Discussion about:
* Security Provisioning and Validation *
* Policy Enforcement Complexity *
* Data Integrity Verification *

11th Middleware Security Group Meeting
San Diego, CA - March 1-2, 2007

Frank Siebenlist, Argonne National Laboratory, franks@mcs.anl.gov
Rachana Ananthakrishnan, Argonne National Laboratory, ranantha@mcs.anl.gov
Michael Helm, ESnet, helm@es.net
Ian Foster, Argonne National Laboratory, foster@mcs.anl.gov

Contents

- **Trust Provisioning & Validation**

- Policy Enforcement Complexity

- Data Integrity Verification Facilities

Attribute-based Access Policy Example

- Enforced policy at ANL compute server:
  - Any ANL-staff has access to resource
- Questions:
  - Who asserts the user's name?
    - The Identity Authority
  - Who asserts the ANL-staff membership?
    - The Attribute Authority
  - Who renders the access decision?
    - The Authorization Authority

Authorization Authority

- Sample scenario:
  - Resource's Policy Enforcement Point calls out to an external authorization service
  - Authorization service's decision must be authenticated
  - Identity asserting the decision is **Authorization Authority**
- Other options
  - Push model Vs Pull model
  - Various communication protocol

Authorization Authority

- The resource needs to trust the external authorization service decision
  - configured through the Authorization Authority
  - service's network address, used protocol, pull/push are not relevant from a trust perspective
- If authorization service is local, then trust might be implicit

Attribute Authority

- Attributes about entities can be maintained and obtained from an external attribute service
  - Example: group membership attribute in VOMS server
  - Attribute consumer contacts external server to retrieve attributes
- Attribute service's group membership assertion must be authenticated
  - Identity asserting the membership is the associated **Attribute Authority**
- Attribute consumer needs to trust the attribute authority:
  - Configured through Attribute Authority
  - Access model does not matter

Identity Authority

- Cryptographic mechanisms for authentication
  - Prove possession of a secret
- Third party bind secret to name
  - PKI, Kerberos
- Name-to-secret assertion must be authenticated:
  - Identity asserting the name is the **Identity Authority**
- Resource needs to trust certain name assertions
  - Configured through the Identity Authority
  - For X509/PKIX, the Certification Authority (CA) asserts the name to public-key binding and defines the correct (and complicated) path validation

Trust-roots Configuration

- ***Trust-root*** implies decisions are derived from the initial trust in those authorities
- Resources have to be pre-configured with **trust-root** information ***before*** any policy can be enforced
  - Identity, Authorization, Attribute Authorities
  - Required for attribute-based authorization
- Servers/Clients require provisioning with the correct trust-root information at deployment
  - Static or dynamic provisioning
  - Periodic updates
  - Maintenance overhead

Signing-Policy Authority

- Resources will only trust authorities within a context defined by its own, local-site policy
  - E.g. ANL's policy will trust LBNL's CA only to sign identity certificates with a name constraint to LBNL's own organization
  - Equivalent policies about attributes
- Signing policy can be enforced in two ways:
  - By auditing of practices
  - Real-time enforcement, e.g. signing policy files Globus Toolkit
- Real-time signing policy enforcement requires an authority to assert the local-site signing policy

Assertion-Validation Authority

- Validation of assertions can be centralized:
  - Services and clients out source validation to central service
  - Trust root configuration needs to be maintained centrally
  - Ensures correct validation code and up to date trust root configuration is used
- Example: XKMS specification defines a service that allows a relying party to hand-over all received certificates to this trusted service, and to obtain the validated name or attribute bindings
- Centralized validation services have to be trusted by the clients and services
  - Done through the **Assertion-validation Authority**

New Collaboration => New Set of Trust-Roots

- Deploying clients/services of an organization requires configuring their trust root
- Cross-organizational collaborations require the involved entities to be provisioned with the trust-roots of all the participating organizations
- Newly included projects need to be provisioned with VO-specific trust-root information
- Layered configuration:
  - VOs leverage site specific configuration
  - Services within VO leverage VO configuration

Provisioning Issues

- Assertion validation can be **very** complex
  - X509 path-validation is tedious
  - Bridge CA with certificate discovery further complicates processing
- Validation code and trust-root configuration needs to be correct on each resource
  - Maintenance issues, especially light-weight clients
  - Out-of-date could imply security exposure
  - Administration of new trust-roots
  - Managing revocation

Centralized Assertion Validation

- Assertion Validation Service
  - Centralize complex processing and trust root configuration
  - Ensure correct validation
  - Ease burden on administrators
  - Existing standardized solutions: XKMS, SCVP
- Resources need to trust Centralized validation service
  - Configured through the **Assertion-validation Authority**

Certificate Validation Profile Support

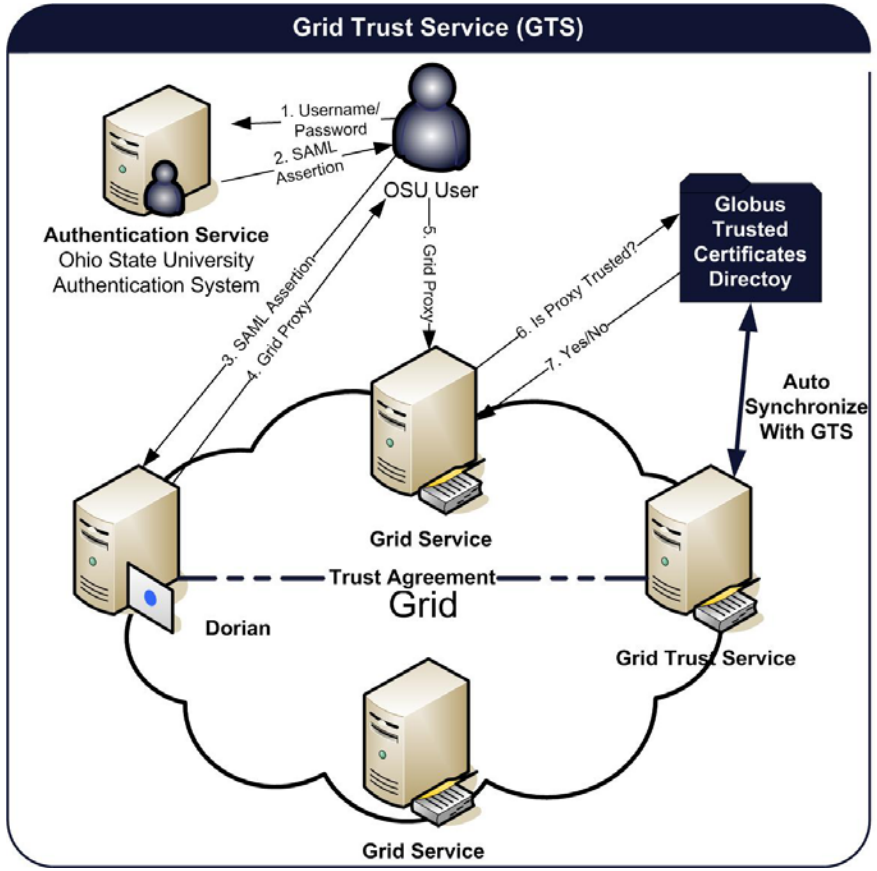
- **Locally Stored Locally Validated Profile (LSLV)**
  - Supported by Globus 4.0.3
  - Directory of Trusted Certificates
  - Certificate Validation against certificates in directory of Trusted Certificates

