Universitatea “*Politehnica*“ din Bucuresti
Facultatea de Automatica si Calculatoare

Malicious client detection in BlobSeer

Catalin Leordeanu
Politehnica University of Bucharest
catalin.leordeanu@cs.pub.ro

Outline

- Introduction
- Malicious client detection
- Policy Enforcement
- Trust Management
- Conclusions and future work

Introduction

- **This work was done as part of the KerData-PUB associated team project.**

Coordinated projects:

- **BlobSeer Policy Enforcement** – Cristina Basescu
  (Master internship, Rennes)

Coordinators: Alexandra Carpen-Amarie (INRIA)
Catalin Leordeanu (PUB)
Alexandru Costan (PUB)

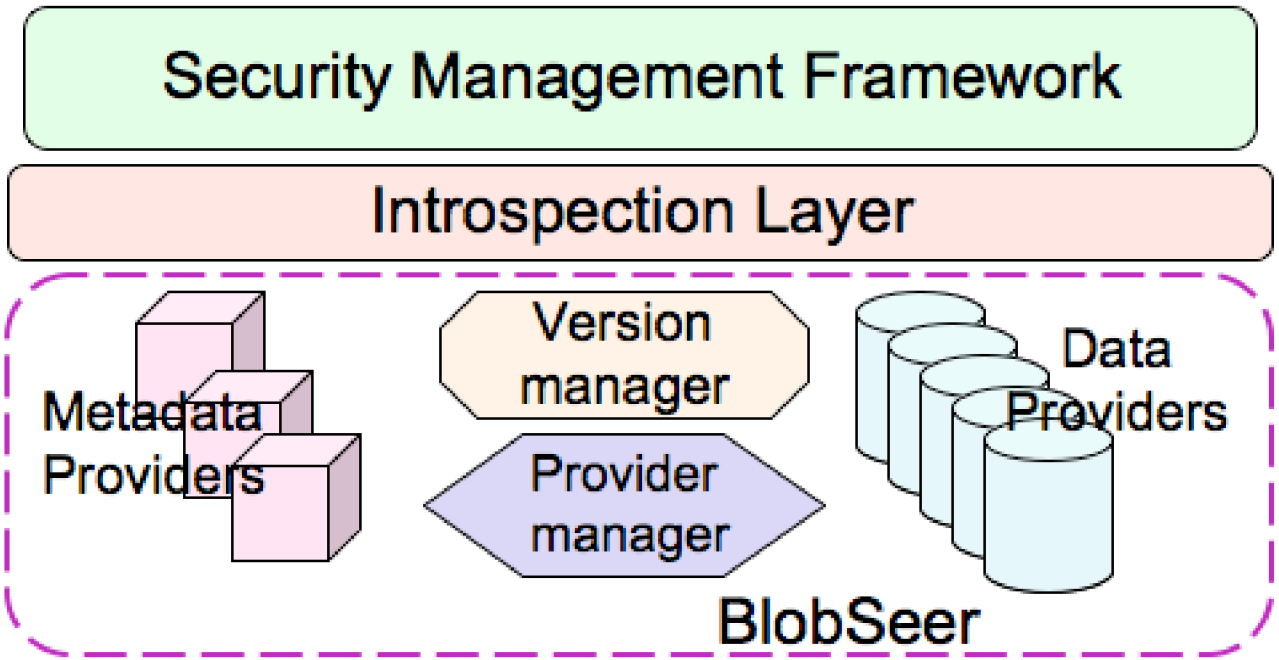
- **BlobSeer Trust Level** – Ana-Maria Lepar
  (Bachelor project, Bucharest)

Coordinators: Catalin Leordeanu (PUB)
Alexandra Carpen-Amarie (INRIA)

Introduction

Our goals:

- Enable the detection of malicious clients for large scale data management systems
- Develop a complete security solution for BlobSeer



Malicious Client Detection

Types of malicious activity:

