Trusted Data Storage for Grid-based Medical Applications

Guido van 't Noordende
Silvia Olabarriaga, Matthijs Koot, Cees de Laat
University of Amsterdam
The Netherlands
guido@science.uva.nl

Overview

- Goal
- Existing approaches
- Technical problems
- Legal aspects
- Technical solution
- Conclusions

Goal

- Make it possible to run medical applications that handle privacy sensitive (medical) data on the Grid.
- Data-centric & collaborative focus:
  - Medical researchers place data on storage system
  - (Other) researchers can access the data for analysis
  - Fine-grained access control
  - Dissemination control

Examples

- Medical Imaging
  - e.g., 2D/3D image analysis / statistics, e.g., MRI
- Research
  - (retrospective) studies on collections of data
  - Collaborative research between hospitals
- 'urgent' computing with deadline for results
  - Blood flow analysis (e.g., surgical planning)
- Computational power in hospitals insufficient for some (relevant) tasks

Issues

- Trust issue: data stored outside hospital
- Security of the storage system
  - Mostly solved, e.g., MDM, encrypted storage with Hydra key-store
- Who administers a key store / storage system?
- Assuming storage system is trusted:
  - Who may access the data, and from which machines / administrative domains?
- Any (Grid) machine may be cause of information leakage

Legal Aspects

- European law (EU):
- General privacy regulations
  - Purpose binding
  - Necessity / minimality / transparency
  - Consent for medical research data: purpose bound
- Medical data regulations
  - Physician legally responsible for ensuring an appropriate level of security to protect data
- Similar laws / regulations in non-EU countries
  - e.g., U.S. HIPAA, PIPEDA, ...

