Virtlab: Proposal for Evolution

José Quintino Rogado

jose.rogado@ulusofona.pt

SITI Research Center

ECATI - Computer Engineering

Universidade Lusófona - Lisboa

Presentation Outline

- Brief Project Summary
- Current Status
- Some Technical Details
- Identified Issues
- Integration with other Applications
- Platform Extension and Deployment
- Final Remarks and Evolution

Project Brief Summary

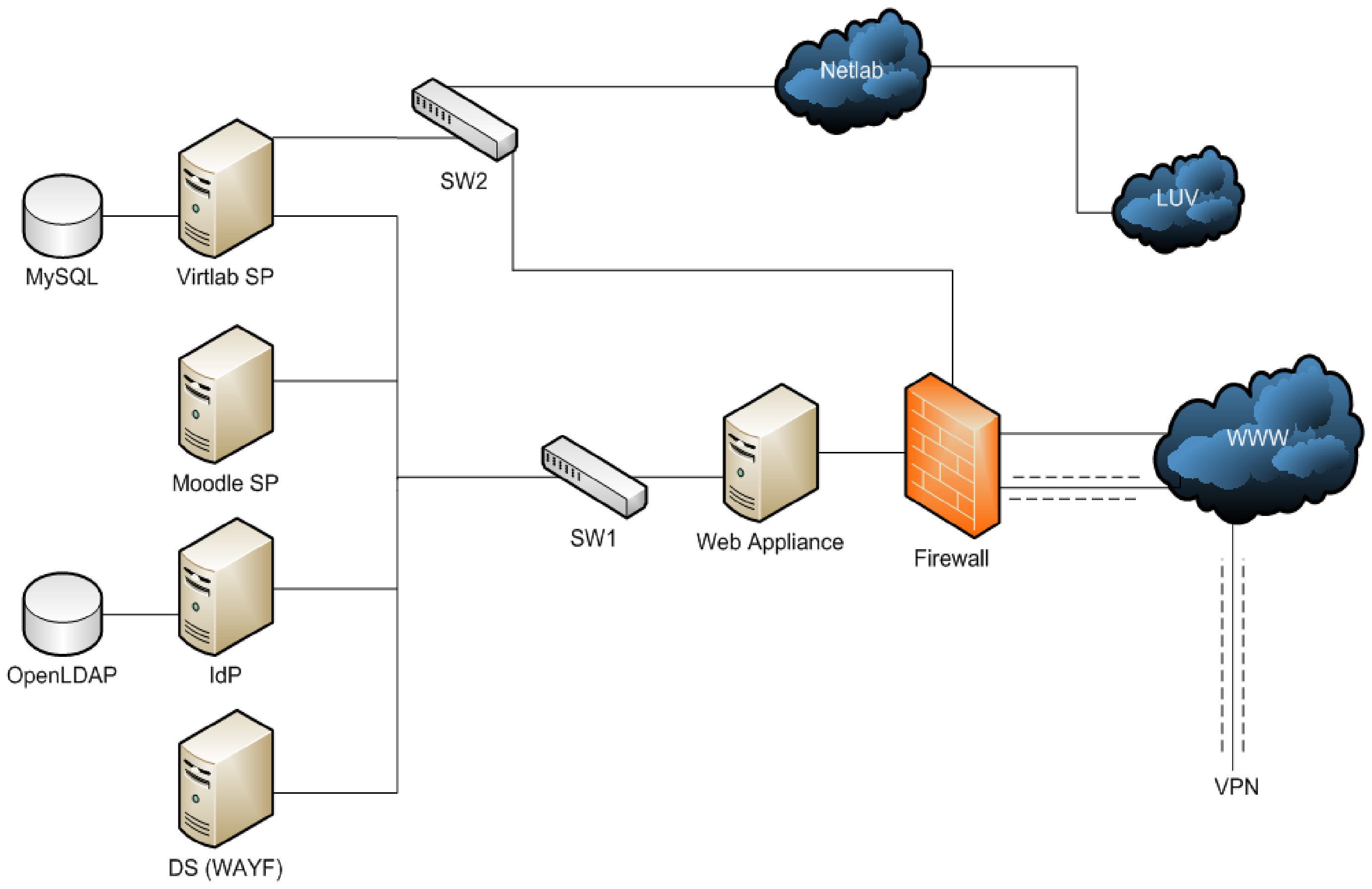
- **Virtlab** is an ongoing project at Universidade Lusófona (ECATI)
  - First concept presented in April 2009
  - Development started in early 2010
- The project initial goals were:
  - To provide e-Learning type access to the laboratorial infrastructure used for computer engineering courses
  - Use virtualization technology, preferably open source
  - Implement authenticated and selective access to predefined Lab resources based on student profiles
  - Allow access from other federated e-Learning platforms
- Additional requirements
  - Integrate with other university platforms (e-Learning, NetPA)
  - Extend the virtualized environments to include generic systems and applications
  - Provide desktop virtualization to support minimal client access