- **Remotely Retrieved Locally Validated Profile (RRLV)**
  - Use trust service to obtain trusted CA certificates and CRLS and store them in the Globus Trusted Certificate directory.
  - Trust Service client manages the Globus Trusted Certificate directory for Globus, keeping it up to date.
  - Only minor changes to Globus required.

- **Supporting Remotely Stored Remotely Validated Profile (RSRV)**
  - Globus contacts Trust Service during authentication to determine if the credentials in question are signed by a Trusted CA
  - Trust Service performs all validation and enforces revocation lists.
  - Support requires SIGNIFICANT changes to the Globus Toolkit

Grid Trust Service (GTS)

- **Grid Trust Service (GTS)**
  - WSRF Grid Service
  - Define and manage levels of assurance.
  - Provides Support for Managing Trusted Certificate Authorities
  - Administrator register/manage certificate authorities and CRLS with GTS
  - Client tools synchronize Globus Trust Framework with GTS
    - Remotely Retrieved Locally Validated Profile (RRLV)
    - Globus is authenticating against the current trust fabric
  - Distributed GTS, Enabling the creation of a scalable trust fabric.

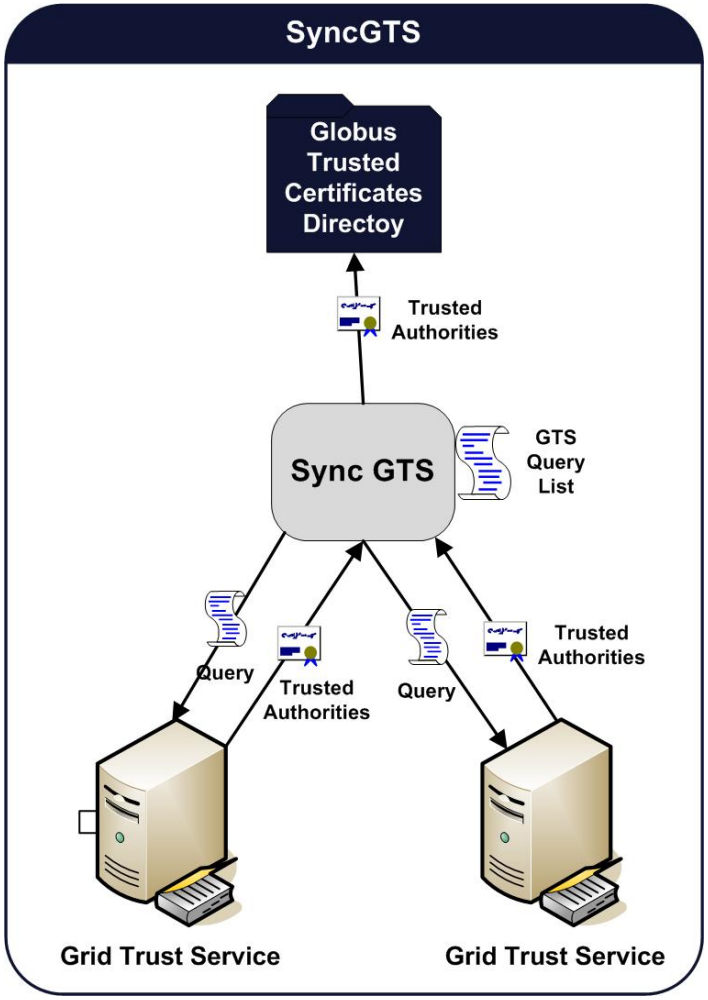


Grid Trust Service (GTS)

- **Trusted Authorities**
  - GTS manages a set of certificate authorities that are trusted in the grid to sign grid credentials.
  - **Trusted Authority –** A certificate authority trusted by the GTS.
    - Name (Subject of the CA Certificate)
    - Trust Level (s) – The level(s) of Trust associated with the CA.
    - Status – The current status of the CA (Trusted or Suspended)
    - Certificate – The ca certificate that corresponds to the private key that is used by the ca to sign certificates. (credentials).
    - Certificate Revocation List (CRL) – CA signed list of revoked credentials.
    - *Is Authority – Specifies whether or not the GTS listing this Trusted Authority is the authority for it.*
    - *Authority GTS – The authoritative GTS for the Trusted Authority*
    - *Source GTS – The GTS from where the current GTS obtained the Trusted Authority from.*
    - *Expiration – The date at which after this Trusted Authority should no longer be trusted.*