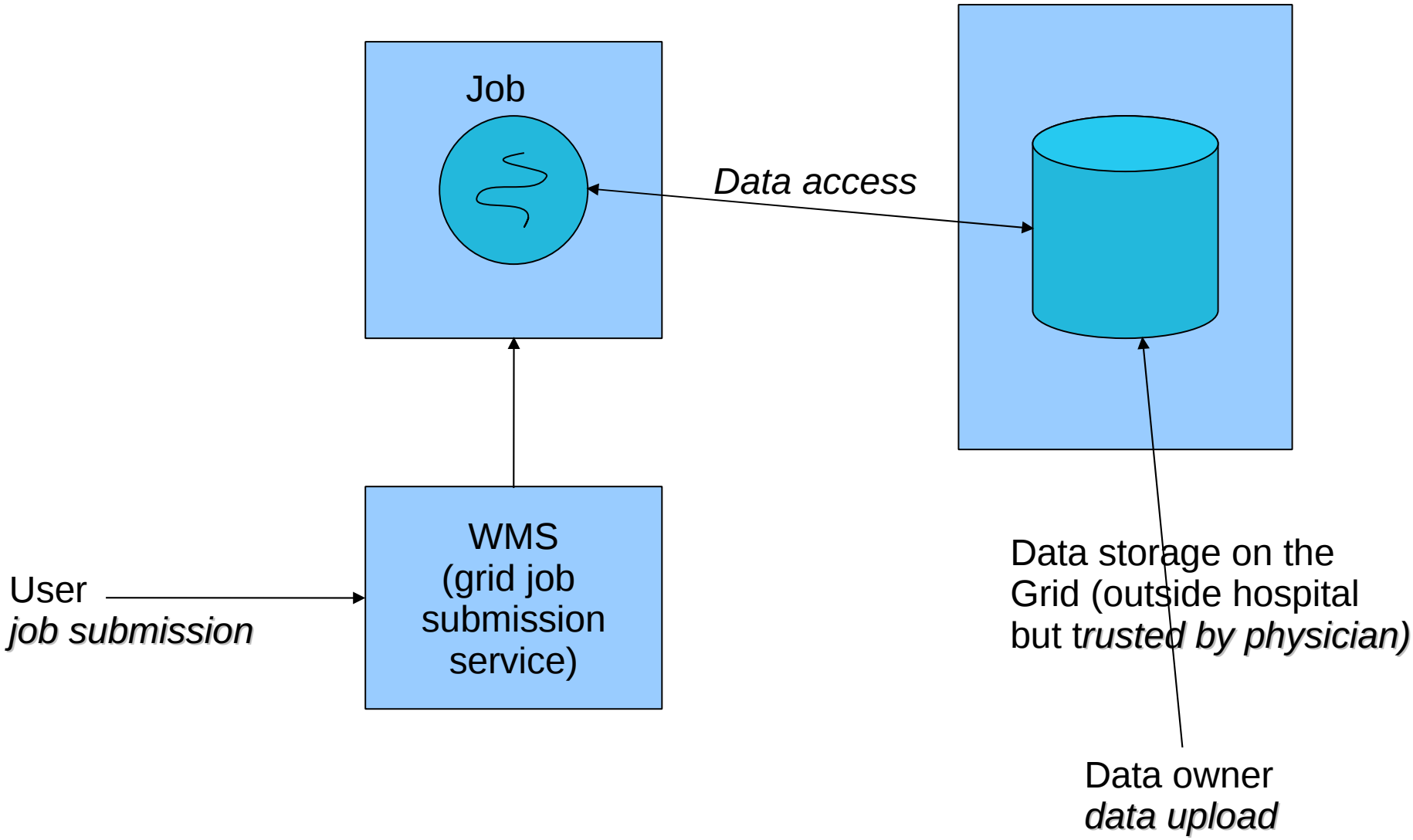
Practical Issues

- Many hospitals deploy 'perimeter defence'
- no export of patient / identifyable patient data beyond hospital walls
- Export data outside hospital permitted for specific research purpose only, if de-identified
- De-identification such as DICOM header blanking often used but dangerous
- E.g., unmodified MRI data contains facial features so a patient's face may be reconstructed

Lessons

- Strict legal constraints on what data may be used by whom and for what purpose
- Physician / hospital is responsible for data protection, hard to get data out
- 'risk assessments' actually take place (often implicitly)
- De-identification is hard to get right
- Even de-identified medical research data with consent should be treated as sensitive data!

Existing (Data) Grid Middleware



NOTE: all machines may be in different administrative domains!

GSI Proxy certificate weakness

- GSI weakness (not new)
- job – proxy certificate binding issue
- Job's code can be replaced by malicious code
- Solution: secure job container
  - sign {code + proxy cert + other data} using job owner's private key
  - must trust that Grid middleware on remote host where job runs verifies the job container

Requirements / Trust Aspects

- From data owner's perspective, ensure:
  - Trust in administration + implementation of the 'trusted' data storage system
    - Sufficient authorization capabilities + safe storage
  - Trustworthiness of the system from which a job accesses the data:
    - Trust job authentication mechanism
    - Administrator (will not purposefully leak information)
    - Software installation, version, configuration
    - Deployment (sufficient protection against intruders?)
    - 'standard' unix (e.g., Linux) distributions trustworthy?
    - Risk assessment!

Requirements

- Data owner / hospital require:
  - *Who* can access data (user, ACL)
  - Trust in storage system + administrator
  - Control from which *hosts,* with what *properties,* in what *admin. domains,* can jobs access the data
- We must know a lot about a cluster node before we allow data to go there:
  - Administrator?
  - Implements proxy-job binding verification?
  - Software configuration? OS version? Encrypted swap space?

Technical Solution: Trusted SRB

- Enforces data owner access control policies
- Per data item (or set):
  - User/role-based ACL
  - Host ACL
    - specifies trusted administrative domains, e.g., with pre-agreed non-disclosure contracts.
  - Remote Host Property List (HPL)
    - Contains required host OS + configuration details
    - e.g., non-shared /tmp directory, file deletion on exit, etc.
    - OS / version, middleware type/version, etc.