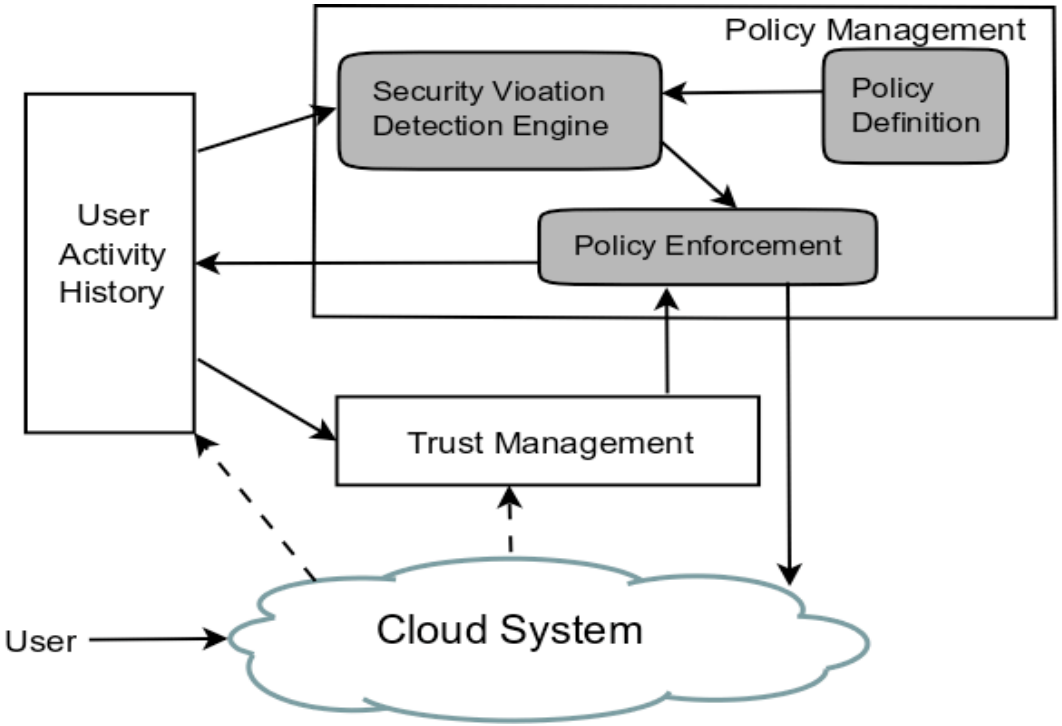
- **Protocol Breach**
  - *Heavy writing without the creation of a new version (WriteNoPublish)*
  - Publish the version and create the metadata tree, but write nothing actually(PublishNoWrite).

- **Policy enforcement** – matching of predefined policies
  - Denial of Service
  - Detection of suspicious activity
  - Crawling
  - Repeated reading of the same data
  - Abnormal client activity

Malicious Client Detection

Challenges:

- BlobSeer has no authentication mechanism or any way to distinguish the users
- each client accesses the information in the same way
- there is no way of certifying the users



Policy enforcement

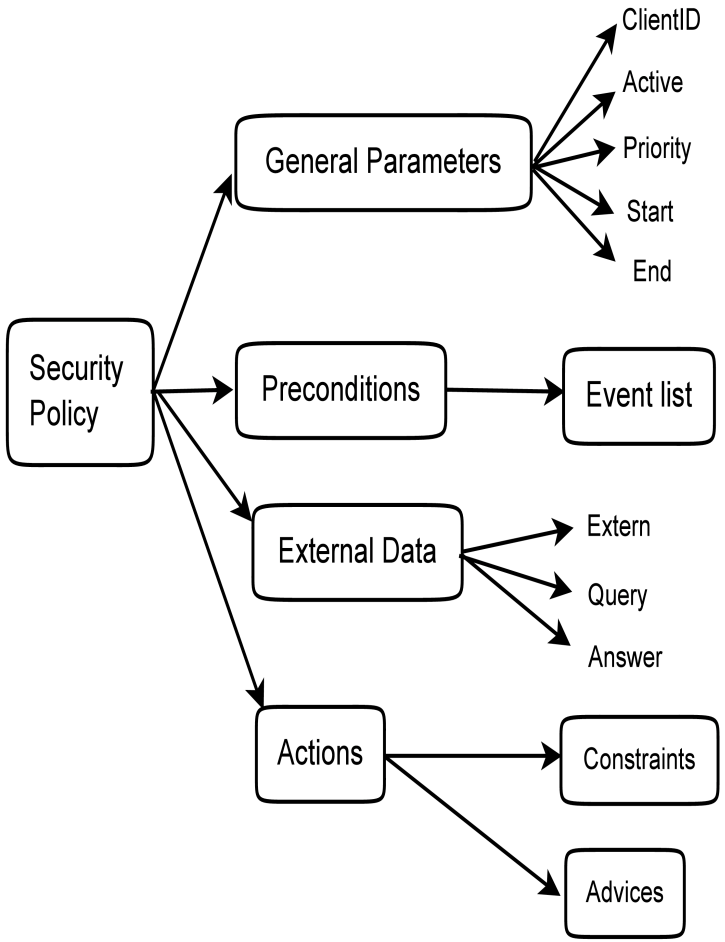
Advantages:

- Simple to use and easy to customize XML patterns that describe malicious activity
- It can take complex actions in the case of policy violations
    - Directly
    - Through the Trust Level

Disadvantages:

- Unable to adapt to unknown malicious activity
- Delay due to the monitoring infrastructure and storage of user history.