Current Status

- As of today, a working demonstrator is available…
  - Provides a valid proof of concept of the envisioned functionalities
- …which represents a lot of work!
  - Two MSc Thesis
  - Integrates results of a couple of Final Graduation Projects (TFC)
- But a lot of work is still needed to fulfill the initial requirements
  - The demonstrator runs in the enclosed environment of a lab
  - Presents some security flaws
  - Uses mock repositories for storing user authentication credentials and profiles
  - Demonstrates SSO with a demo Moodle installation
- Does not integrate with real ULHT repositories or applications
  - But the functionality demonstrated so far is worth pursuing the effort!

Demonstrator Infrastructure

Source: “*Federated Virtual Environments” M.Sc. Thesis, José Faísca*



Core Demonstrator Functionalities

- A robust virtualization infrastructure built only on Open Source components
  - Linux, KVM, QEMU, *libvirt*
  - Supports several virtualization paradigms (Hypervisor, Containers)
- Access control to the virtualization infrastructure is done through the SAML/Shibboleth authentication protocol
  - Virtualization environment is protected by the “Service Provider” Shibboleth component, coupled with a VM Broker module
- Access to the infrastructure is performed through a Web Interface
  - Authentication and authorization follows the HTTP oriented Web SSO Shibboleth protocol
  - The rights obtained are further *delegated* to a Java applet that implements remote non-web access to the virtualized environment