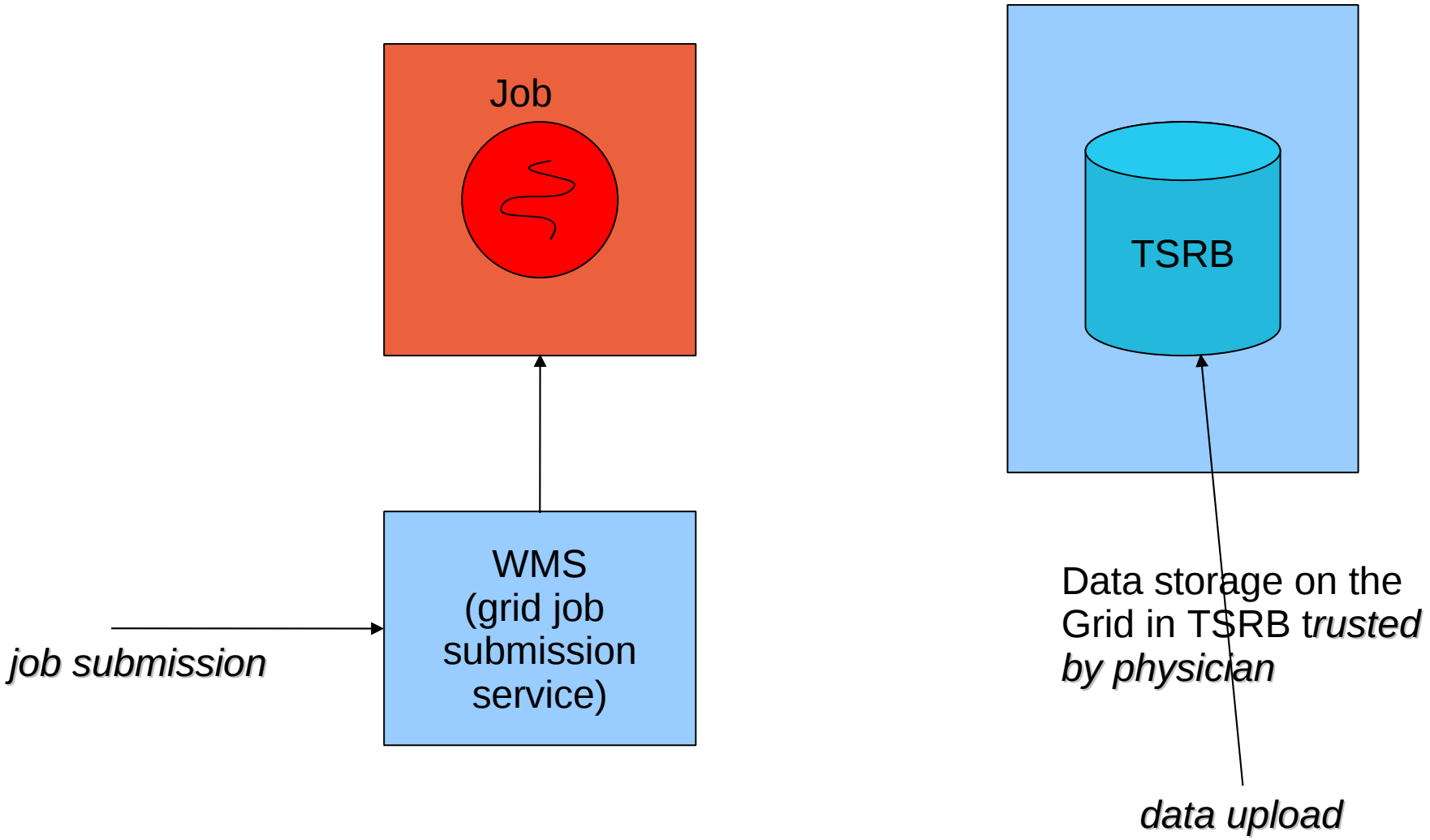
TSRB Overview

- Users define User ACL, Host ACL and RHPL
- TSRB enforces the above
- HPL allows for flexible management and identification of hosts with different properties (dynamic) within an administrative domain
- Client side (e.g., gridFTP):
  - Channel setup using regular GSI / proxy certificate (for user authentication)
  - *Microcontract* required for each transaction