Policy Definition

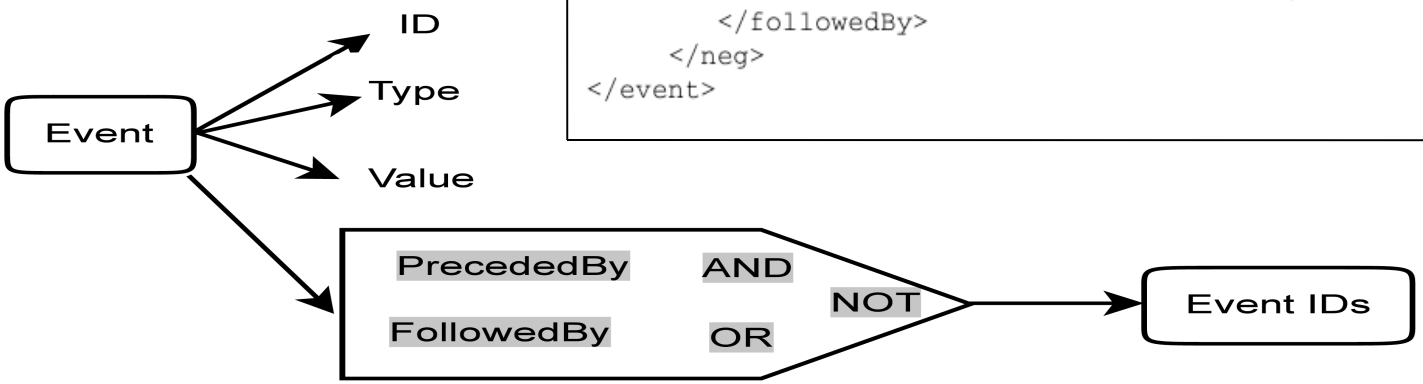



```
<securityPolicy id="1_25">
     <clientID rvalue="c" value="c"/>
     <active value="true"/>
     <priority value="1"/>
     <start value="w1"/>
      <end value="c1"/>
      <preconditions>
         <event id="w1" type="prov_write_summary">
         ...
         </event>
         ...
         <event id="p2" type="vman_write">
         ...
         </event>
         <event id="c1" type="check">
         ...
         </event>
      </preconditions>
...
</securityPolicy>
```

