Trust & Identity Incubator
(De)Provisioning users activity

Uros Stevanovic

Sprint Demo, 2020-07-02

Public

www.geant.org
Remote access
Remote access

“user”
Remote access

“user”

“resources”

www.geant.org
Remote access

“user”

“AAI”

“resources”

www.geant.org
Web vs non-web services

• Using web services typically involves “in-time” release of attributes
  • Up-to-date info (e.g. behind a proxy/IdP)
  • Flows are well defined
  • Numerous activities are solving this problem

• Non-web services require “specialized” flow
  • Additional credentials (e.g. SSH keys)
  • Once created, users “bypassing” IdPs (or proxies)
Users using resources

- Provision (and deprovision) users:
  - Create accounts
  - Provide necessary info to services
  - Provide access credentials (e.g. SSH keys)
Users using resources

• Provision (and deprovision) users:
  • Create accounts
  • Provide necessary info to services
  • Provide access credentials (e.g. SSH keys)

• Information up-to-date?
Users using resources

- Provision (and deprovision) users:
  - Create accounts
  - Provide necessary info to services
  - Provide access credentials (e.g. SSH keys)
- Information up-to-date?
- Deprovision users?
(De)Provision users activity

- Two (at least) solutions:
  - PERUN
  - FEUDAL
- Decentralized vs Centralized
- Asynchronous vs sequential
- General vs per-service data format
PERUN

• Membership Management Service
  • VO and group management
  • User management
  • Service management
  • Customizable
• Synchronization to/from external source (e.g. LDAP)
• REST API
• Developed and maintained by CESNET
• One use case: MMS for a proxy
• Typically “hidden” from a “regular” user
### Personal:

<table>
<thead>
<tr>
<th>Organization</th>
<th>Masarykova univerzita</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workplace</td>
<td>FI</td>
</tr>
<tr>
<td>Research group</td>
<td>MetaCentrum staff</td>
</tr>
<tr>
<td>Preferred mail</td>
<td><a href="mailto:slavek@ios.muni.cz">slavek@ios.muni.cz</a></td>
</tr>
<tr>
<td>Mail</td>
<td><a href="mailto:slavek@ios.muni.cz">slavek@ios.muni.cz</a></td>
</tr>
<tr>
<td>Phone</td>
<td>+420 724 062 225</td>
</tr>
<tr>
<td>Address</td>
<td>N/A</td>
</tr>
<tr>
<td>Preferred language</td>
<td>en</td>
</tr>
<tr>
<td>LoA</td>
<td>2 (verified identity)</td>
</tr>
<tr>
<td>EDU person affiliation</td>
<td>N/A</td>
</tr>
</tbody>
</table>

### Membership:

<table>
<thead>
<tr>
<th>Status</th>
<th>VALID change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Member is properly configured and have access on provided resources.</td>
<td></td>
</tr>
<tr>
<td>Expiration</td>
<td>never change</td>
</tr>
<tr>
<td>Member type</td>
<td>Person</td>
</tr>
<tr>
<td>Sponsored by</td>
<td>N/A</td>
</tr>
<tr>
<td>Member ID</td>
<td>19179</td>
</tr>
<tr>
<td>User ID</td>
<td>3255</td>
</tr>
<tr>
<td>Password reset</td>
<td><img src="image" alt="Send password reset request..." /></td>
</tr>
</tbody>
</table>
PERUN User Provisioning

- Flow is not “user-centric”, i.e. deployment is typically not decided by the user

- Centralized model
  - Master + slave model
  - More tightly integrated (akin to “business environment”)
  - Trust level required between sites and PERUN is high
  - Customized communication (format per service)

- Synchronous deployment
  - Service needs to be online
  - Typically SSH connection to services
FEUDAL

• Federated User Credential Deployment Portal
  • Web application (OIDC client)
  • Provision/deprovision users
  • Deploy credentials (e.g. SSH)
  • AuthZ discrimination

• Architecture
  • Web portal (UI)
  • Backend+database (user info and credentials)
  • Clients (deployed service side) + “adapters”

• Pub-Sub
FEUDAL

- Web portal (user interaction point)
- REST API
- Backend
  * Django
- RabbitMQ (Pub-Sub)
- Clients (Go, Python, etc)
- Scripts
  * Adapters
- JSON data format:
  * Status
  * User info
  * credentials
**FEUDAL**

- **OIDC client**
- **User-centric flow**
  - Typically user is in control
  - Deployment per service, per VO
- **Decentralized model**
  - Server + client model
  - Clients run at sites (admin control), trust level not necessarily very high
  - Client only receives the info (user_info, JSON)
  - Standardized communication
- **Asynchronous communication**
  - Pub-sub, outgoing connection at clients
  - Flexible messaging (resending upon failure, onboarding, etc)
Side-by-side comparison