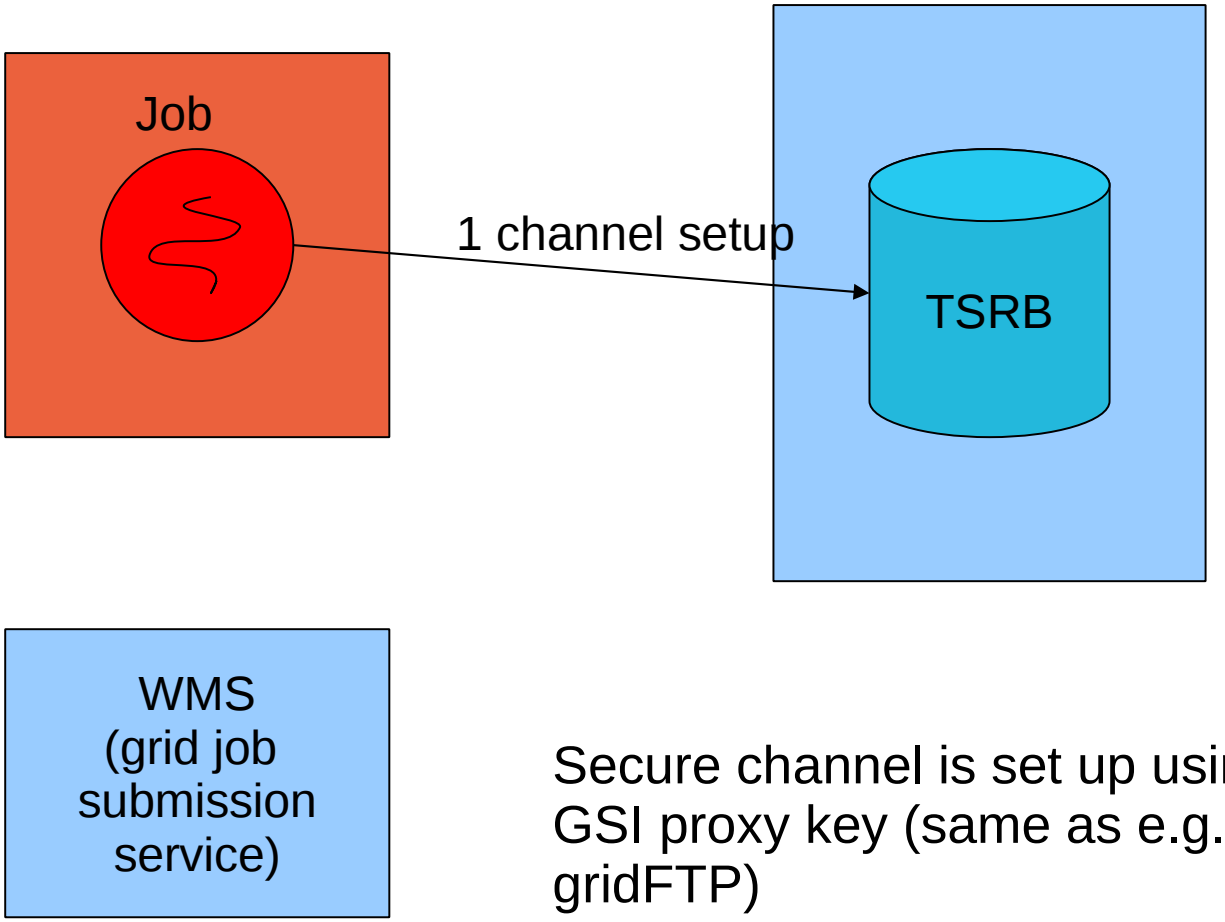
Microcontracts

- A *microcontract* is a fine-grained, per-transaction agreement which binds user identity, host identity and HPL to each transaction
- Microcontracts contain:
  - Host property list (HPL) of job's host
  - Name of requested file + operation
  - (Hash of) user's public key / proxy certificate
  - Microcontract signed using host key
- Microcontracts allow auditing all transactions

Walkthrough (1)

