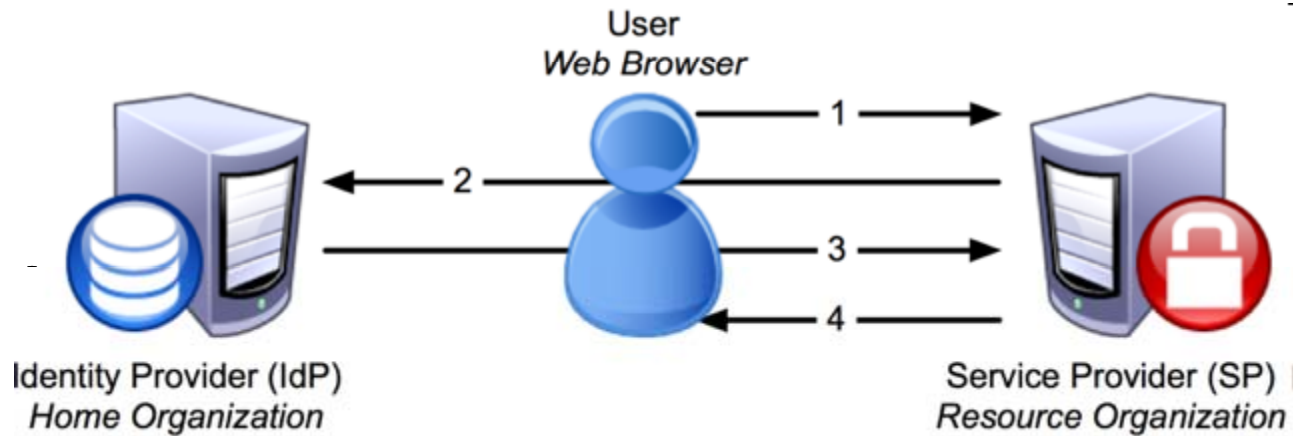


# Shibboleth Integration

# Shibboleth Basics



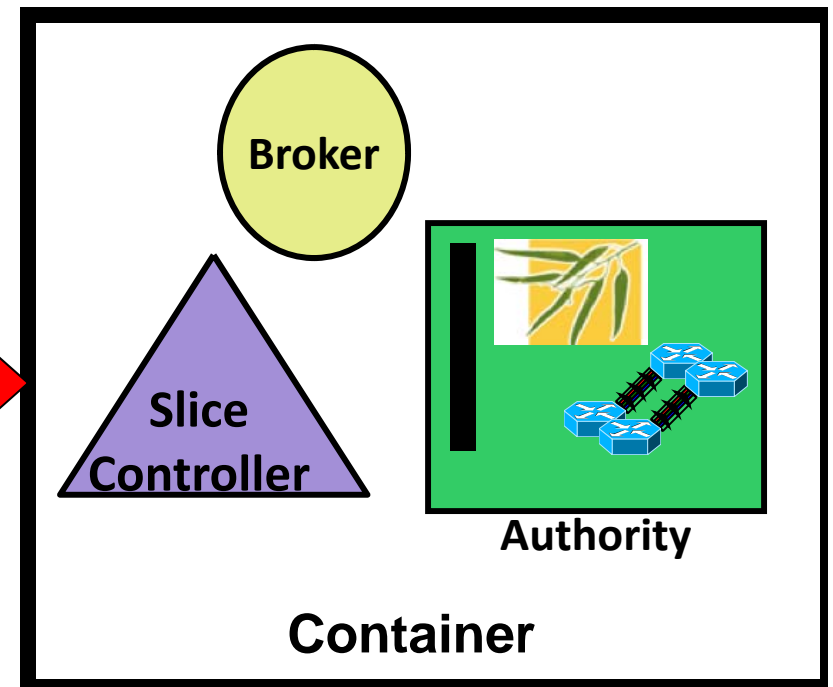
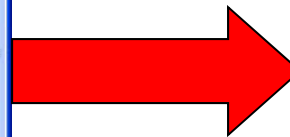
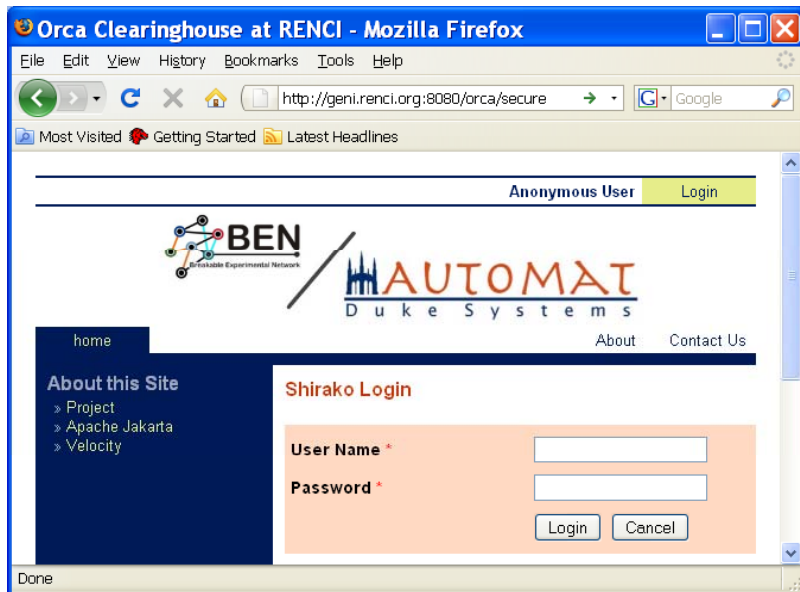
- Services need protecting
- Users need authentication
- Identity providers know user information
- Shibboleth glues it all together