Additional Functionalities

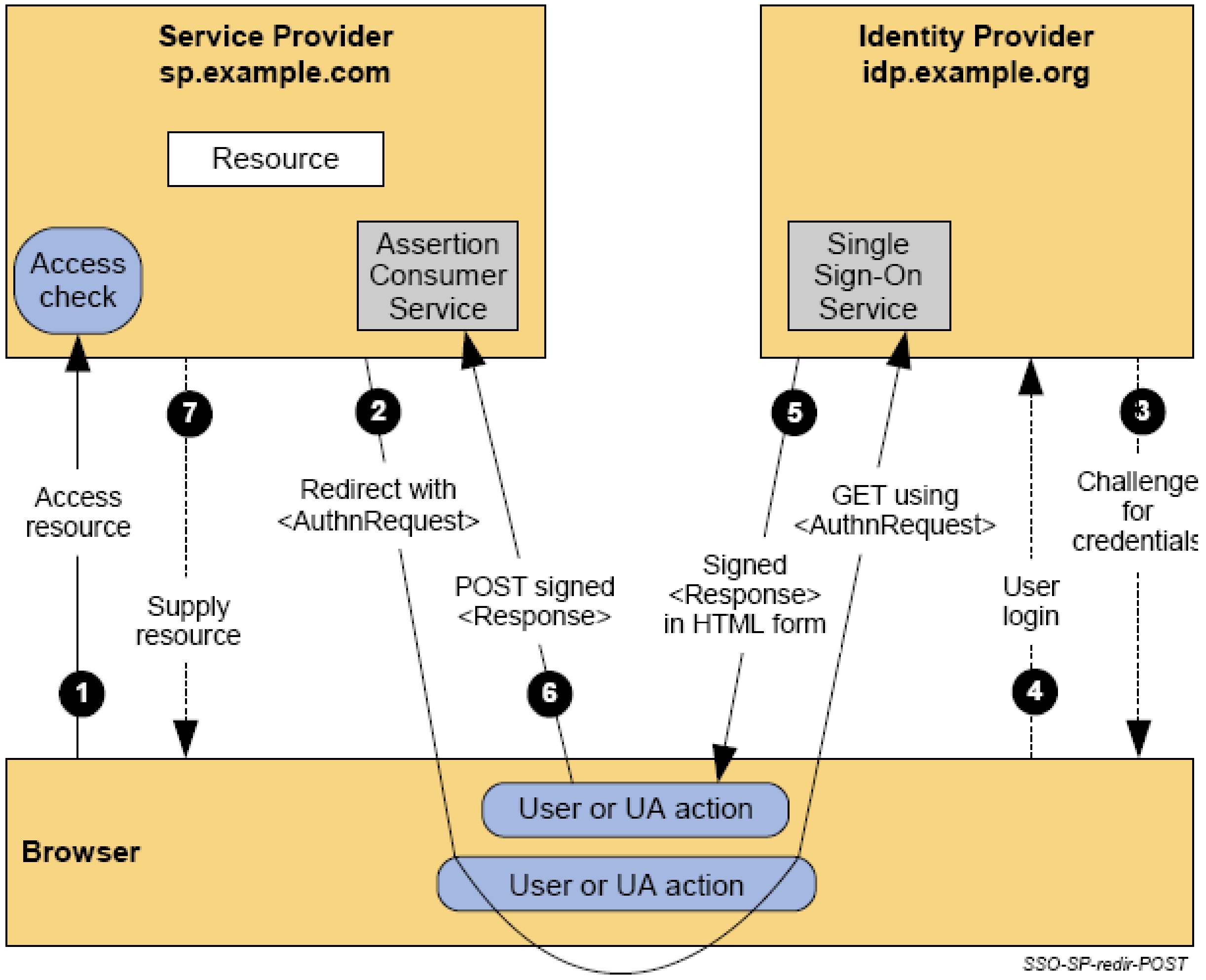
- A Web based management component
  - Interfaces the local profile database for provisioning and management of user profiles and access rights
  - Interfaces the *libvirt* library to configure the virtual machines attributes and control their execution
- All secure traffic is routed through a “Reverse Proxy”
  - Acts as a SSL/TLS concentrator releasing the rest of the components from heavy encryption tasks
  - Supports load balancing for solving scalability issues
- Another project spawned from the Virtlab activity
  - Definition of a minimal graphical end-user environment to access the virtual environment
  - Will probably be addressed in a future presentation

A Few Technical Details

- The SAML/Shibboleth authentication sequence involves four entities:
  1. The user agent (usually a browser)
  2. A resource to which the user requests access
  3. An Identity Provider (IdP) that connects to a credential repository
  4. A Service Provider (SP) that protects the resource
- The IdP is responsible for:
  - Performing user authentication against the repository
  - Collecting user profile attributes (possibly from other repositories)
  - Exporting this information in the form of a signed *Assertion* to the SP
- The SP is responsible for:
  - Redirecting non-authenticated users to the IdP
  - Consuming the *Assertion* provided by the IdP
  - Granting or denying access to the resource based on the user profile and access policies
- Several forms of exporting the *assertion* are possible