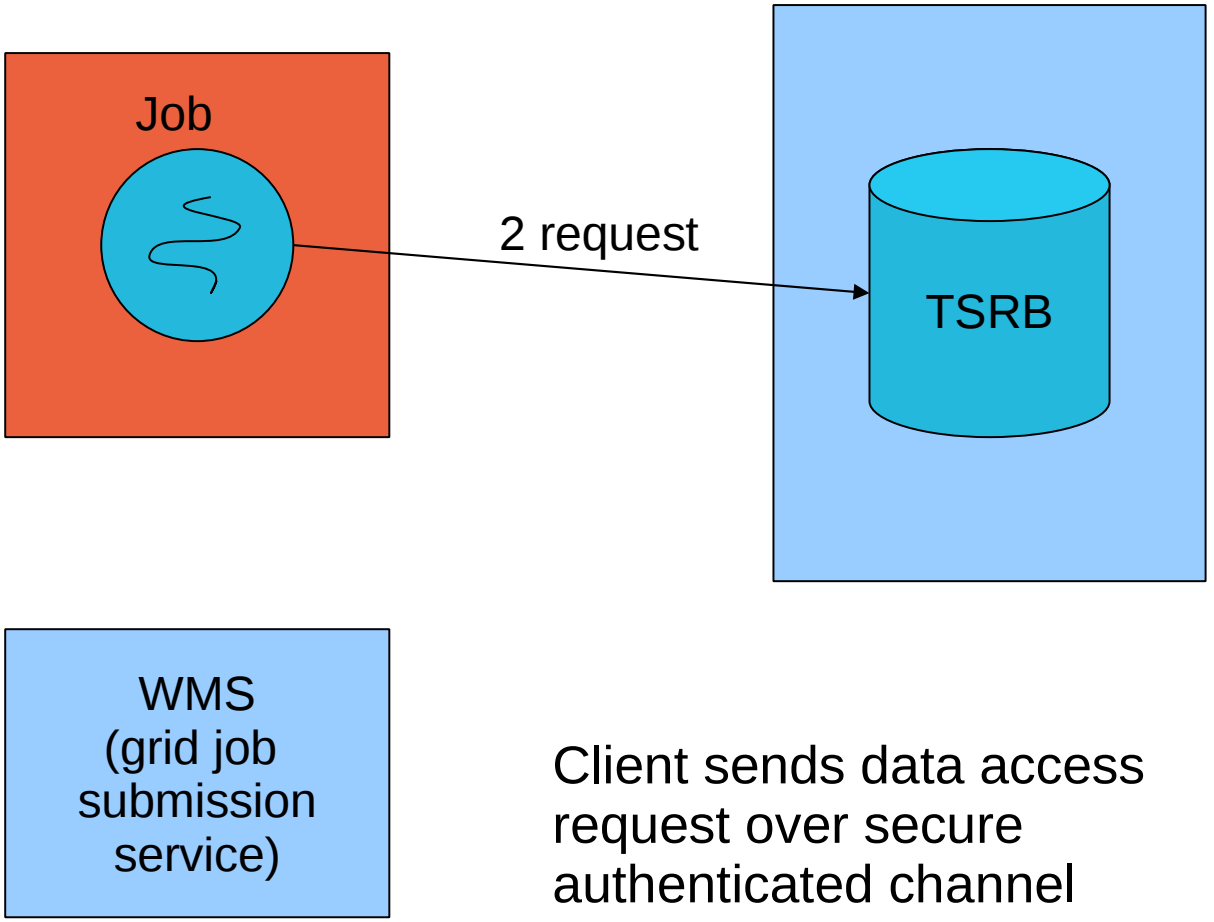


Walkthrough (2)



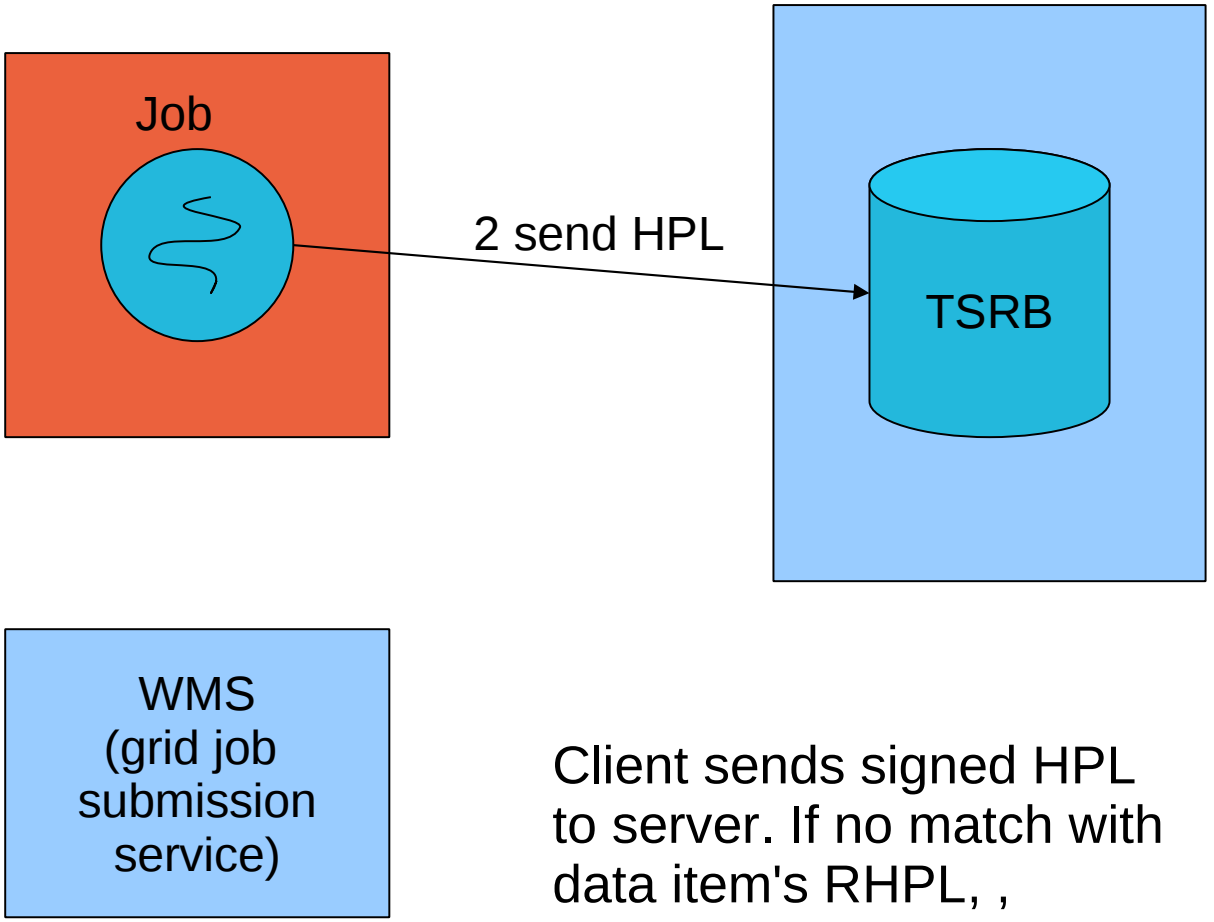
Secure channel is set up using GSI proxy key (same as e.g., in gridFTP)

Walkthrough (3)



