

# Virtualized Web Portals in EGI Federated Cloud

Aleš Křenek, Radim Peša, Tomáš Raček, Vlastimil Holer, Daniel Kouřil,  
Lubomír Ontkoc

MUSTweek, Brno, March 5–10

# Security

## Motivation

- ▶ Various views
  - ▶ Users, service operator
- ▶ Different requirements
- ▶ We only show low-level bricks, always consider your needs

## Security functions

- ▶ Protection of information sent
  - ▶ encryption and/or integrity protection
- ▶ Authentication (of client and/or server)
- ▶ Access control

# Security in Web Portals

- ▶ Two possible levels to address security requirements
  - ▶ Application itself (framework)
  - ▶ Web server (application container)
- ▶ Well-established approaches to security for www
- ▶ Protection of information
  - ▶ HTTP over TLS/SSL (https)
- ▶ Authentication
  - ▶ X.509 certificates, password-based
  - ▶ Single Sign-On
- ▶ Access control
  - ▶ Fine-grained (usually ad-hoc on application level)
  - ▶ Coarse grained (usable in container)
  - ▶ Based on authenticated identity or additional information

# Using TLS for Web Portals

## Enabling TLS

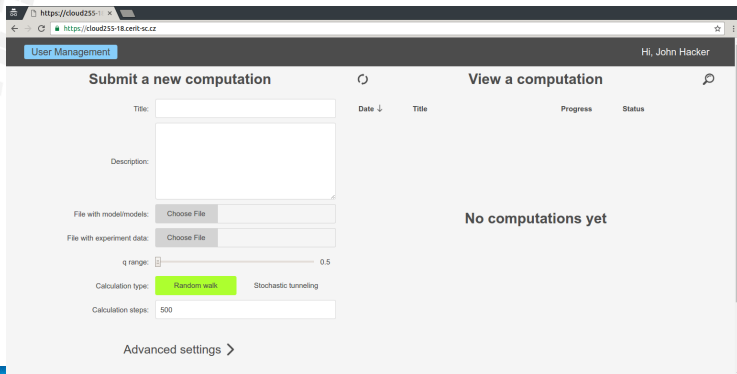
- ▶ Server-based authentication and channel encryption
- ▶ Client-side authentication also possible (out of scope for today)
- ▶ Requirements:
  - ▶ Digital certificate by a recognized CA
  - ▶ Automated process of getting the credentials

## Certification Authority

- ▶ Key cornerstone of traditional PKI
- ▶ CAs available differ in many aspects and certificate types
- ▶ IGTF provide a global trust platform for eScience but not accepted by common applications
- ▶ Let's encrypt CA available (if it matches the requirements)

# Security Assignment #1

- ▶ Deploy client to obtain X.509 server credentials from Let's encrypt CA
- ▶ Enable TLS/SSL support in Apache



# Enable TLS in Portal

## Short Recap

- ▶ Start from the configuration you finished yesterday
- ▶ Continue to use your Docker container ([radimpesa/mustweek2017](#))
- ▶ Weapons are available from [git@github.com:ICS-MU/westlife-mustweek2017.git](https://github.com/ICS-MU/westlife-mustweek2017.git)
- ▶ For slides see the [talks/](#) directory

## Actions to perform

- ▶ Start with two provided scripts in [security/](#) directory
  - ▶ [letsencrypt\\_setup.sh](#) – introducing let's encrypt client
  - ▶ [https\\_setup.sh](#) – enables TLS/SSL in Apache
- ▶ Both scripts are simplified, more care is needed for production use
- ▶ Extend saxsPortal to run these scripts during deployment

# Authentication & Access control

## Basic requirements

- ▶ Authentication is sufficiently user friendly
- ▶ Portal does not implement its own mechanisms for AAI
- ▶ Mechanisms work with dynamic portals

## Solution

- ▶ Utilization of federated authentication
- ▶ Authentication mediated by the *IdP-SP Proxy* of West-Life
- ▶ Configured on side of Apache, i.e. transparent for application

## Guide to federated AAI

- ▶ *Training for service providers* by AARC project

# Federated AAI – Summary

## Key terms

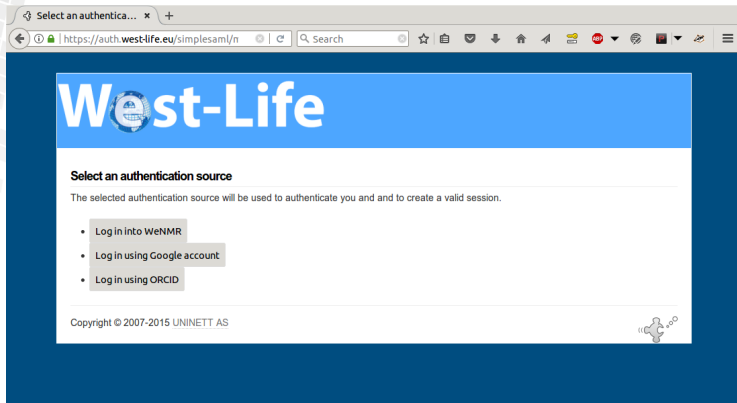
- ▶ Service Provider – SP
- ▶ Identity Provider – IdP
- ▶ Metadata (for SP and IdP)
- ▶ IdP-SP Proxy
- ▶ SAML
- ▶ Shibboleth, SimpleSAMLPhp, ...

## Enabling SAML on SP

- ▶ Configure SAML support on web server (or application)
- ▶ Enable selected IdP(s) (add corresponding metadata)
- ▶ Register with federation and/or IdP directly



# West-Life Idp-SP Proxy Service



# Security Assignment #2

- ▶ Enable SAML on the portal and link it with the West-Life IdP-SP Proxy

## Actions to perform

- ▶ Start with `saml_setup.sh`
- ▶ Extend saxsPortal to run the script
- ▶ After the portal is deployed perform additional steps:
  - ▶ Send SP metadata to IdP
    - ▶ You can find metadata of your SP at `https://\$HOSTNAME/mellon/metadata`
    - ▶ Check that the link is correct, and let us know
  - ▶ Verify the authentication after the SP has been registered
    - ▶ Visit `https://\$HOSTNAME/auth\_test/`
    - ▶ You should be redirected to IdP (`auth.west-life.eu`)
    - ▶ Use your credentials to log in
    - ▶ You should be redirected back to SP and see an environment dump

# Access Control based on Attributes

- ▶ IdP release two pieces of information
  - ▶ Information about successful authentication
  - ▶ Additional attributes linked to the user
- ▶ Attributes provides further information, like name, email attributes, affiliation, ...
- ▶ Attributes issued by a Proxy may also carry VO specific information (group membership, etc.)
- ▶ Few catches though
  - ▶ Attribute release policy – improved lately
  - ▶ Different naming and/or semantics of attribute release by different IdP
- ▶ Utilization of attributes for access control
  - ▶ Access rules enforced by web server
  - ▶ Access control implemented by the application (e.g. mapping to internal identities).

## Security Assignment #2

- ▶ Take a look at the application and try use attributes returned by the IdP
- ▶ Consult [https://\\$HOSTNAME/auth\\_test/](https://$HOSTNAME/auth_test/) to see what attributes are returned by the IdP