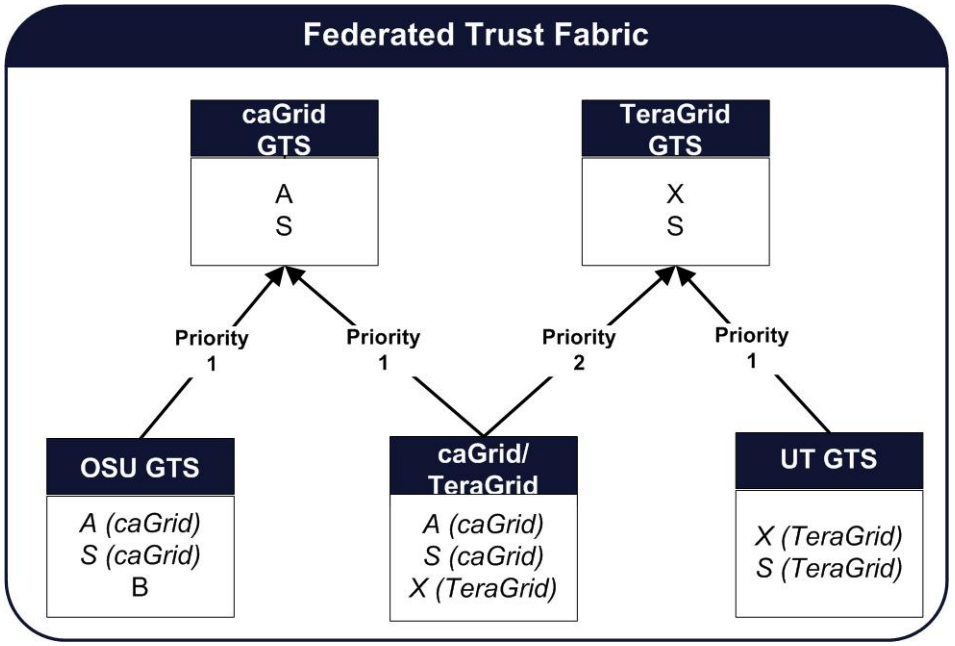
SyncGTS

- **Toolkit used for synchronizing client and service containers with the GTS**
- Takes a set of GTS Queries and executes them on a GTS, synchronizing the results of the queries with the Globus Trusted Certificates Directory.
- Supports multiple execution mechanisms.
  - Grid Service in a grid service container
  - Embedded in a client or service
  - Command Line



Grid Trust Service (GTS) Federation

- **GTS Federation**
  - A GTS can inherit Trusted Authorities and Trust Levels from other Grid Trust Services
  - Allows one to build a scalable Trust Fabric.
  - Allows institutions to stand up their own GTS, inheriting all the trusted authorities in the wider grid, yet being to add their own authorities that might not yet be trusted by the wider grid.
  - A GTS can also be used to join the trust fabrics of two or more grids.



OTP & Trust-Root Provisioning

Bootstrap User's Trust-Root Config from Secure OTP Authentication

Enhanced MyProxy/GridLogon Svc

Secure mutual OTP-Authentication and Key-Exchange

OTP AuthN Server + user's security config

Short-Lived Cert + Provisioning of CA's, AuthZ/Attr Authorities

OTP

user-workstation (initially not configured)

MyProxy and Grid Trust Service

- MyProxy
  (creds swiss-army knife)
  - (optionally) provisions clients with CA Certificates and CRLs
  - Only C-clients and no webservice protocol
- Grid Trust Service (GTS)
  - provisions clients with CA certificates and CRLs
  - Only Java-clients and webservice protocol
  - Hierarchical centralized admin model
- Functionality insufficient...
  but is on the right path forward

Status Quo

- Trust-Root provisioning is static or very limited
  - Clients and service configuration changes requires **real** effort
  - Every new collaboration requires **manual** provisioning of participating entities
  - Out-of-date, i.e. incorrect, configurations lead to security exposures
- Assertion revocation and signing policy validation is primitive or non-existing
  - Inability to validate signing-policy in real-time requires overly-strict CA-agreements
  - Out-of-date revocation information leads to security exposure
- Assertion validation not centralized
  - Complex validation code needs to be up-to-date on each client and service
  - Bridge CA deployment too complex for current middleware

