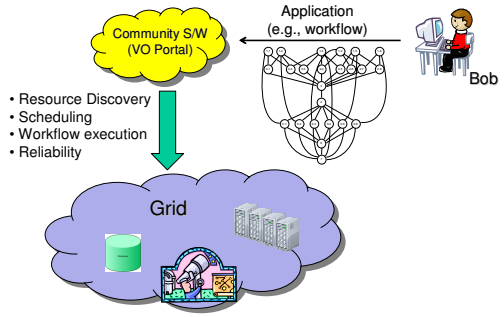
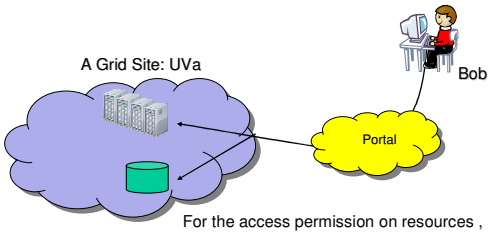
Automated, Least privilege Grid Delegation

Presented by Sang-Min Park

Grid / E-science Typical Scenario



Grid / E-science Typical Scenario



For the access permission on resources , request must pass two policies:
1)Resource's policy: "*Does UVa allow Bob to run job on cluster?*"
2)Bob's delegation: "Is Bob ok if Portal delete his file?"

State-of-art of Grid Delegation

- **Impersonating Delegation**
  - "Bob" generates X509 Proxy certificate with time limit (e.g., 1 day)
  - "Portal" uses Bob's proxy certificate when making request
  - This means:
    - Portal can do whatever Bob can do on Grid until certificate expires
    - Portal generally run jobs from many users (e.g., all astronomers in nation)
    - What happens if Portal is compromised?...It's **disaster!**
  - Grid community REALLY concerns about this problem
- **(Too) Many Policy Languages**
  - Proxy certificate standard allows policy be embedded into an extension field
  - So delegation is no more a problem "if policy describes delegation well"
  - So problem solved because user will write their policy with pleasure?

Example Policy (in SecPal)

(People will hate these things!)

What's ideal?

1. **Delegation should be (close to) least privilege**
2. **User should not be demanded to write policy**
   - They will never do that
   - They will make lots of errors

My Research Goal

Let's create least privilege delegation automatically!

How that's possible?

- In Grid application life cycle, user's already have application's description before execution
  (They accept the fact they at least have to describe their app.)
- This application description implicitly says what are the necessary privileges to run it

```
<job>
    <executable>/usr/bin/echo</executable>
    <directory>/home/scientist </directory>
    <cpu> 8 </cpu>
    <inputFile> http://fabrikam.com/file </inputFile>
    <argument>"Eat and sleep well"</argument>
    <stdout>0.stdout</stdout>
    <stderr>0.stderr</stderr>
</job>
```

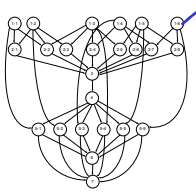
- So the remaining part is "**to extract the delegation policy from the application description**"

SecPal as an underlying policy language

- **SecPal**
  - Logic-based policy language from Microsoft Research
  - Based on formal model and proof is available for policy's property
  - It can be used to
    - Establish trust between entities
      "Fabrikam.com says Alice can possesses emailAddress=alice@fabrikam.com
      "Bob says Fabrikam.com can say %x can possesses emailAddress=@fabrikam.com"*
    - Grant permission
      "Bob says %x can read "Bob's file" if %x possess emailAddress=@fabrikam.com"*
    - Delegate restricted rights
      "Bob says Alice can say %x read "Bob's file" if x possesses emailAddress="@ fabrikam.com"*
  - Currently, evaluation engine and SDK for writing policy is available in .NET

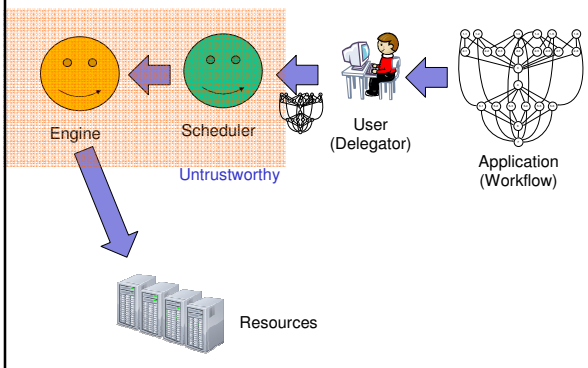
Grid Model and Entities

- **Application**
  - Assume workflow is our application
  - A node of graph (typically DAG) represents single job
  - An edge represents data dependencies between jobs
  - Workflow has become almost a de facto standard way of writing Grid Application

