didmos2 Authenticator

1. Overview

didmos2 Authenticator is the central authentication component in the didmos2 software suite. It is based on the SATOSA proxy and supports the SAML and OpenID Connect protocols.

SATOSA is composed of different modules. **Backend modules** represent authentication methods and connect to different authentication sources. The result of a backend module consists of an user identifier and possibly additional user attributes from the authentication source. Conversely, **frontend modules** are used to connect to various services. They convey the information (which is based on whatever the backend module produced and potential modifications in micro services) back to the relying parties. Finally, **micro services** perform all kinds of tasks (like routing or attribute modifications) between frontend and backend modules. They can be further divided into **request micro services** (these run when routing from the frontend to the backend takes place, before any response from the backend is produced) and **response micro services** (these run on the way back from the backend to the frontend, after the backend has produced its result).

The following SATOSA modules are currently used in didmos2 Authenticator:

Backend Modules	local	Login with local account at internal ldap
	facebook	Social Login with Facebook
	google	Social Login with Google
	linkedin	Social Login with LinkedIn
	github	Social Login with Github
	Saml2	Login with SAML2 IDP or federation
	Saml2UCS	Login with UCS SAML IDP
	externalldap	Login with external LDAP or AD
Response Micro Services	LdapAttributeStore	Query additional user attributes at internal ldap
	Privacyidea	MFA with privacyIdea for internal users
	ldap (ldap_account_registration)	Shadowaccount registration for all but internal users
Request Micro Services	DiscoveryRouter	Frontend service to choose login method
Frontend Modules	OIDC	OIDC Provider
	SAML2	SAMl IDP

2. Deployment

The following environment variables must be set, in order to start the didmos2 Authenticator service as part of the didmos2 software suite with default configuration:

```
SATOSA_MONGODB_PASSWORD: <password as used for the account with the name satosa in didmos2-mongodb>

SATOSA_SSO_ENCRYPTION_KEY: <random value>

SATOSA_STATE_ENCRYPTION_KEY: <random value>

SATOSA_BASE_HOST: <external hostname for the didmos2 Authenticator deployment, e.g. auth.didmos2.de>
```

All other settings are initialized with default values and can be adjusted as described below.

If you intend to use didmos2 Authenticator with the rest of the didmos2 software suite, consider also the following settings:

DIDMOS2_CLIENT_PROVISION: <redirect uri for the didmos2 lui frontend, e.g. https://didmos2.de>

This automatically registers the didmos2 lui client on startup.

For deployment instruction in a setup with other didmos2 software components, please see the didmos2-demo-compose project.

For a stand-alone deployment (without external loadbalancer) the following docker-compose.yml file can be used as a starting point:

```
version: "3.1"

services:
  auth:
      image: registry.gitlab.daasi.int/didmos2/didmos2-auth:v2.4.0
      depends_on:
        - mongo
      environment:
        SATOSA_MONGODB_PASSWORD: satosa
        SATOSA_SSO_ENCRYPTION_KEY: secret
        SATOSA_STATE_ENCRYPTION_KEY: secret
        SATOSA_BASE_HOST: auth.daasi.devel
        SATOSA_ENABLE_SSL: "Yes"
      ports:
        - 443:443/tcp
      networks:
      - didmos2
  mongo:
    image: registry.gitlab.daasi.int/didmos2/didmos2-mongodb:v2.2.0
    networks:
      - didmos2
    volumes:
      - didmos2-mongo-db:/data/db
  ldap:
    image: registry.gitlab.daasi.int/didmos2/didmos2-openldap:v2.3.0
    environment:
      ACCESSLOG_PW: CHANGEME
      MANAGER_PW: CHANGEME
    ports:
      - 389:389/tcp
    networks:
      - didmos2
    volumes:
      - didmos2-openldap-db:/var/lib/ldap:rw

volumes:
  didmos2-mongo-db: {}
  didmos2-openldap-db: {}

networks:
  didmos2:
```

3. CLI

A command line interface (CLI) is provided for some configuration tasks, which can be accessed like so:

```
docker exec {container-name} didmos2-auth-cli

Example: docker exec didmos2-demo-auth didmos2-auth-cli
```

3.1. Use CLI to configure OIDC clients

The following commands are available for configuration of OIDC clients:

- clients list
- clients show
- clients rm
- clients add

Use the --help flag for a description of the parameters for each command.