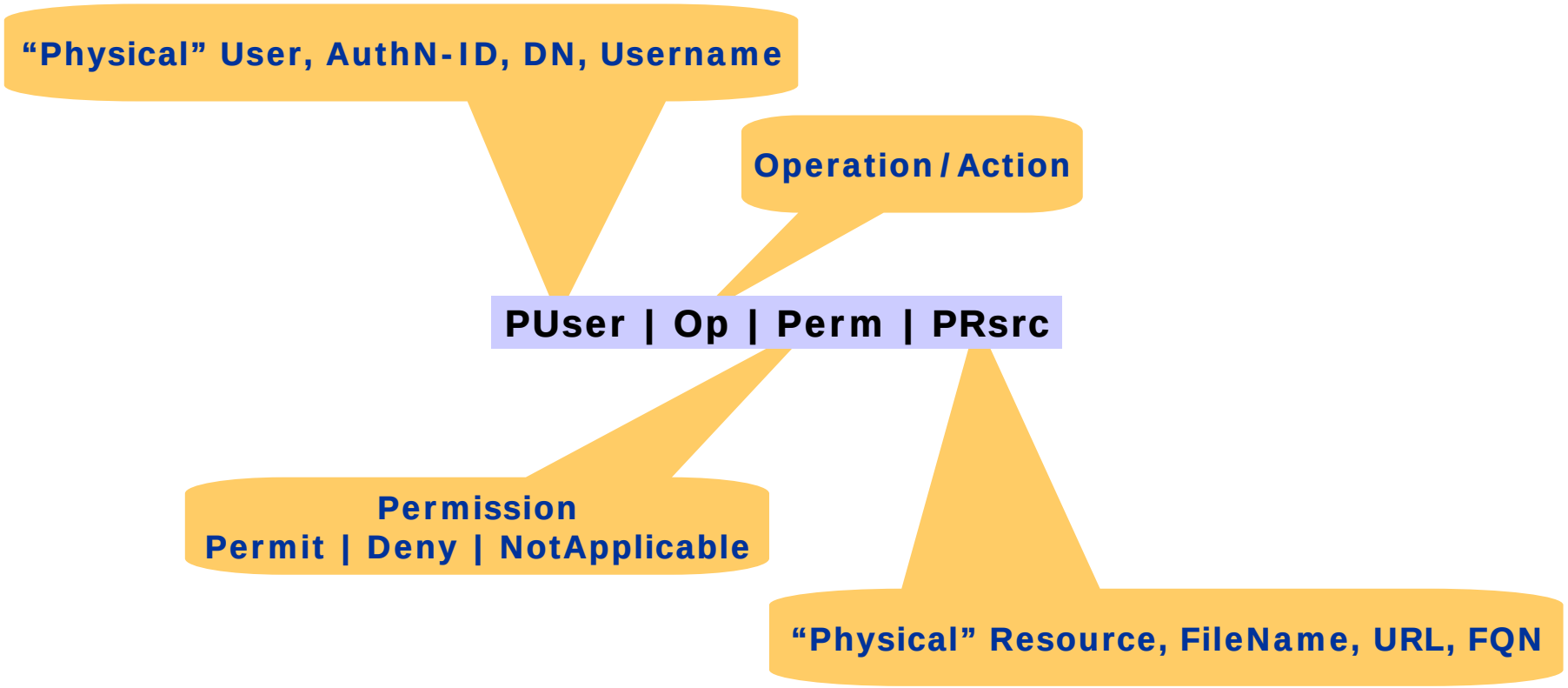
Path Forward

- Enhance MyProxy/GTS to provision **all** trust roots required for organization, VO, and/or collaboratory
  - Centralized admin of clients and services' security-configuration
- Enhance Grid-middleware to transparently
  - Enable real-time, dynamic configuration provisioning
  - Validate signing policy
  - Maintain client and service security-config up-to-date
- Centralize processing of complex validation
  - Enhance Grid-middleware to optionally deploy and leverage centralized assertion-validation services

Contents

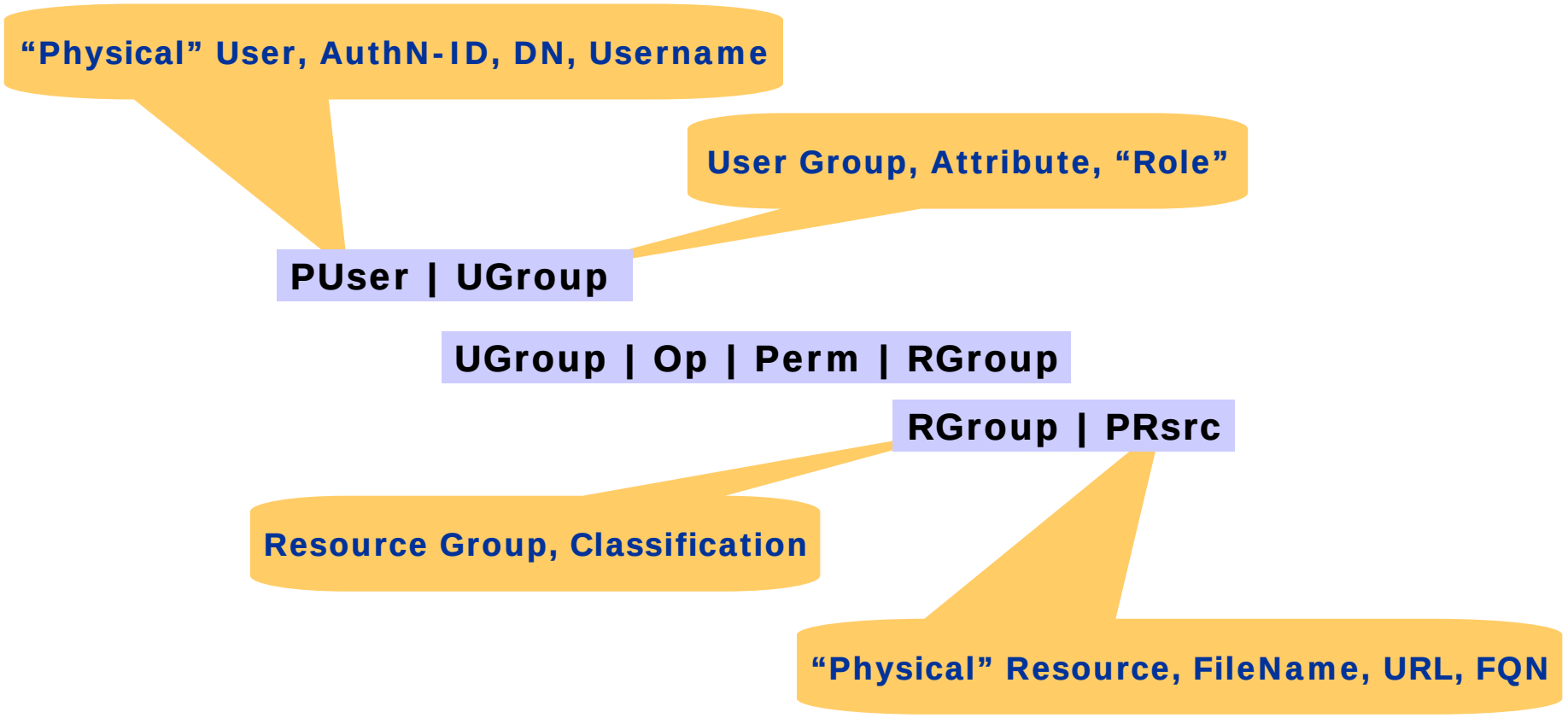
- Trust Provisioning & Validation
- **Policy Enforcement Complexity**
- Data Integrity Verification Facilities

Access Policy Taxonomy (1)