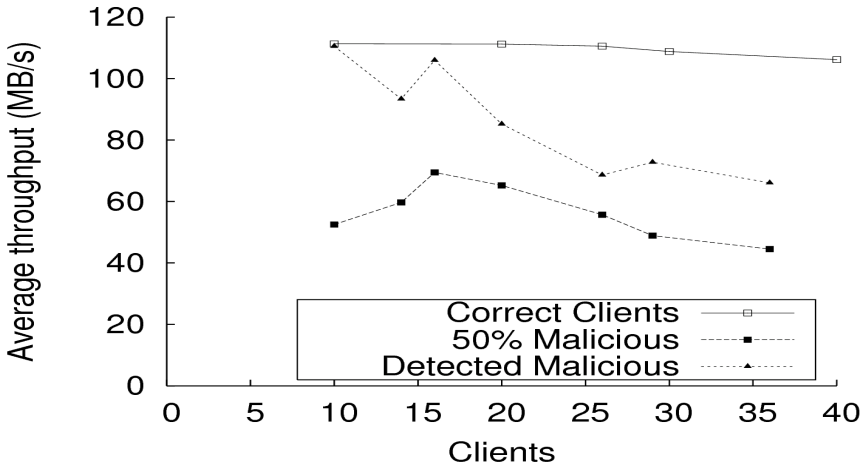
Policy Definition

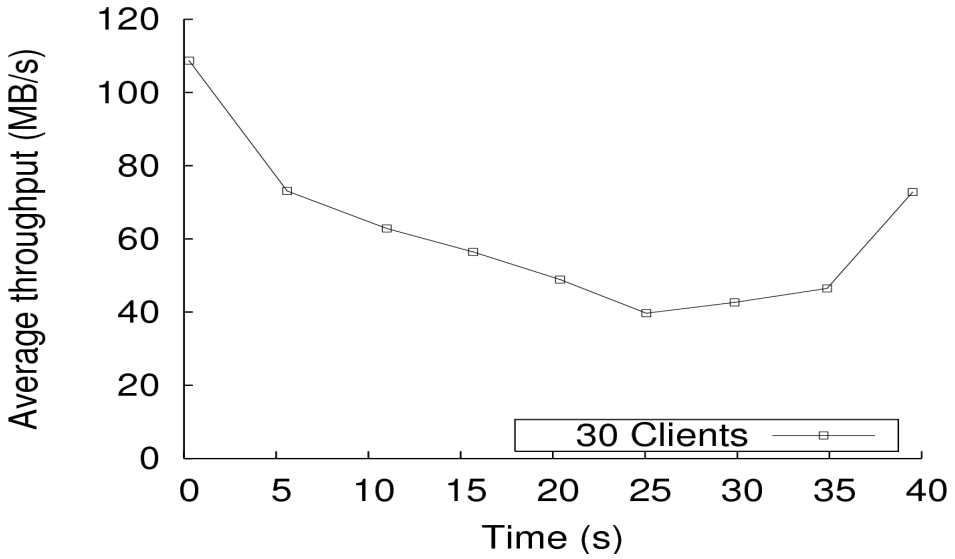
Events may form logic expressions using the operations AND, OR, NOT, PrecededBy or FollowedBY

```
<event id="w1" type="prov_write_summary">
     <blobId id="bId" rvalue="" value="b"/>
     <clientID rvalue="" value="c"/>
     <WriteSizeCount id="wsc" rvalue=""/>
     <thresholdWriteSize id="tws" value="1080"/>
     <supThresholdWriteSize id="stws" value="2000"/>
     <firstDate id="fd" rvalue=""/>
     <lastDate id="ld" rvalue=""/>
     <distance id="dist" value="7000"/>
     <continuous>
        <refEvent value="wa"/>
        <refEvent value="bId"/>
        <refEvent value="wsc"/>
        <refEvent value="ld"/>
     </continuous>
     ...
     <neg>
        <followedBy>
             <refEvent value="p2"/>
             <count value="1"/>
             <distance value="<= fd + dist"/>
        </followedBy>
     </neg>
</event>
```



Experimental Results





Trust Management

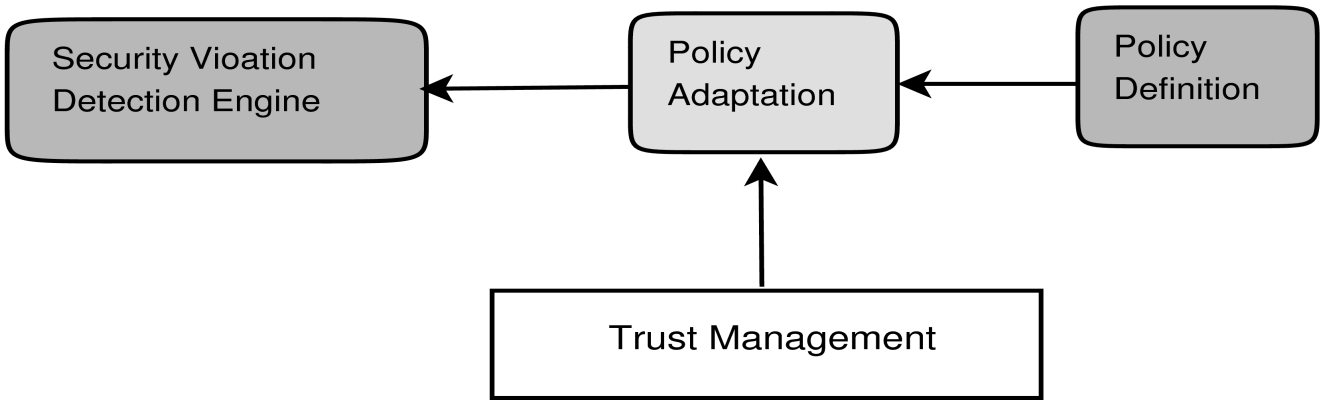
- Each event has an effect on the Trust Level of the user

$$\sum A_i * Age(time_i) * SystemState(i)$$

$$Age(i) = \begin{cases} \text{CONST}, i \in [0, UMM] \\ a^{i}, i > UMM \end{cases}$$

- Also uses the system state of the providers to determine the gravity of the malicious activity
- The Trust level can be between 0 and 100.
- If a user has a high Trust Level he may be rewarded with relaxed security policies for a period of time.
- A low Trust level may be punished by more restrictive policies.

Policy Adaptation



Generates custom policies according to each client's Trust Level:

- fair clients will have more relaxed policies
- malicious clients will have stricter policies

Policies will be customized using a set of predefined rules. The rules specify which parameters of the policies can be modified and by what amount.

Conclusions and future work

Conclusions:

- We designed a malicious client detection architecture
- Right now, the Policy Enforcement and Trust Level modules are functional
- We tested the Policy Enforcement module using large scale deployments

Future work:

- Build a complete, functional security framework for BlobSeer
- Finish the policy adaptation implementation
- Develop complex test scenarios

Questions ?