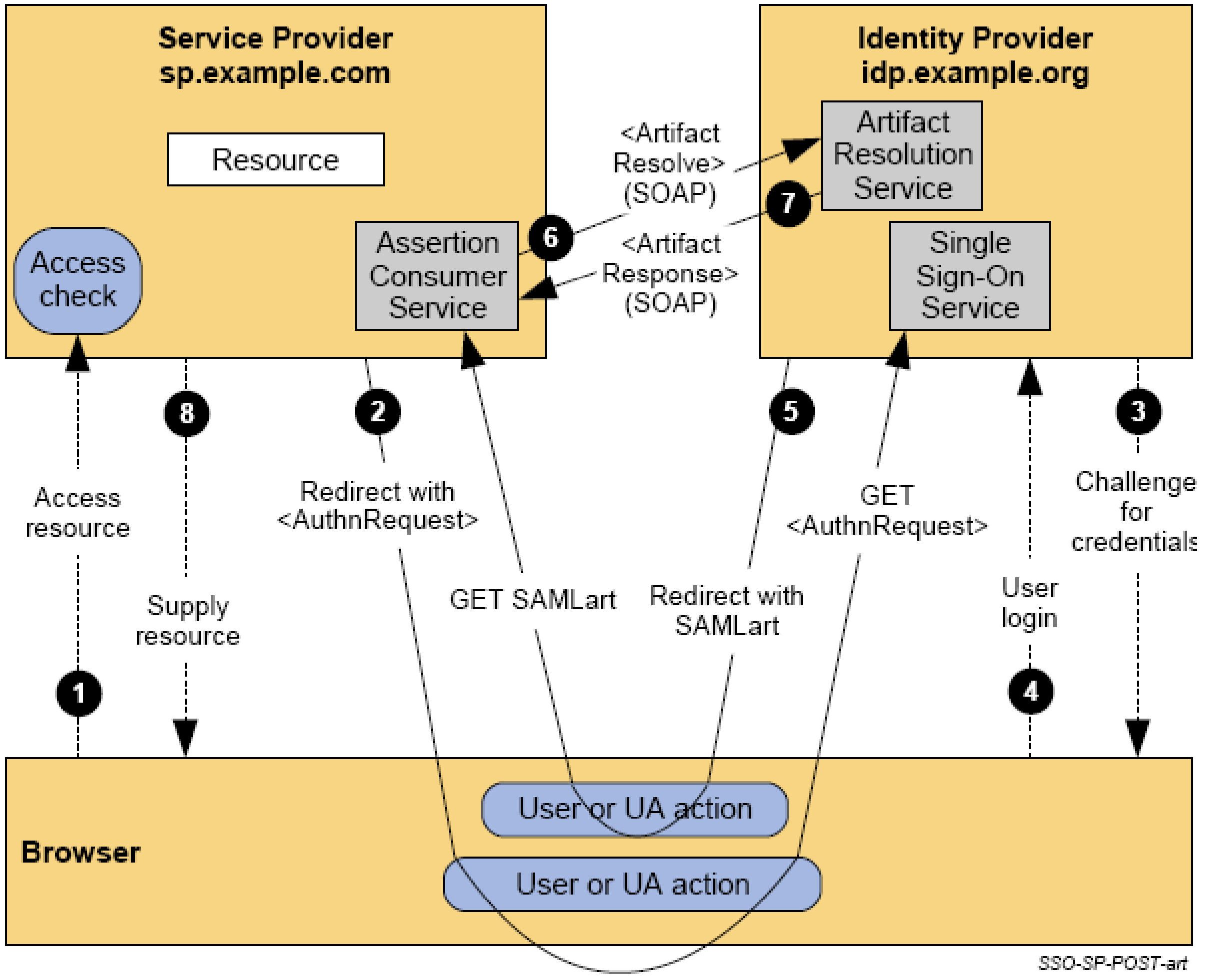
SAML SSO Redirect Sequence

Source: *SAML V2.0 Technical Overview* - OASIS



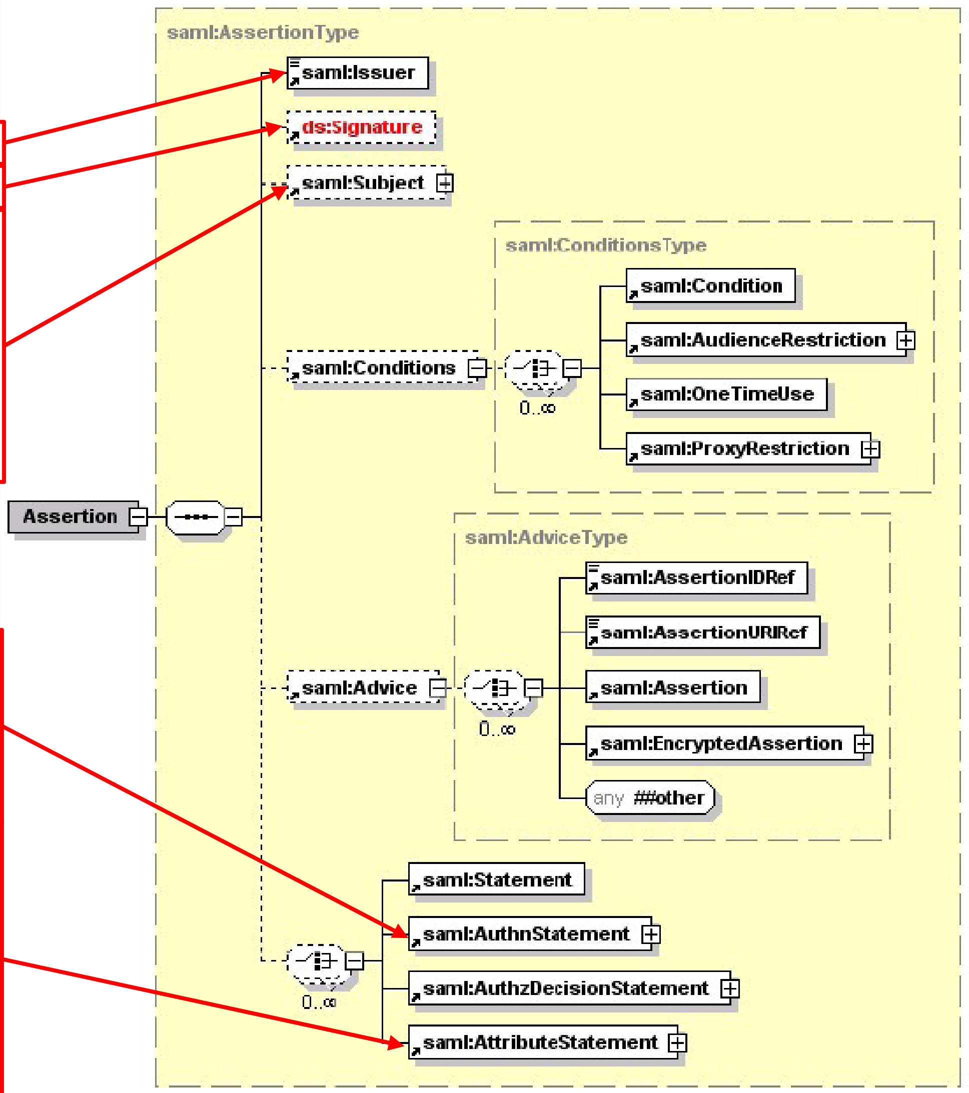
SAML SSO Redirect w/ Artifact

Source: *SAML V2.0 Technical Overview* - OASIS



Assertion Example

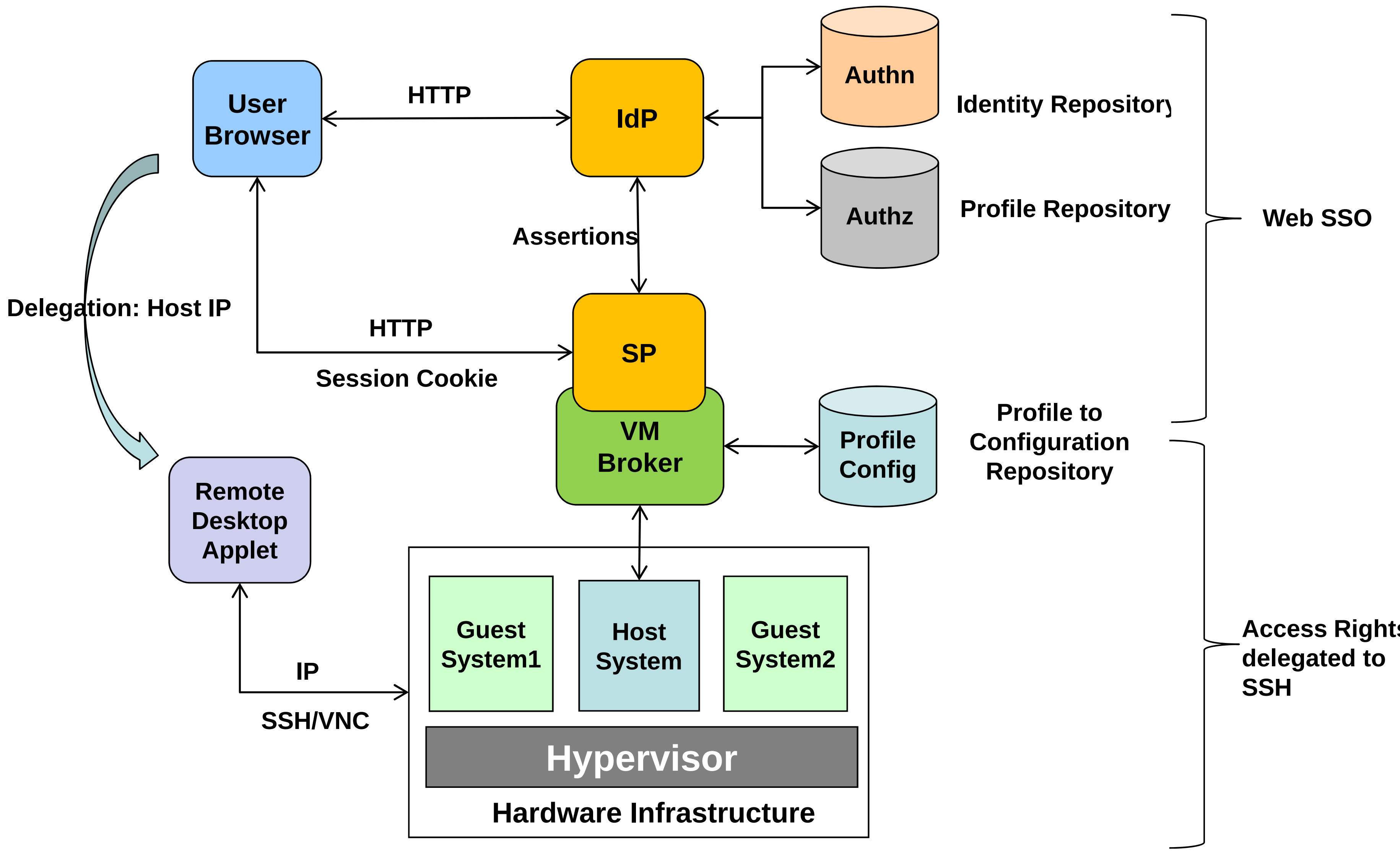
```xml
<saml:Assertion
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  ID="b07b804c-7c29-ea16-7300-4f3d6f7928ac"
  Version="2.0"
  IssueInstant="2004-12-05T09:22:05Z">
  <saml:Issuer>https://idp.example.org/SAML2</saml:Issuer>
  <ds:Signature
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#">...</ds:Signature>
  <saml:Subject>
    <saml:NameID
      Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient">