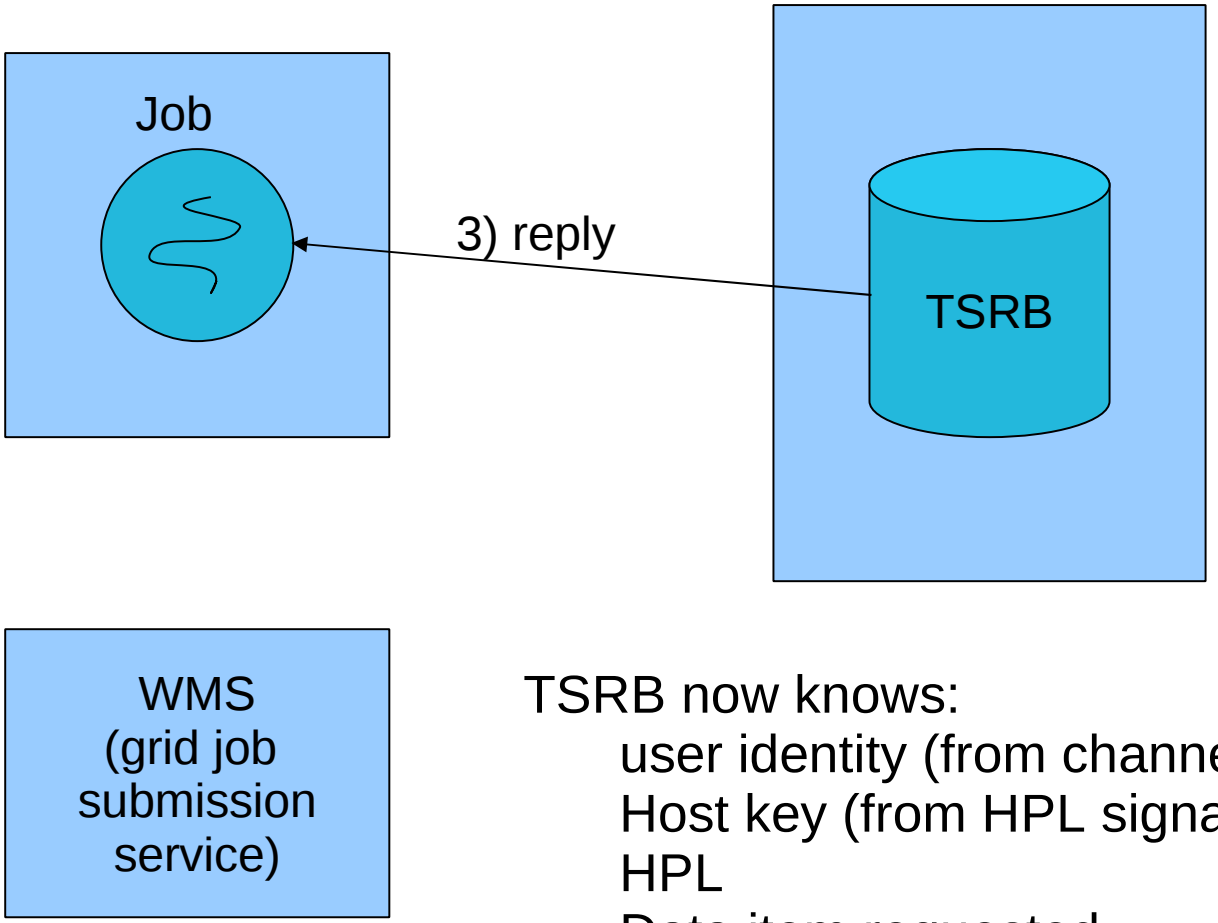
Client sends data access request over secure authenticated channel
Error returned if user not in User ACL

Walkthrough (4)



Client sends signed HPL
to server. If no match with
data item's RHPL, ,
an error is returned.

Walkthrough (5)