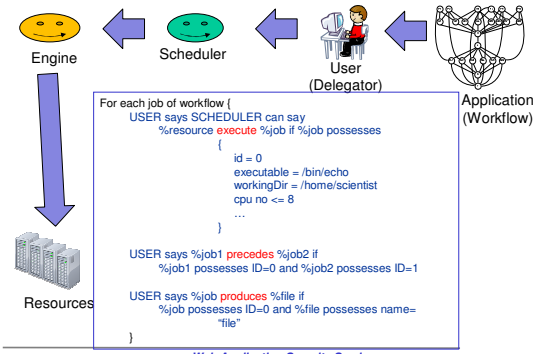


```
<job>
    <executable>/usr/bin/echo</executable>
    <directory>/home/scientist </directory>
    <cpu> 8 </cpu>
    <inputFile> http://fabrikam.com/file </inputFile>
    <argument>"Eat and sleep well"</argument>
    <stdout>0.stdout</stdout>
    <stderr>0.stderr</stderr>
</job>
```

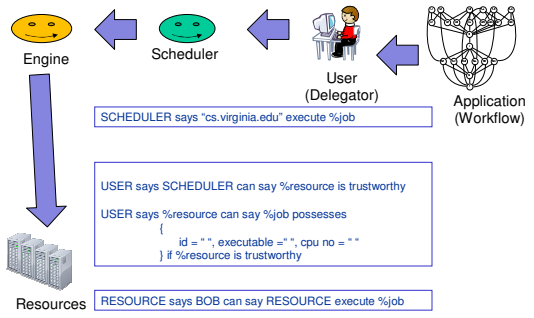

Grid Model and Entities



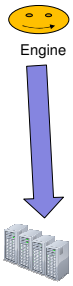
Translation Rule 1: Assert job's attribute



Translation Rule 2: Job's Execution



Translation Rule 2: Job's Execution



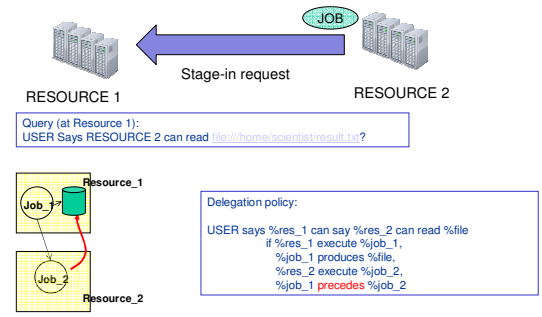
When Engine request job execution at RESOURCE

From the job request message, generate this token
RESOURCE Says %job possesses
{
executable = "/bin/echo",
cpu no = 9
…
}
Resource's query:
RESOURCE execute %job ?

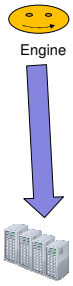
SecPal engine at RESOURCE returns permit decision only if
1)BOB said JOB can possess such attributes
2)SCHEDULER said RESOURCE is where JOB should run

If Scheduler is trustworthy, correct job will be executed on correct resource
If Scheduler is not trustworthy, correct job will be executed on somewhere

Translation Rule 3: File Access



Translation Rule 4: Job's execution order



Job should be executed in an order specified in workflow

USER says %res can execute %job_2
if %job_1 has finished,
%job_1 precedes job_2

USER says %res can say %job has finished
If %res execute %job

Resource needs to assert on job's successful completion

RESOURCE says %job_1 has finished

Evaluation

- **Implementation**
  - Implemented delegation translation rule on SecPal.NET
  - Grid entities (scheduler, engine, resource) are simulated within the .NET implementation
- **Evaluation Method**
  - Use Case Study (qualitative evaluation)
  - Performance (quantitative evaluation)
    - Does this matter? Yes.
    - Workflow consists of 1000s of jobs
    - Each job will generate few policy entries
    - Preliminary result show SecPal query evaluation is NOT fast (few seconds for simple policy)
    - Still working on…

Future Work

- **Implementation on Real Grid**
  - Currently implementation is proof of concept on .NET/Laptop
  - How can we integrate the mechanism with the real, production Grid software stack? How can we carry the policy statement?.. Needs to convince resource's additional overhead for SecPal-based authorization is minimal.
- **Policy size matters**
  - Typical workflow will generates too many policy entries
  - Size can be a burden on medium carrying the policy
  - Query evaluation can take too long
  - Are there ways to reduce the policy size?

Questions?