      3f7b3dcf-1674-4ecd-92c8-1544f346baf8
    </saml:NameID>
    <saml:SubjectConfirmation
      Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
      <saml:SubjectConfirmationData
        InResponseTo="aaf23196-1773-2113-474a-fe114412ab72"
        Recipient="https://sp.example.com/SAML2/SSO/POST"
        NotOnOrAfter="2004-12-05T09:27:05Z"/>
    </saml:SubjectConfirmation>
  </saml:Subject>
  <saml:Conditions
    NotBefore="2004-12-05T09:17:05Z"
    NotOnOrAfter="2004-12-05T09:27:05Z">
    <saml:AudienceRestriction>
      <saml:Audience>https://sp.example.com/SAML2</saml:Audience>
    </saml:AudienceRestriction>
  </saml:Conditions>
  <saml:AuthnStatement
    AuthnInstant="2004-12-05T09:22:00Z"
    SessionIndex="b07b804c-7c29-ea16-7300-4f3d6f7928ac">
    <saml:AuthnContext>
      <saml:AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
     </saml:AuthnContextClassRef>
    </saml:AuthnContext>
  </saml:AuthnStatement>
  <saml:AttributeStatement>
    <saml:Attribute
      xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
      x500:Encoding="LDAP"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      Name="urn:oid:1.3.6.1.4.1.5923.1.1.1.1"
      FriendlyName="eduPersonAffiliation">
      <saml:AttributeValue
        xsi:type="xs:string">member</saml:AttributeValue>
      <saml:AttributeValue
        xsi:type="xs:string">staff</saml:AttributeValue>
    </saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>
```