# Orca today

The screenshot shows a Mozilla Firefox browser window titled "Orca Clearinghouse at RENCI - Mozilla Firefox". The address bar displays the URL "http://geni.renci.org:8080/orca/secure". The browser interface includes a menu bar (File, Edit, View, History, Bookmarks, Tools, Help), navigation buttons (back, forward, refresh, home), and a search bar with the Google logo. Below the browser window, the website content is visible. At the top right, it says "Anonymous User" and "Login". The main header features the "BEN Breakable Experimental Network" logo on the left and the "AUTOMAT Duke Systems" logo on the right. A dark blue navigation bar contains the text "home", "About", and "Contact Us". On the left side, there is a dark blue sidebar with the heading "About this Site" and a list of links: "» Project", "» Apache Jakarta", and "» Velocity". On the right side, there is a "Shirako Login" section with a light orange background. It contains two input fields labeled "User Name \*" and "Password \*", and two buttons labeled "Login" and "Cancel". The status bar at the bottom of the browser window shows "Done".

# Orca today

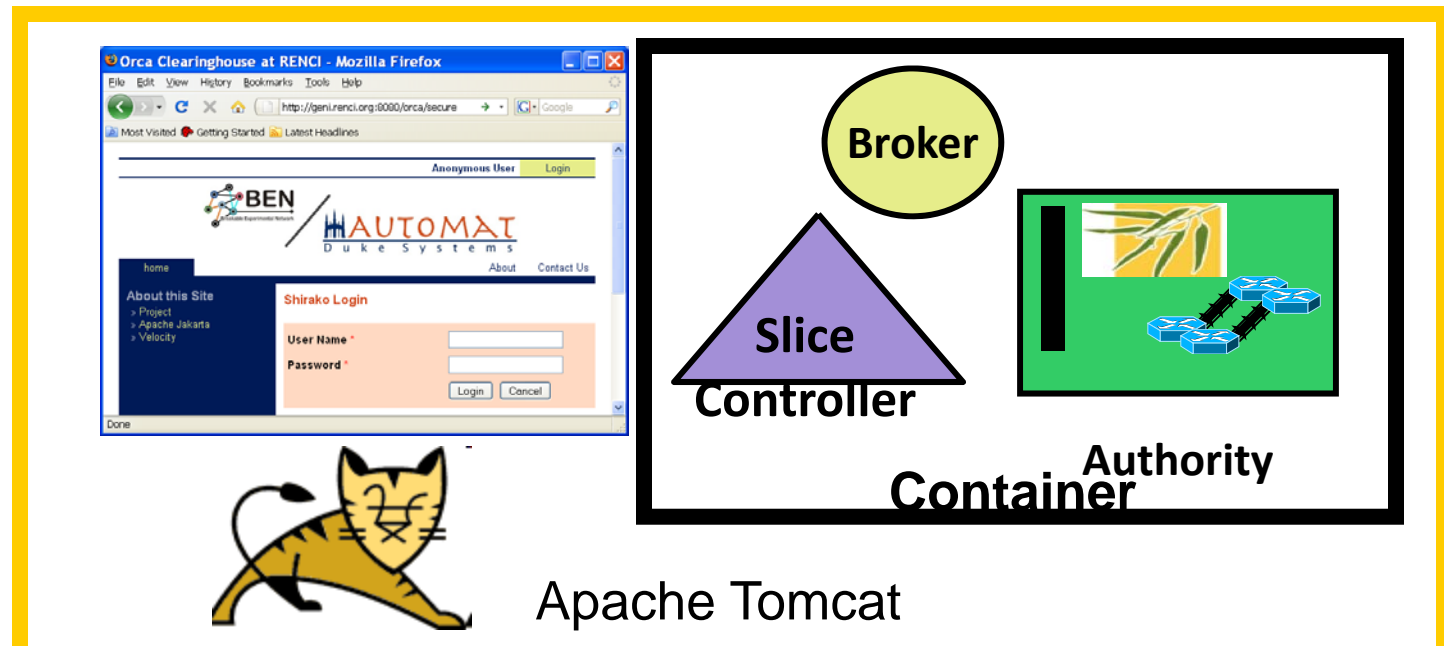
- User authenticates to portal
- Portal decides user capabilities
  - Role based access control