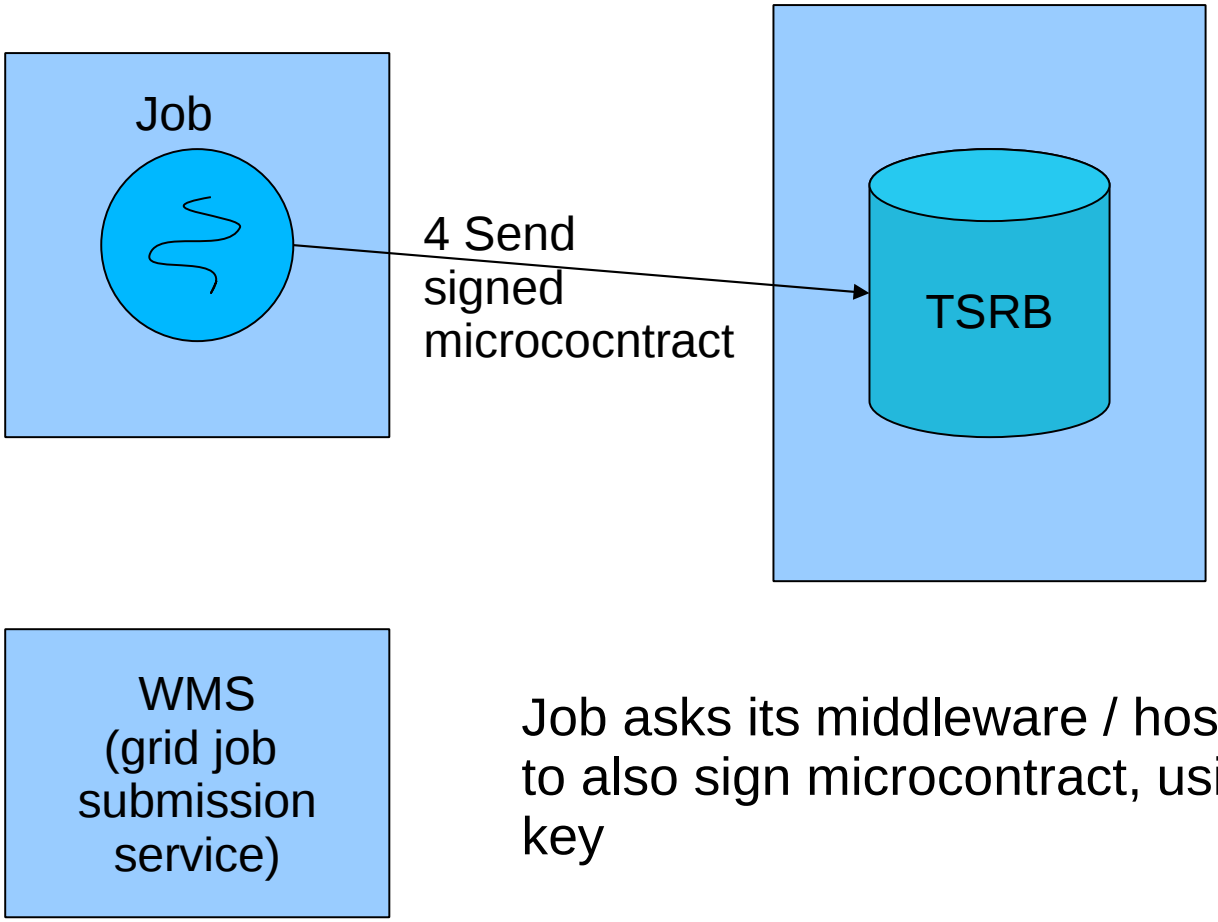


TSRB now knows:

- user identity (from channel setup)
- Host key (from HPL signature)
- HPL
- Data item requested

If all checks out, a microcontract
is prepared and signed by the TSRB
and sent back to the job.

Walkthrough (6)