3.1.1. Example: Add a new client

```
docker exec -it didmos2-demo-auth didmos2-auth-cli clients add --flow code --client-name Foobar --redirect-uri
https://foo.bar

--client-secret not provided, but required for --flow = code or both. Generated random value for client_secret:
dRWl2zin2VD7AyOOAJghweNF
Generated random value for client_id: 9SHRfIcW034AQY8s
Client successfully created: {'_id': ObjectId('5d5d4eff791d34483ec0bbee'), 'data': {'application_type': 'web',
'client_secret': 'dRWl2zin2VD7AyOOAJghweNF', 'redirect_uris': ['https://foo.bar'], 'client_id':
'9SHRfIcW034AQY8s', 'response_types': ['code'], 'client_name': 'Foobar'}, 'lookup_key': '9SHRfIcW034AQY8s'}
```

lookup_key is always the client_id of a client and must be unique. If --client-id is empty, a random id is generated. If --client-secret is empty and --flow is either code or both, a random value is generated.

4. Configuration

4.1. Full environment variable reference

Base configuration	Default	Description	Required in docker-compose.yml
SATOSA_BASE_HOST		e.g. auth.didmos2.de	*
SATOSA_STATE_ENCRYPTION_KEY		Random value used for enc of state cookie	*
SATOSA_SSO_ENCRYPTION_KEY		Random value used for enc of sso cookies	*
SATOSA_DISOVERY_ADDITIONAL_PARAMS		If set, additional config parameters are used for the discovery module. Currently the only purpose is settings this to "bypass_target: Saml2UCS" to bypass discovery and directly enter the specified authentication method	
SATOSA_ENABLE_SSL	No	Allow running without a dedicated proxy by offering TLS support. This must be combined with exposing port 443.	
Internal LDAP authentication	**Default**		
SATOSA_LDAP_ACTIVE	Yes	Activate local didmos2 login	
SATOSA_REGISTRATION_URL		e.g. https://didmos2.de/selfreg	*
MongoDB connection			
SATOSA_MONGODB_USERNAME	satosa	Username for mongodb service	
SATOSA_MONGODB_HOST	mongo	Host for mongodb service	
SATOSA_MONGODB_PORT	27017	Port for mongodb service	
SATOSA_MONGODB_DATABASE	satosa	Database name	
SATOSA_MONGODB_PASSWORD		Password for mongodb service	*

OIDC Frontend			
SATOSA_OIDC_DYNAMIC_REGISTRATION	No	Allow dynamic registration of oidc clients	
Internal LDAP Credentials			
SATOSA_INTERNALLDAP_URL	ldap://ldap:389	Internal LDAP Host	
SATOSA_INTERNALLDAP_BIND_DN	uid=satosa,ou=accounts,ou=DSA,dc=didmos,dc=de	Bind DN for internal LDAP	
SATOSA_INTERNALLDAP_BIND_PASSWORD	PdefaultWsatosaD	Bind Credential for internal LDAP	
SATOSA_INTERNALLDAP_SEARCH_BASE	ou=data,ou=default-tenant,dc=didmos,dc=de	Search base for users in internal LDAP	
SATOSA_INTERNALLDAP_CREATE_BASE	ou=social-people,ou=data,ou=default-tenant,dc=didmos,dc=de	Base DN for creation of shadow accounts	
PrivacyIdea MFA			
SATOSA_MFA_PRIVACYIDEA_ACTIVE	No	Activate privacyIdea MFA for local accounts	
SATOSA_MFA_PI_URL		privacyIdea URL	
SATOSA_MFA_PI_USERNAME		privacyIdea admin user	
SATOSA_MFA_PI_PASSWORD		privacyIdea admin password	
SATOSA_MFA_PI_CHALLENGE_TOKENTYPES		Tokentypes, which require challenge & response, e.g. "email, sms"	
Facebook Social Login			
SATOSA_FACEBOOK_ACTIVE	No	Activate Facebook login	
SATOSA_FACEBOOK_CLIENT_ID			
SATOSA_FACEBOOK_CLIENT_SECRET			
Google Social Login			
SATOSA_GOOGLE_ACTIVE	No	Activate Google login	
SATOSA_GOOGLE_CLIENT_ID			
SATOSA_GOOGLE_CLIENT_SECRET			
LinkedIn Social Login			
SATOSA_LINKEDIN_ACTIVE	No	Activate Linkedin login	
SATOSA_LINKEDIN_CLIENT_ID			
SATOSA_LINKEDIN_CLIENT_SECRET			
Github Social Login			
SATOSA_GITHUB_ACTIVE	No	Activate Github login	
SATOSA_GITHUB_CLIENT_ID			
SATOSA_GITHUB_CLIENT_SECRET			
SAML2 Login			
SATOSA_SAML2_ACTIVE	No	Activate Saml2 login	