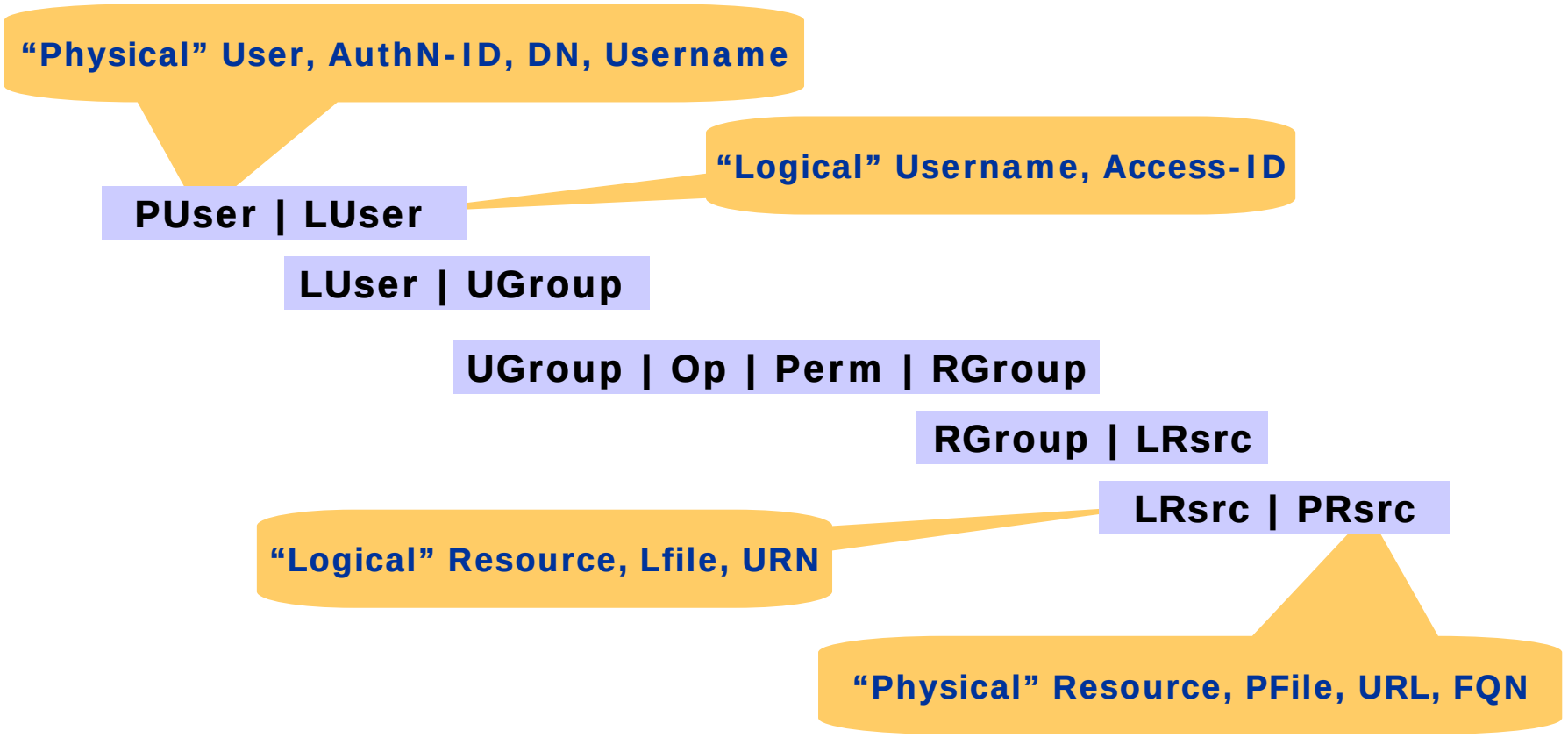
Identity-based, ACL-like, most simple policy statement

Access Policy Taxonomy (2)



Grouping Abstractions
policy (mostly) defined on groups

Access Policy Taxonomy (3)



"Logical" Abstractions
support multiple authN-mechs
resource location transparencies

Access Policy Taxonomy (4)

PUser | LUser

LUser | UGroup

Luser / UGroup | Role

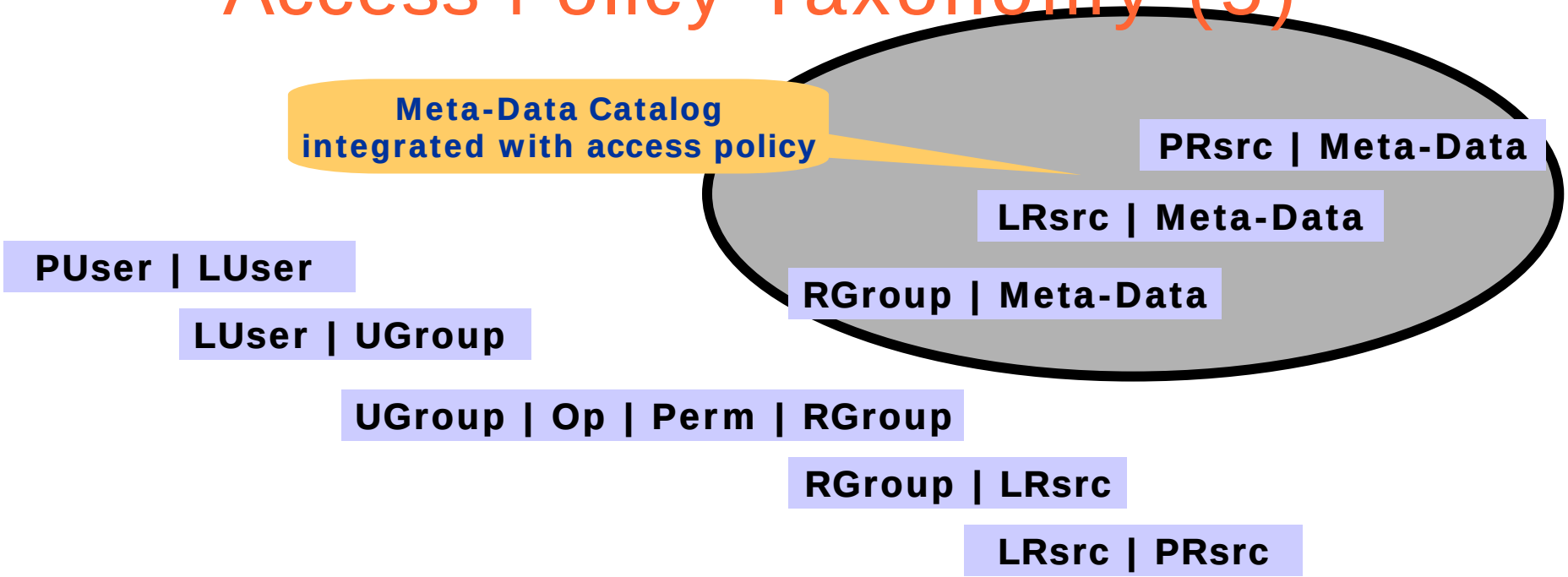
Puser / Luser / UGroup / Role | Op | Perm | Rgroup / LRsrc / PRsrc