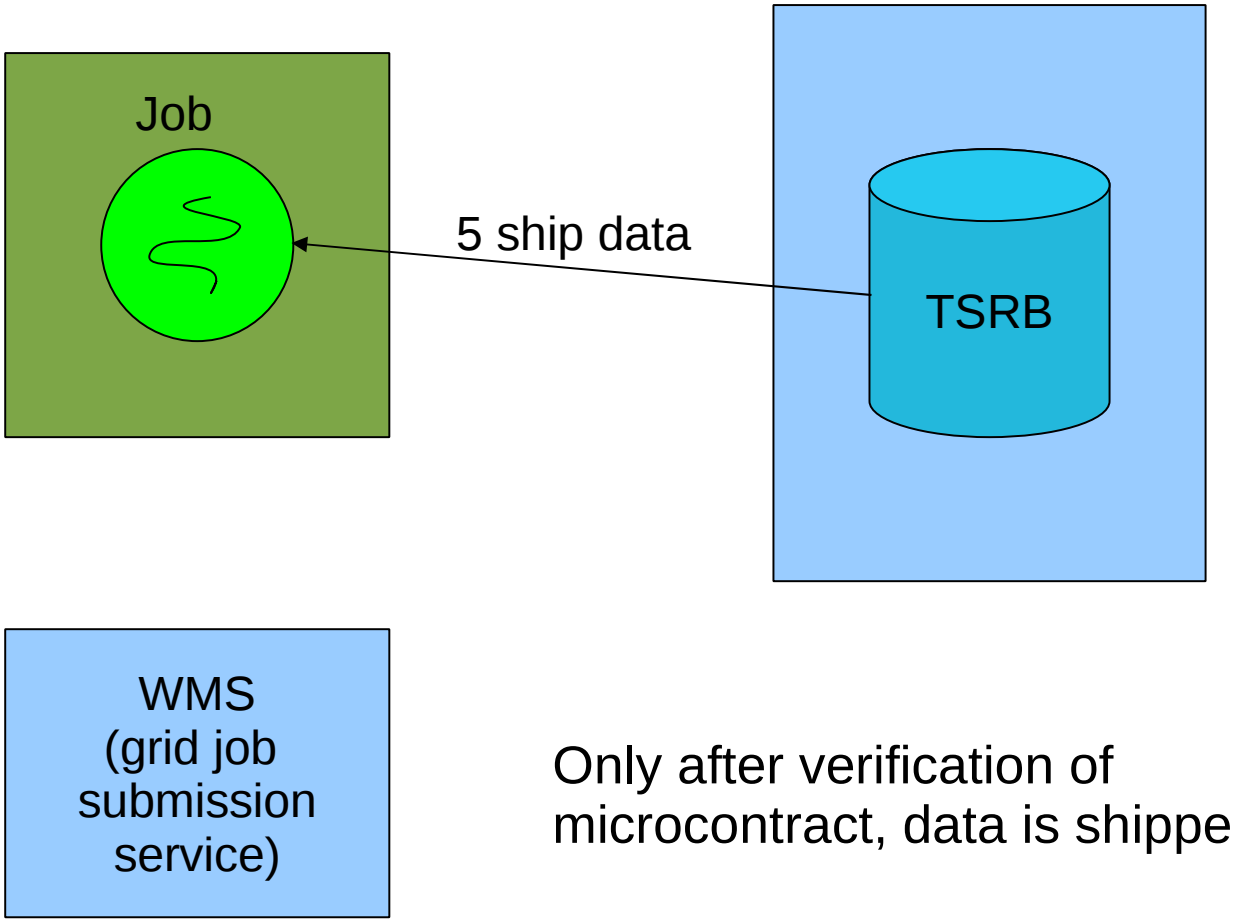


Job asks its middleware / host
to also sign microcontract, using host key

Microcontract signature securely binds all information required for verification

Only now is all information securely verifyable + auditable

Walkthrough (7)

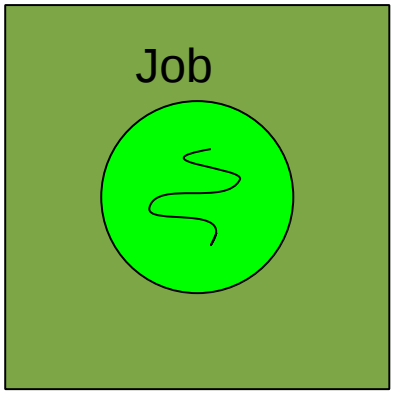


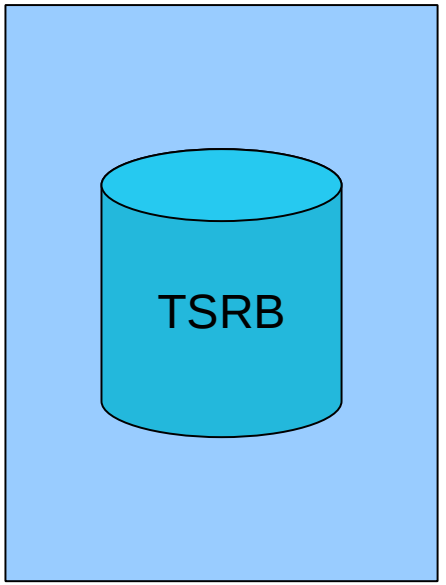
Only after verification of microcontract, data is shipped.

Microcontract is stored for auditing,

microcontract can be shipped to a separate (trusted) auditor process.

Walkthrough (8)





WMS
(grid job
submission
service)

After job exit, data (e.g., in /tmp/gridmap-user)
is deleted by middleware, subject to agreement in the microcontract

(file deletion should be mandatory property in HPL in my opinion).

Summary

- Trusted SRB trusted to store data safely
- Trusted SRB trusted to enforce policies:
  - Host ACL + Host Property List verification
  - User ACL (regular access rights)
- Trusted SRB has microcontract-based protocol
  - Securely binds job identity, HPL and host key (administrator identity) to allow verification of policy
  - Allows for auditing of each transaction

Discussion

- HPL contains information on job authentication mechanism, operating system, middleware and configuration details required to evaluate policy
  - HPL contains job authentication, operating system, middleware, and configuration details
  - e.g., data handling after exit, job container verification capabilities, etc.
- Microcontract-based data transactions
- TSRB allows for secure enforcement of trust-based access control and distribution policies resulting from data owner's risk assessment

Thank you

- Questions?