Django SPID plugin Documentation

Release 0.1

PDP Free Software User Group <info@pdp.linux.it>

Oct 08, 2017

Contents:

CHAPTER 1

Disclaimer

Siamo a pochi minuti dalla PR:

Il progetto django-spid, è ovviamente in versione 0.1, si riassumono per praticità in questo spazio i risultati raggiunti:

- Documentazione del plugin con tema integrato Italia *sphinx_italia_theme* con hook da Github -> ReadTheDocs: http://spid-django-plugin-documentation.readthedocs.io
- La Django app django-spid (nel repo spid-django) contenitore per implementazioni specifiche di SAML SP per SPID https://github.com/pdpfsug/spid-django/
- Il fork della app django *django-saml2-auth* per consentire il supporto di IDP multipli https://github.com/pdpfsug/django-saml2-auth
- Un mockup di una API REST per il recupero delle informazioni degli IDP (metadata, logo, cert, attributes_map) https://idpapi.azurewebsites.net/api/IdentityProviders ospitata su VM MS Azure
- Un servizio (implementato in django-spid, runserver di Django) attivo su hackamerino.labs.befair.it su VM MS Azure comprende: * integrazione bottone ufficiale SPID http://hackamerino.labs.befair.it/ * prova URL di autenticazione presso un IDP http://hackamerino.labs.befair.it/accounts/login/?idp_metadata=https://idp.spid.gov.it:9443/samlsso * un modello per il salvataggio delle info degli IdP (ma se si va avanti con la API REST dedicata mi sembra meglio)
- Un progetto su RH OpenShift http://spid-django-pdp-spid-django.apps.justcodeon.it/saml2_auth/ abbozzato per il continuous delivery

insomma, il codice è qui, le configurazioni di GitHub ovviamente rimangono sul nostro repo...

CHAPTER 2

Introduzione

About

Testo di prova carino.

Progettazione

Testo di prova bellino.

CHAPTER 3

Installazione

1. L'installazione consiste nel clonare inizialmente la seguente repo.
2. Successivamente si procederà con la creazione di un ambiente virtuale, mediante *virtual-env*, nella quale l'utente installerà le dipendenze richieste dall'applicativo software attraverso il comando:

   ```
   python pip install -r requirements.txt
   ```

3. Fatto ciò, occorrerà inizializzare il database affinché l'intero sistema cooperi al raggiungimento dello scopo, tramite il seguente comando:

   ```
   ./manage.py migrate
   ```

4. In seguito sarà necessario avviare il server:

   ```
   ./manage.py runserver
   ```

5. A questo punto, basterà visualizzare sul proprio browser il localhost puntandolo sulla porta **8000**, valore di default, visualizzando così l'interfaccia grafica prodotta dal software.
6. Infine sarà possibile tenere sotto controllo il log degli eventi nella bash da cui è stato lanciato il comando *runserver* per poter leggere eventuali errori.

CHAPTER 4

Django SPID Plugin

Il plugin SPID per Django si basa sull'applicazione Django django-saml2-auth

Di tale applicazione è stato fatto un fork su https://github.com/pdpfsug/django-saml2-auth per consentire il supporto di multipli IdentityProvider.

Poi è stata creata l'app django-spid che include le fixtures con i valori degli IdP SPID italiani supportati dal bottone SPID SP access button

Cosa è necessario fare

Attestare il proprio SP sugli IdP di cui si vuole supportare l'autenticazione

Per ulteriori opzioni di configurazione vedere il repository github della app.

CHAPTER 5

Appendice: SAML

Il protocollo SAML in breve

SAML è il mezzo (o più specificatamente il framework/protocollo) attraverso cui SPID realizza la parte di autenticazione.

Le entità (**entity**) coinvolte nel protocollo SAML sono fondamentalmente 2:

1. il **Service Provider** (*SP*)che è il servizio web di cui l'utente vuole usufruire
2. l' **Identity Provider** (*IdP*) che è il server di autenticazione SSO e verifica le credenziali dell'utente

Esempi di IdP sono Poste Italiane, Aruba, etc., quindi probabilmente non lo si dovrà reimplementare.