RGroup | LRsrc

LRsrc | PRsrc

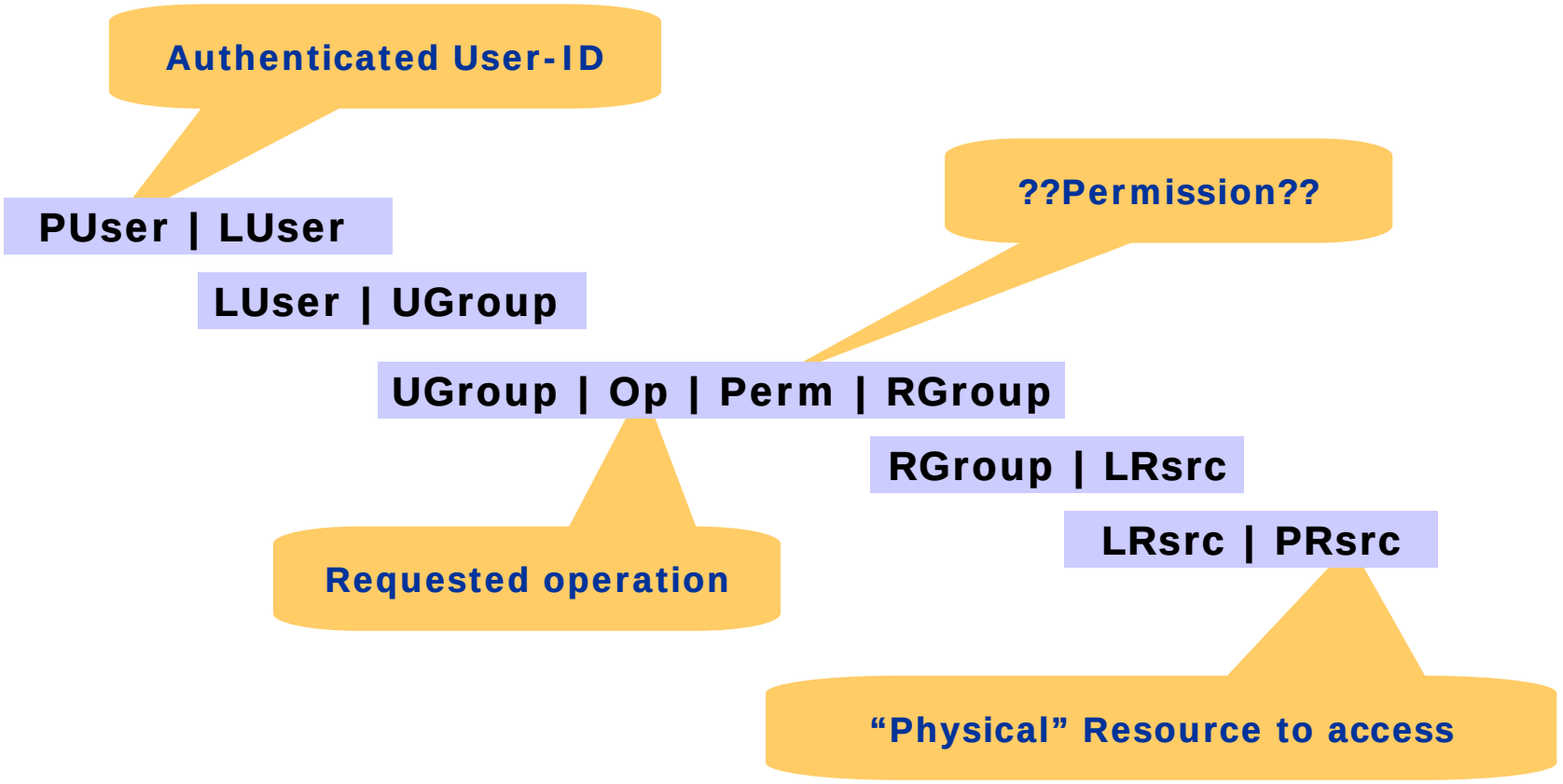
Policy on physical, logical, roles and groups
…plus hierarchical groups / roles, etc., etc…

Access Policy Taxonomy (5)



