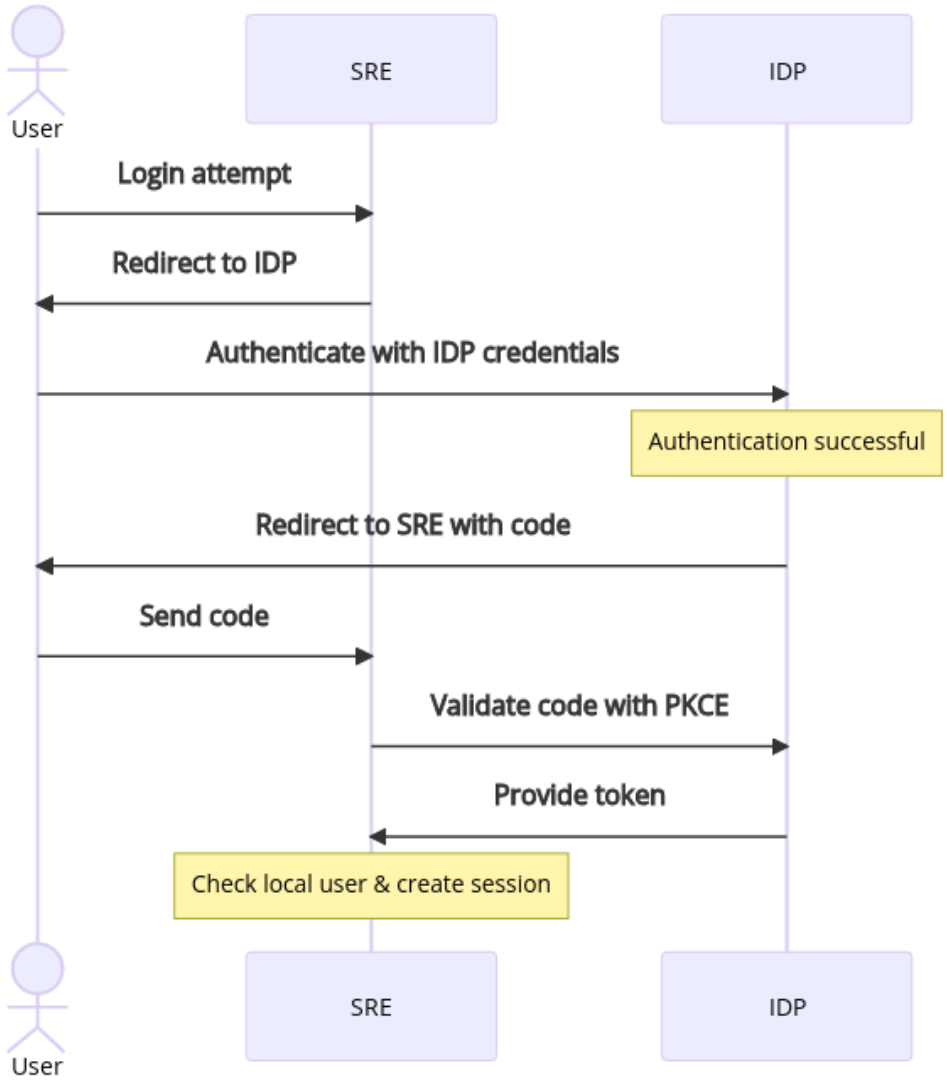
SRE facilitates external authentication via an OpenID identity provider by incorporating the Authorization Code Flow with Proof Key for Code Exchange (PKCE). It does not support the standard Authorization Code Flow with a client secret.

OpenID settings are configured in */opt/sre/etc/sre.cfg* configuration file on element managers in the *[authentication_oidc]* section.

Below are the supported options:

Parameter	Value	Mandatory	Description
wellknown_endpoint	https URL	**Y**	Well-known endpoint refers to a standardized discovery mechanism that allows SRE to obtain information about the OpenID Provider
issuer	https URL	**Y**	A URL that identifies the OpenID Provider, the entity responsible for authenticating the user and providing identity information
audience	string	**Y**	Unique identifier assigned to a client application when it registers with an OpenID Connect provider (client_id)
tenant	string	**Y**	If a user information claim with the key `sre_user_<tenant>` is present in the returned token, that claim must include the email address of the user.

The diagram below illustrates the OpenID flow SRE supports:



Sample configuration section:

```
[authentication_oidc]
wellknown_endpoint=https://id.example.com/auth/realms/sre/.well-known/openid-
    ↪ configuration
issuer=https://id.example.com/auth/realms/sre
audience=GUI
tenant=somecustomer
```