Issue: Virtlab Access Delegation

- Authentication and authorization are granted by the SAML *assertion*
  - A session token is then sent to the browser by the resource manager
  - Materializes the access rights to the required resource
  - Only valid for subsequent HTTP accesses to the resource manager
- The remote access to the virtualized host does not use HTTP
  - The remote desktop runs in a Java Applet which connects to the virtualized host through a VNC protocol tunneled in a SSH connection
- The access rights received with the assertion are ***delegated*** to a Secure Shell (ssh) connection
  - In the form of the host IP address provided to the applet
  - The user still has to authenticate to the host system using a generic username/password (can be easily changed)
- This access is not secure and breaks the SSO paradigm created by SAML/Shibboleth
  - Another more secure form of delegation has to be implemented

SSO and Delegation



Simple Straightforward Solution

- Change delegated credentials
  - IP + **SSH private Key**
  - Could be encrypted and inserted in auth cookie
  - Matching public Key must be inserted in host generic user home dir
- Advantages
  - Relatively easy to implement
  - Removes spurious login to host system
  - Slightly increases security level
- Drawbacks:
  - One key for all user sessions, otherwise every user needs an account on virtualization host
  - No mechanism for key rotation, although a key pair substitution could be performed periodically (i.e.: 24h)
  - Needs additional functionality in VM Broker
  - Not completely secure

More Secure Solutions

- Implement a PAM (Plugabble Authentication Module) for Linux following SAML/Shibboleth protocol
  - Interesting and useful functionality
  - Some developments already exist in this field:
    - Emmanuel Dreyfus **Crude Saml** available as a NetBSD package
      - http://hcpnet.free.fr/pubz/#crudesaml
      - http://pkgsrc.se/security/pam-saml
      - http://groups.google.com/group/shibboleth-users/msg/afcdf2bcc85b4727
- Another approach: use Shibboleth Kerberos login handler
  - Work in progress at Internet2.edu
  - A Kerberos ticket becomes a Shibboleth assertion
  - Implements a reverse approach: delegation from host access to web access, implying modifications on current Virtlab approach
  - Allows SSO between Windows domains (or Linux + Kerberos PAM) and Web Applications
  - Only valid in Intranets (or DCE type delegation is needed…)

Issue: Authz Profile Management

- Currently, the Virtlab demonstrator uses a private database to store users, profiles and virtual resources
  - Authorization is performed per user
  - Modified for each user provisioned in the system
- Although this approach is sufficient for a valid proof of concept, it is not suited for a live deployment
  - Thousands of users exist
  - Impossible to scale
- In a live deployment:
  - Authn must be performed against ULHT “official” IdP
    - Not an issue, since the protocol is the same
  - Authz must be based on profiles imported from various sources:
    - Active Directory used by IdP has profile basic info
    - Other repositories need to be queried to create the whole profile
    - Course enrollment attributes is crucial for authorization decisions
      - Moodle uses this info to decide which students access which resources

Possible Solution

- Introduce Profile Types (~Roles)
  - Access is granted based on profile type and not on individual attributes
    - RBAC (Role Bases Access Control) approach
  - Local Provisioning still needed, but with reduced number of entries
- Solving this issue implies:
  - Modifying the VM Broker access control paradigm
  - Modifying the metadata profile propagated by SAML assertions
  - Unification of profile management ULHT wide
- This approach does not solve the problem of multiple repositories and profile information used by ULHT applications
  - A global effort for repository unification is needed and has been initiated with the migration of profiles to an Active Directory repository
  - Ideally, the definition of profile types and user allocation to profiles should be globally performed
- Typically an Identity Management issue common in large corporations