FEUDAL

• OIDC client
• User-centric flow
  • Typically user is in control
  • Deployment per service, per VO
• Decentralized model
  • Server + client model
  • Clients runs at sites (admin control), trust level not necessarily very high
  • Client only receives the info (user_info, JSON)
  • Standardized communication
• Asynchronous communication
  • Pub-sub, outgoing connection at clients
  • Flexible messaging (resending upon failure, onboarding, etc)

PERUN

• MMS (Membership management service)
  • Flow is not very “user-centric”, i.e. deployment is typically not decided by the user
• Centralized model
  • Master + slave model
  • More tightly integrated (akin to “business environment”)
  • Trust level required between sites and PERUN is higher
  • Customized communication (format per service)
• Synchronous deployment
  • Service needs to be online
  • Typically SSH connection to services
Usage consideration

- FEUDAL and PERUN have complementing flows/use cases
- Tight integration, easy-to-understand deployment, easy VO deployment → PERUN
- Flexible model, user may decide, asynchronous decentralized communication → FEUDAL
Usage consideration

- FEUDAL and PERUN have complementing flows/use cases
- Tight integration, easy-to-understand deployment, easy VO deployment → PERUN
- Flexible model, user may decide, asynchronous decentralized communication → FEUDAL

How to proceed?
Use cases

• Not PERUN or FEUDAL, but both
• Centralized model needed/expected/reasonable ➔ PERUN:
  • Cloud apps (e.g. GSuites)
  • Mail lists
  • LDAP (executed by PERUN)
  • Windows apps
• Decentralized model ➔ FEUDAL:
  • Provision users for SSH access to VMs
  • LDAP (executed on site’s side)
  • Mail lists (via LDAP)
  • Further plugins
Up-to-date info / Deprovision users

• PERUN + FEUDAL:
  • PERUN is an MMS (users’ info is up-to-date)
  • FEUDAL is a “client” of PERUN (or other MMS)
• Centralized Model → PERUN directly executes action
• Decentralized Model → PERUN via FEUDAL updates info
<table>
<thead>
<tr>
<th>PATH</th>
<th>METHOD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>at/</td>
<td>PUT</td>
<td>Update a user using an access token. The access token is used to retrieve an up-to-date userinfo.</td>
</tr>
<tr>
<td>userinfo/</td>
<td>PUT</td>
<td>Update a user using a plain userinfo.</td>
</tr>
<tr>
<td>users/</td>
<td>GET</td>
<td>Retrieve the subjects of the registered users. Can be filtered by vo.</td>
</tr>
<tr>
<td>users/?vo=&lt;vo&gt;</td>
<td>GET</td>
<td>Retrieve the subjects of the registered users. Can be filtered by vo.</td>
</tr>
<tr>
<td>user/&lt;sub&gt;/</td>
<td>GET</td>
<td>Check if the user with sub &lt;sub&gt; is registered.</td>
</tr>
<tr>
<td></td>
<td>DELETE</td>
<td>Delete the user with sub &lt;sub&gt; from feudal.</td>
</tr>
</tbody>
</table>
API

- JSON based API, userinfo
- API:
  - Get all users (also per VO)
  - Update user info
  - Delete a user
  - Check if user exists

```json
{
  "userinfo": {
    "iss": "https://proxy.acc.eduteams.org",
    "sub": "<sub>@eduteams.org",
    "name": "Uros Stevanovic",
    "given_name": "Uros",
    "family_name": "Stevanovic",
    "email": "uros.stevanovic@kit.edu",
    "ssh_key": "<some_key>",
    "eduperson_entitlement": [ "<group1>",
                               "<group2>"
                           ],
    "eduperson_targeted_id": [ "<some string>@eduteams.org" ],
    "eduperson_principal_name": "urost@acc.eduteams.org",
    "edupersonScopedAffiliation": [ "member@acc.eduteams.org" ]
  }
}
```
DEMO

- FEUDAL “look and feel” + SSH use case
- FEUDAL API (MMS + FEUDAL)
- FEUDAL update of user info
- FEUDAL LDAP use case
Achievements

• Provision users, via PERUN, FEUDAL, or PERUN+FEUDAL:
  • Cloud applications
  • LDAP (+ Mail lists)
  • Access to VMs (SSH)
  • Windows applications
• Centralized + Decentralized
• Up-to-date info
• Deprovision users
Thank you

Any questions?

www.geant.org