SATOSA_SAML2_METADATA		URL to Saml2 metadata	
SATOSA_SAML2_WAYF_ACTIVE		Use WAYF yes/no	
SATOSA_SAML2_WAYF_URL		URL to WAYF	
SATOSA_SAML2_METADATA_SIGNED		Set to "No"	
Externalldap Login			
SATOSA_EXTERNALLDAP_ACTIVE	No		
SATOSA_EXTERNALLDAP_LDAPURL			
SATOSA_EXTERNALLDAP_BINDDN			
SATOSA_EXTERNALLDAP_BINDPWD			
SATOSA_EXTERNALLDAP_SEARCHBASE			
SATOSA_EXTERNALLDAP_SEARCHATTRIBUTE			
SATOSA_EXTERNALLDAP_IDATTRIBUTE			
UCS Login			
SATOSA_UCS_ACTIVE	No		
didmos2 specific settings			
DIDMOS2_CLIENT_PROVISION		If set, automatic provisioning of a didmos2 lui client with the value of its redirect_uri is triggered during startup. E.g.: DIDMOS2_CLIENT_PROVISION: https://didmos2-demo.daasi.de	

5. Advanced configuration

5.1. General strategy to override configuration files

Inside the container most SATOSA configuration is generated automatically when starting the container.

For this, various config files in /etc/satosa exist as template files with the .DOCKERCONFIG file format. If you want to override or extend these files, one strategy is to copy these .DOCKERCONFIG files to the host system, modify them (while keeping the placeholders intact) and then mount the modified versions into the docker container. This way the general deployment strategy continues to work, while allowing for various modifications.

5.1.1. Example

Consider the configuration file for attribute mapping (see chapter "Attributes") and copy the content from a running container to the host system:

docker cp {name_of_container}:/etc/satosa/conf/internal_attributes.yaml.DOCKERCONFIG .

In various places placeholders are used, which should not be touched (e.g. ###SATOSA_SAML2_EXTERNAL_IDENTIFIERS###).

However, removing the "fe_name" block would prevent the system from returning the name attribute to any services. Likewise additional attributes could be added here (however, consider, that these attribute must generally be available in the internal LDAP server and must also be added to /etc/satosa/plugins/ldap_attribute_store.yaml).

The modified file could then be mounted as a file volume to override the default .DOCKERCONFIG file.

5.2. UCS Mode (only applicable when running didmos Auth in UCS)

A mode to run didmos2 Authenticator in UCS is available. This mode can be used to activate the Saml2_UCS backend and connect to the UCS SAML IDP:

```
SATOSA_UCS_ACTIVE: "Yes"
```

Note that this setting should be combined with deactivating all other login methods and especially the internal LDAP login:

```
SATOSA_LDAP_ACTIVE: "No"
```

To complete setup, the following volumes must be added to the satosa service in docker-compose.yml:

```
volumes:
  - $PWD/satosa-credentials:/etc/satosa/credentials
  - $PWD/ucs-metadata.xml:/etc/satosa/ucs-metadata.xml
```

In satosa-credentials/ the files saml2_ucs_backend.crt and saml2_ucs_backend.key must be present.

The metadata of the SAML IDP in UCS must be present as ucs-metadata.xml.

6. Attributes

6.1. OIDC Frontend

The following scopes and claims are available:

Scope	Claim	Description
openid	sub	Public, unique, opaque identifier (didmosUUID)
email	email	Email of the user (if available)
profile	name	Display name of the user (if available)