Plans for Evolution

- The issues described so far are normal in the current phase of the project, where the development is still in progress
  - Do not imply fundamental modifications to the core architecture
  - Can be solved through additional development investment
- With this issues solved, the resulting platform constitutes an advanced foundation for leveraging other innovative projects
- Implementing a generic Authentication and Authorization Infrastructure
  - To provide Single Sign-On between Web (and possibly non-web) applications
- Providing an Open Source virtualization environment for hosting generic environments and applications
  - To aggregate and optimize heterogeneous hardware platforms
- Creating a seamless and unified secure access to campus resources
  - That can be used inside or outside the university premises

SSO Generalization

- Currently, every ULHT Web application has its own authentication and authorization method
  - Possibly using distinct credential repositories
- As a mid term goal (~1 year) SSO could be extended at least to the following two:
  - Moodle - the e-Leaning platform
  - NetPA - the corporate portal for student and academic activity government (timetables, inscriptions, class summaries, etc…)
- Adapting Moodle to Shiboleth SSO has already been done, and is part of the demonstrator functionality
  - But needs to be adapted to the much more recent and heavily customized deployed version of the platform
  - Implies profile unification mentioned previously
- Adapting NetPA is a more complex task, although technically feasible
  - Involves modifying proprietary Java code
  - Access is not free and involves an external enterprise

Extension of VM Environments

- The demonstrator contains simple and minimal OS environments
  - Sufficient for Computer Engineering classes (Linux and Windows)
- Remote Access is implemented by a simple applet
  - Mostly suited for command line and simple GUI access
- To support richer environments, including application development and/or sophisticated graphical interfaces, involves
  - Supporting Desktop Virtualization (provisioning, administration, …)
  - Optimizing remote access using an advanced Remote Desktop Protocol (RDP, RFB, …) and possibly a client side application
  - Providing client side graphical hardware acceleration, possibly using PCoIP technology
- These aspects will be addressed by the S-CASE project
  - The Virtlab infrastructure can be used as a first deployment environment
  - Close interaction will be needed and expected

Platform Deployment Requirements

- This evolution is only possible if the extended platform is deployed ULHT wide
  - Some requirements are mandatory
- The campus network architecture has to be optimized and possibly reconfigured to support the platform specificities
  - Secure traffic routed through the Reverse Proxy
  - Multiple instances of critical platform resources with load balancing support (Reverse Proxy, IdP, SP, …)
  - Several public IP addresses, with specific firewall rules (SSL, SSH)
- Scalable Hardware Infrastructure
  - As the number of users and applications increases, a dedicated and scalable cluster of machines needs to be dimensioned and configured
  - Maintaining multiple environments of which several versions can be instantiated implies a large storage capacity
- In order to evaluate load and traffic requirements, a pilot deployment involving a limited number of users should be considered

Final Remarks

- This proposal for evolution consists of three parts
  1. Redesign and reimplementation of parts of the demonstrator
  2. Extension of its functionalities to support a wider range of utilization
  3. Changing parts of the ULHT IT infrastructure to support a correct deployment of the extended platform
- Items 1 and 2 are basically advanced development tasks
  - Involving security, virtualization and programming skills
  - Can possibly be achieved in the context of M.Sc. Thesis
- Item 3 is a more complex task since it implies:
  - Institutional decisions that may take time to take
  - Direct participation of IT staff
  - Possibly important investment in hardware components
  - May have enormous impact on the whole institution core activity
- But if successful, it can bring important benefits to the University learning and computing environments

Possible Evolutions

The implementation of the functionalities described in this presentation enable the application of the following principles:

- Dematerialization of the hardware infrastructure
  - All configurable and specific hardware is removed from the classrooms and offices and replaced by a managed cluster of servers located in a data center
- Virtualization of the system and application software
  - All configurable and specific software does not run on client machines but on virtualized platforms supported by the consolidated hardware infrastructure
- Unified authentication and access control
  - All users credentials and profiles are unified and stored in an unique repository, and access to resources is governed by global policy rules that are specific to roles and not users

Taken as a whole, these are the basic principles that define and enable a Cloud Computing environment…