# Orca Authentication

- “Container managed security”
  - Tomcat, not ORCA containers
- Tomcat manages access to pages
- Portal manages roles

- “Container”
- “ContainerManager”
- “ContainerSecurity Manager” etc
  - Nothing to do with users



# Baby Steps

- Get Tomcat to use shibboleth
  - Instead of database-driven authentication
  - Requires Tomcat to understand shibboleth
- Mostly done already
  - Using OIOSAML from the Danish govt
  - Uses reference shib identity provider
    - Reuses same RBAC the portal already knows
- Investigate using the “real thing”
  - Several new non-standard capabilities
  - Significantly more setup, not quite saml2.0 compliant

# Flying Cars and Jetpacks

- Present user credentials to actors
  - Finer-grain access control
  - Policy decisions based on attributes
- Use shibboleth to distribute public keys
  - Let someone else solve PKI!
- Replace ORCAs ad-hoc trust brokering with formal SAML assertions?
  - What else?

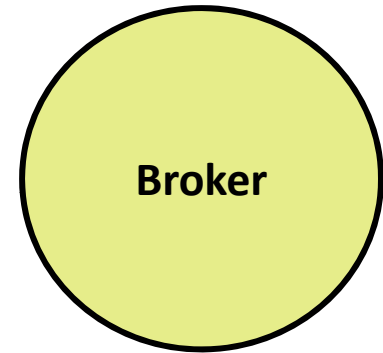
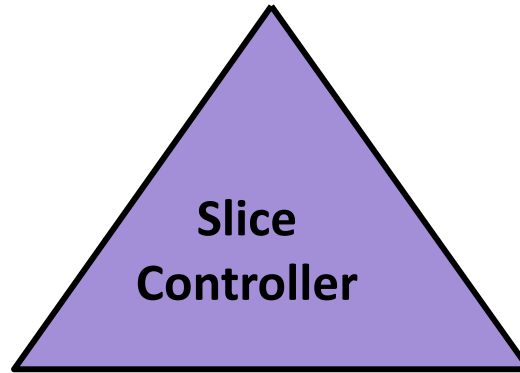
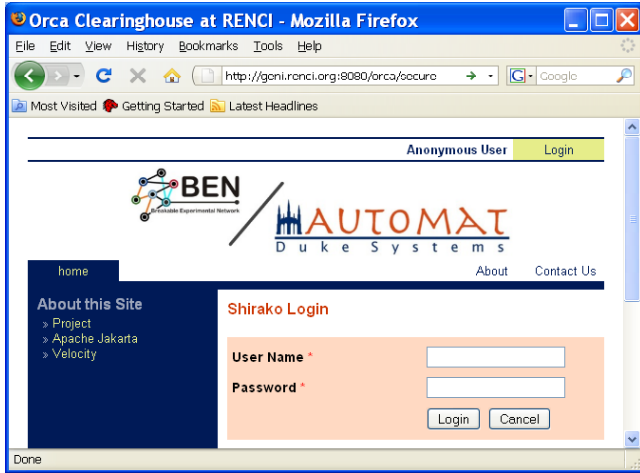
# Implementation Details

- Option 1: Leverage existing trust relations
- Minimal changes, quick-and-easy
- User authenticates to portal
- Portal sends identity/attributes to SC
- SC adds attributes to request properties
- Broker and/or AM looks at properties
- Basically, attributes instead of roles



# Implementation Details – II

- Option 2: Get shibboleth to mediate trust between actors
- Actors policies based user identity, in addition to peer identity.
- Shibboleth to mediate trust between actors
- “Delegated identities”
  - Only SC has to know the identity provider
  - Actors don’t have to do their own signing



**Identity Provider (IdP)**  
*Home Organization*