Più in dettaglio le fasi del processo di autenticazione sono mostrate come da schema seguente

L'entityID

L'unico campo richiesto obbligatoriamente nella configurazione di un **Service Provider** è l'entity ID. L'entity Id di un Service Provider deve essere univoco all'interno dell'Identity Provider cui si registra, per questo di solito si usa il FQDN.

L'AssertionConsumerService

Almeno un elemento di questo tipo deve essere definito nei metadati di un **Service Provider**.

<AssertionConsumerService> [One or More] One or more elements that describe indexed endpoints that support the profiles of the Authentication Request protocol defined in [SAMLProf]. All service providers support at least one such endpoint, by definition.

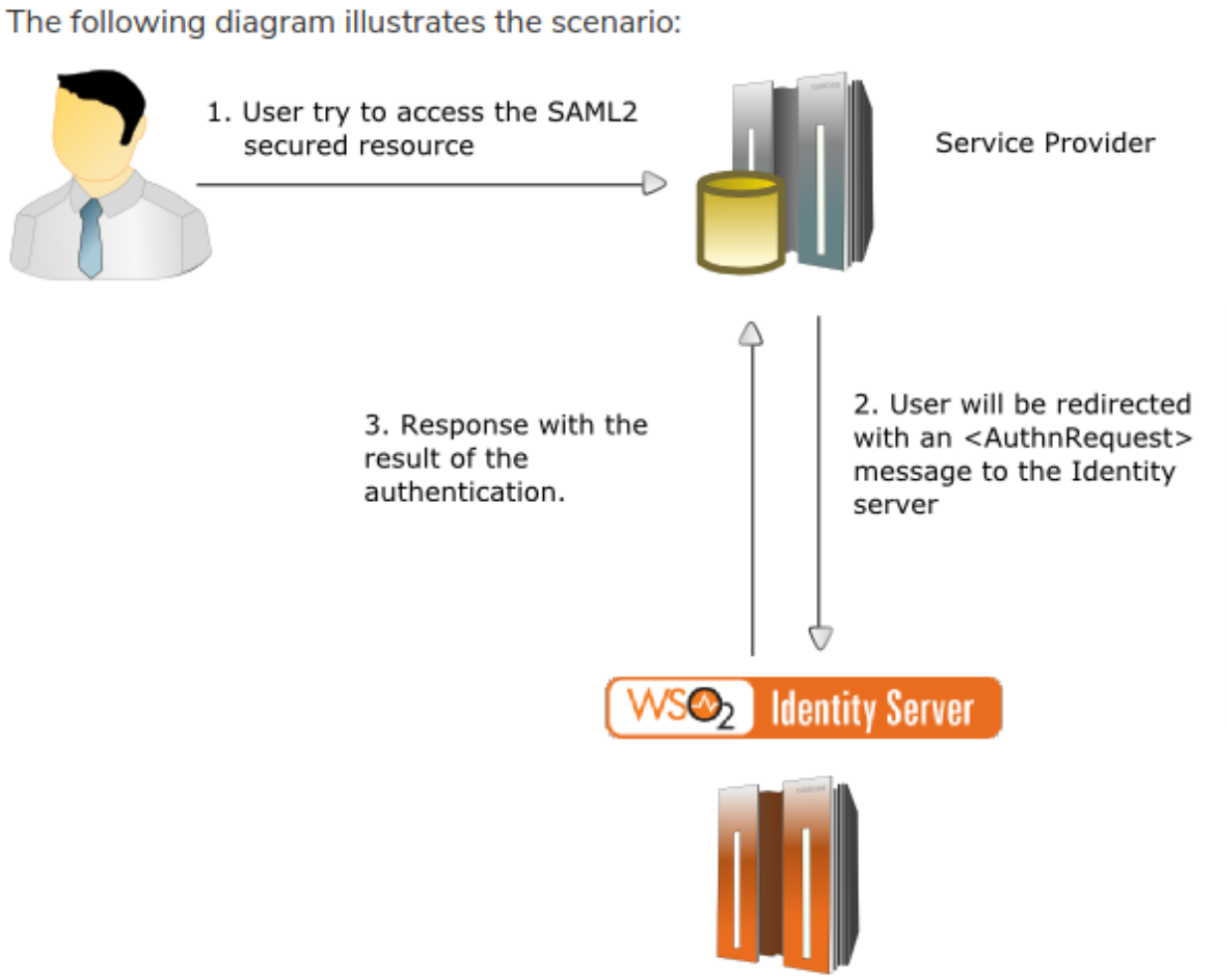

Fig. 5.1: La relazione tra i 3 soggetti del protocollo SAML

SPID – REGOLE TECNICHE

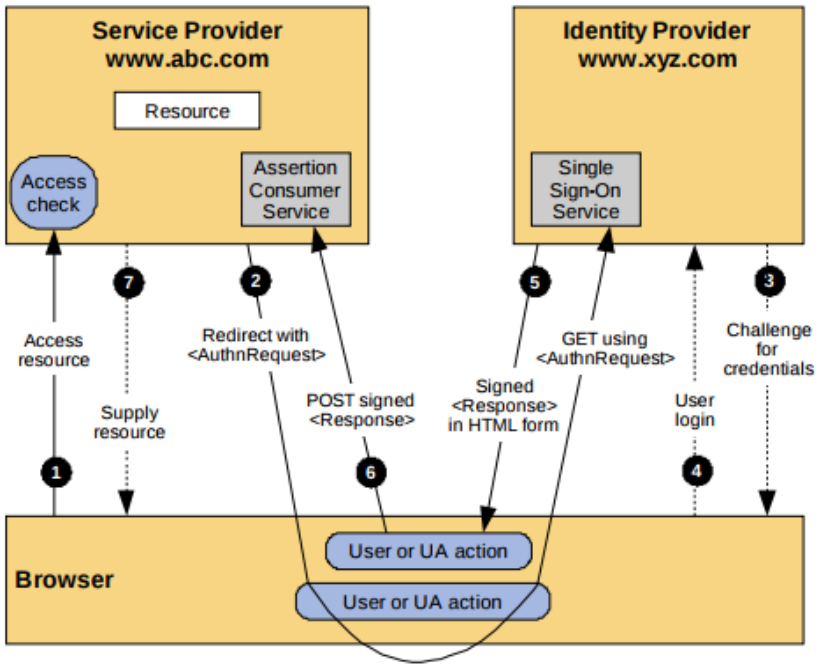

Figura 1 - SSO SP-Initiated Redirect/POST binding

Fig. 5.2: Le fasi del processo di autenticazione SAML

The <md:AssertionConsumerService> element is used to configure handlers that are responsible for consuming SAML assertions; that is, they process an assertion according to a profile, extract its contents, create a new user session, and typically produce a cookie to represent the session.

I metadati

Ogni **entità** del protocollo deve esporre dei metadati e questi sono differenti sia per tipologia che per contenuto. Infatti ogni entità può assumere uno dei ruoli:

- SSO Identity Provider
- SSO Service Provider
- Authentication Authority
- Attribute Authority
- Policy Decision Point
- Affiliation

e a seconda di quelli assunti specifica degli elementi XML differenti ad esempio:

- SPSSODescriptor: per le funzioni di Service Provider
- IDPSSODescriptor: per le funzioni di Identity Provider

Nell'SP si devono configurare i metadati dall'IdP e l'IdP deve avere i metadati dell'SP

Altro

Ulteriori informazioni posso essere trovate:

- https://wiki.oasis-open.org/security/FrontPage
- https://docs.spring.io/spring-security-saml/docs/current/reference/html/configuration-metadata.html

CHAPTER 6

Indices and tables

- genindex
- modindex
- search