ESO SSO Implementation

Paul Harrison ESO

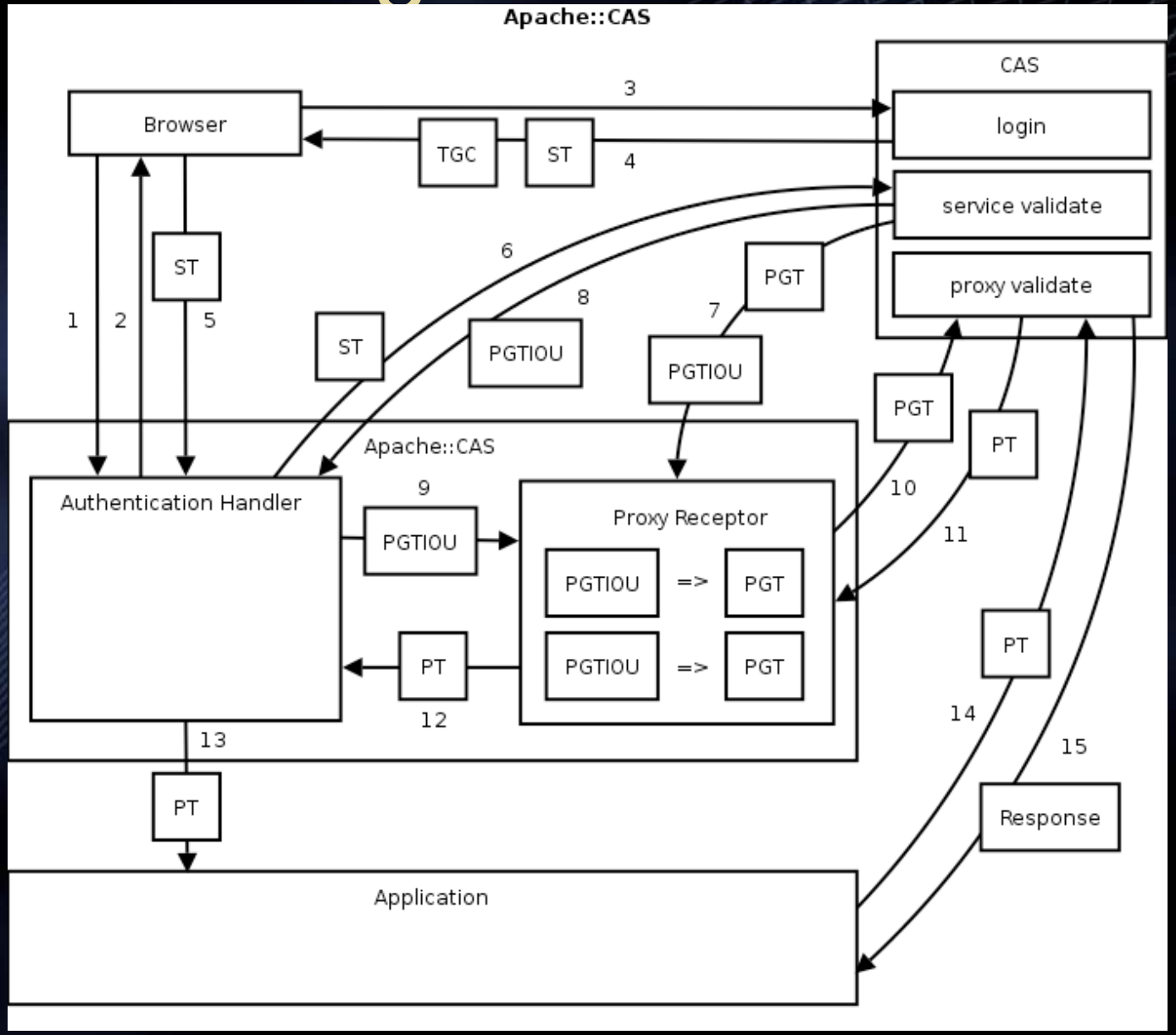
SSO

- ESO focus from different direction
  - Existing user database
  - Applications are mainly traditional web portal
    - not web services
    - even Perl cgi!
  - However different applications using different authorization/authentication mechanisms
  - Drive is to create a web portal single sign on (SSO) system.
- Add VO SSO compatibility later - federation between ESO identity and VO identity is the next step.
  - Browser based SSO and service to service SSO use different mechanisms

Web Portal/Browser SSO

- Evaluated many different systems
  - Decision dictated by Portal needs.
- Using CAS developed at Yale (not Globus)
  - http://www.ja-sig.org/products/cas/
  - Reasonably mature
  - Clients for many platforms
  - Open and active
- For Java clients we hide all of the specifics behind the http://acegisecurity.org/ façade for Spring

CAS Messages



CAS issues

- No “single sign off”
  - User has to close down browser to ensure that all of the application sessions are deauthenticated
  - Could be added to CAS with a call-back mechanism to the applications, but complicated in practice by a user not necessarily knowing the scope of what they are logging out of.
- No pre-written integration with MyProxy

SSO interoperability

- Sign into web portal with existing (VO) certificate
  - Standard certificate easy
  - Proxy certificate less easy
    - Irritating to keep having to load into browser in same way as standard cert (if possible at all).
    - Possibly an applet could be used to manage the proxy certificate.
- Sign into VO “grid” - need to do similar trick as was done with pubcookie and MyProxy

Temporary Users

- Want to be able to support VO user who has not previous ESO credentials in the ESO web portal
  - needs to be done by generating a temporary ID in the ESO web portal
- So we need to define a minimum set of common user metadata that is retrievable from VO "community" service to avoid having to ask user directly.
  - Schema
  - Standard interface
  - Have to know which "community" a particular user comes from
    - Proxy certificates provide the link

Trust Relationships/Identity

- Can be complex in practice, because we want to support

1. Users with simple logon/password - community service generates the basic certificate.
2. Users with “real” grid certificates - i.e. issued by national certification authority.

User can have two identities

- Probably start with first style and move to second
- What do do with data holdings

Need a mechanism for distributing/ recognising the “less trusted” CA roots.

- Software needs to be able to apply automatically a lower level of authorization to the type 1

Conclusions

- User should not be bothered with details - that is after all the aim of SSO
  - But it might be difficult to give single identity
  - And we need to define the interfaces and behaviours between our individual SSO efforts.