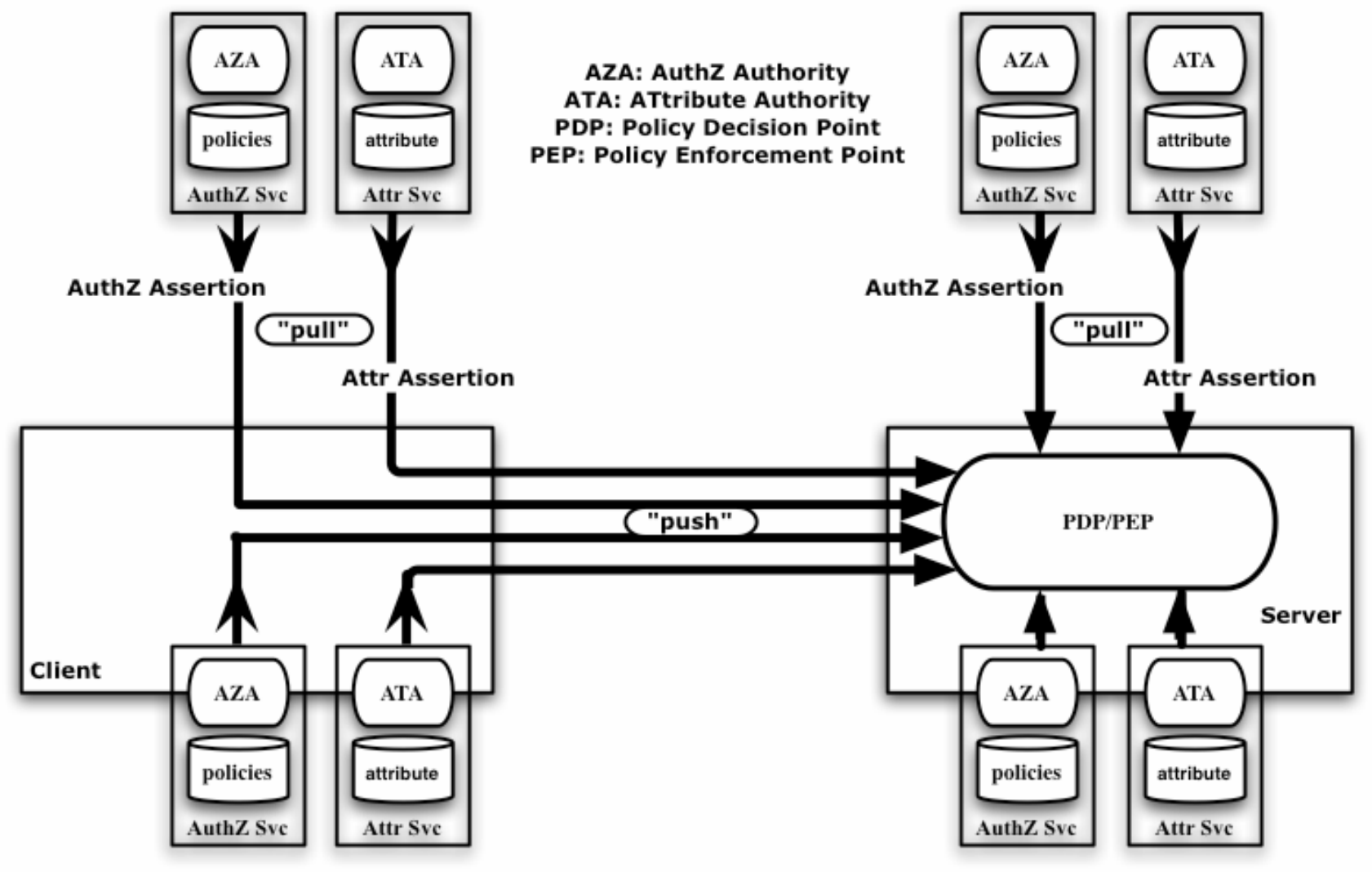
Meta-Data Catalog Integration
allows for "secure-browsing"

Access Determination (1)

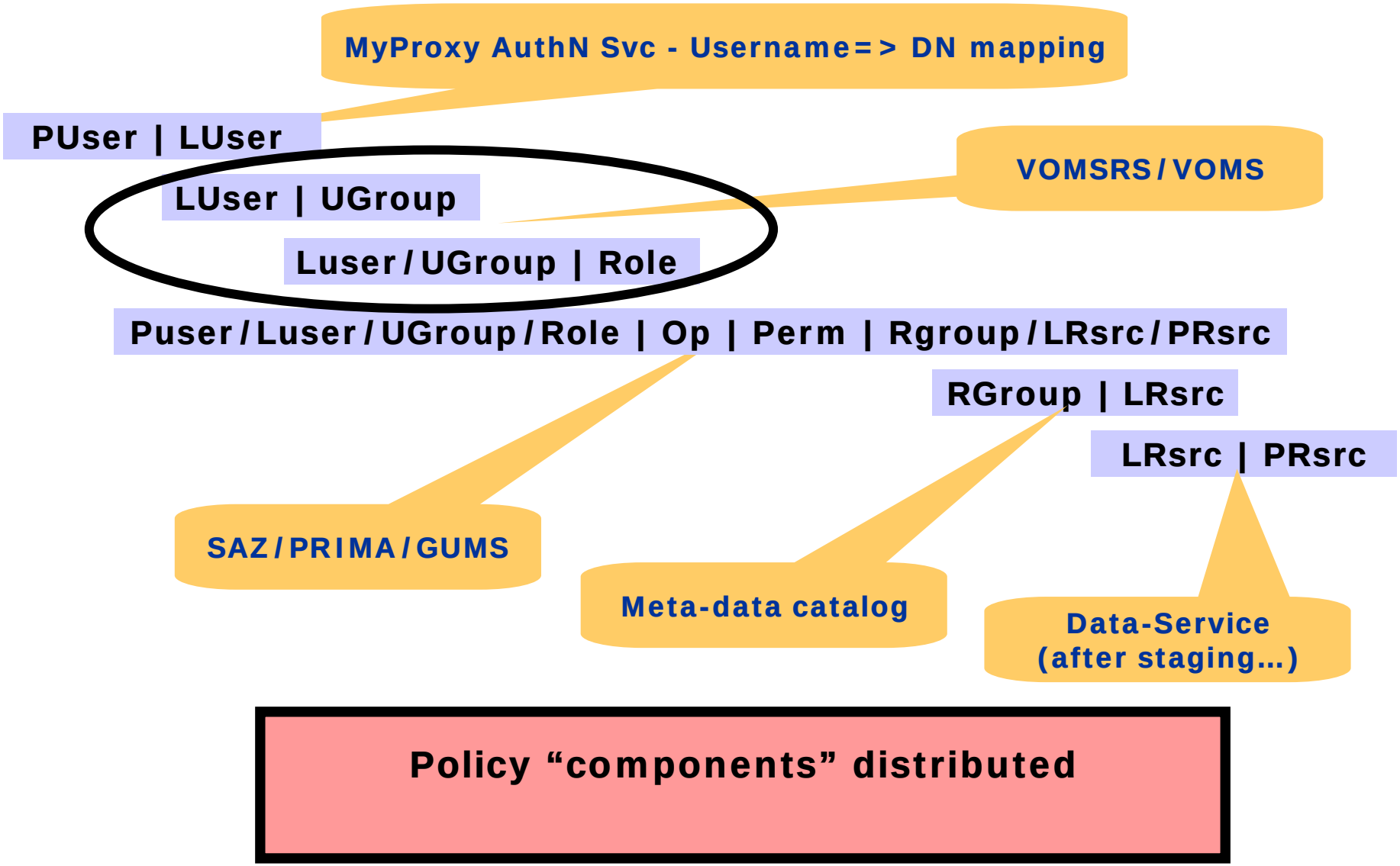


Can Subject invoke Operation on Resource?
Can AuthN-ID invoke Operation on Physical-Resource?

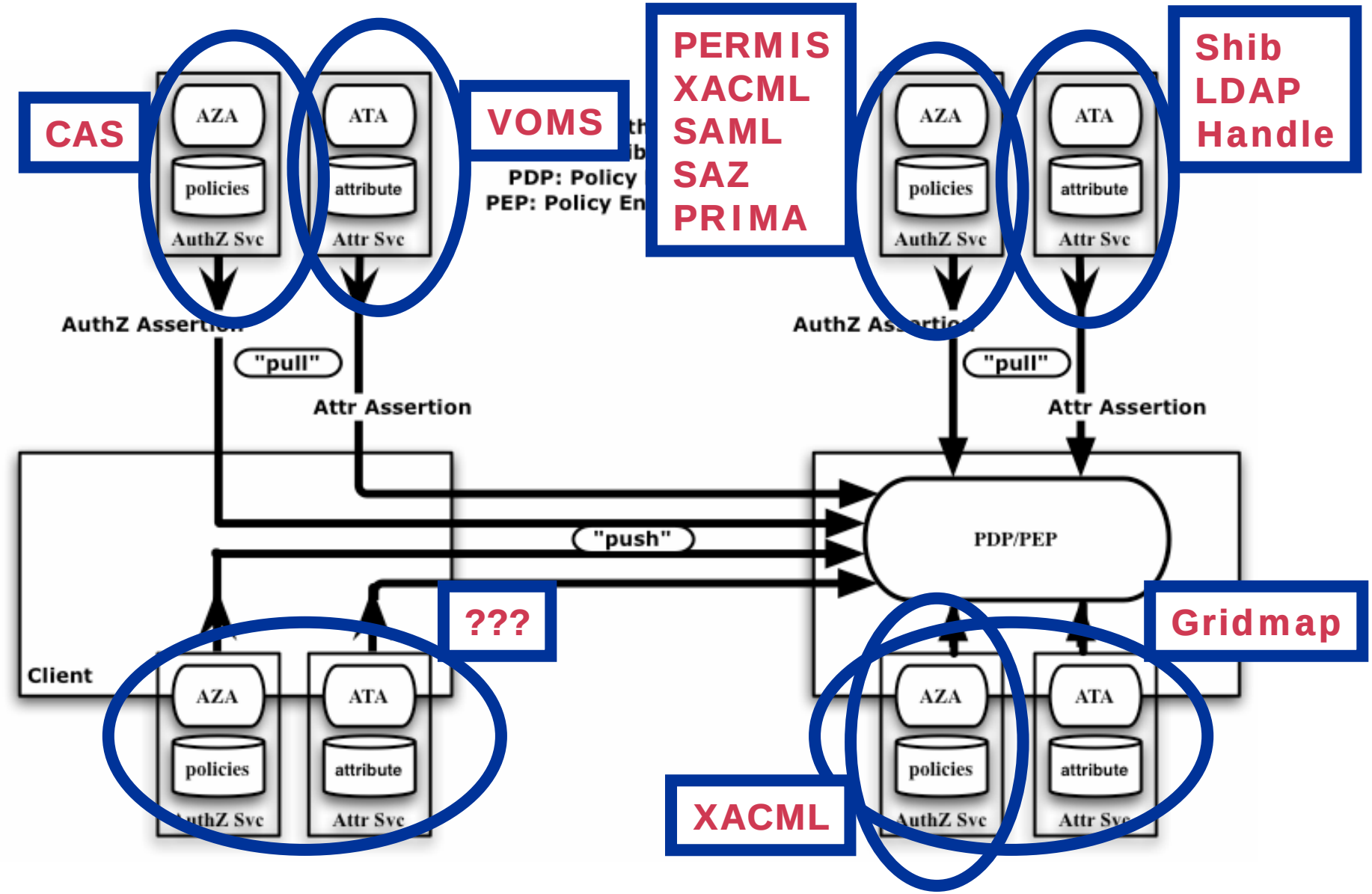
Policy Assertions from Everywhere



Access Determination (2)



Policy Assertions from Everywhere



Policy Evaluation Complexity

- Single Domain & Centralized Policy Database/Service
  - Meta-Data Groups/Roles membership maintained with Rules
  - Only Pull/push of AuthZ-assertions
- ...

- **Challenge is to find right "balance"**
- **(driven by use cases…not by fad/fashion ;-))**

- ...
- Split Policy & Distribute Everything
  - Separate DBs for meta-data, rules & attribute mappings
  - Deploy MyProxy, LDAP,VOMS, Shib, CAS, PRIMA, XACML, PRIMA, GUMS, PERMIS, ???

AuthZ & Attr Svcs Topology

- Policy Enforcement Use Cases determine “optimal” AuthZ & Attr Svc Topology
- Client pull-push versus Server pull
  - Network-hurdles/firewalls
  - Crossing of admin domains
- Separate Attributes from Rules (VOMS/Shib)
  or
  Separate Policies from Enforcement Point (CAS)
  - Separation of duty - delegation of admin
- Replicating of Policy-DB or Call-Out
  - Network overhead versus sync-mgmt overhead

- **!!! Choose “Most Simple” Deployment Option !!!**
  (ideally, services and middleware should allow all options…)

Contents

- Trust Provisioning & Validation
- Policy Enforcement Complexity
- **Data Integrity Verification Facilities**

Data Integrity Protection

- Data “Corruption”
  - Many, many copies of the original data files and model-code
  - Many “opportunities” for undetected changes
    - Independent from normal integrity protection for storage and data moving
  - Accidental, script-kiddies or worse…
- Integrity Protection
  - Identify and guard the “original”
    - Most files are immutable…maybe make them all immutable…
  - Use file-signatures/digests (SH-1/256, ???)
    - Tripwire-like change detection
  - Digest part of meta-data, communicate expected digest with URL/URI, independent digest-services, embed digest in URI, use digest-value as “natural” name for file…**file-name=digest-value**
    - Learn from file-sharing P2P application!
  - Integrate integrity checks in file-moving apps
    - http, DataMoverLight, GridFTP, Opendap, RLS, etc.
  - Define procedures for data corruption detection

Conclusion

- Need for centralized managed configuration management of security trust-roots and related information
  - Build on MyProxy/GTS efforts…
- Need for Creds/Assertion Validation Services
  - XKMS/CVS?
- Need for Data Integrity Facilities
  - No available solutions ?