A Component Security Infrastructure

Y. David Liu
Scott Smith
http://www.jcells.org
Motivation

Secure software systems

Build software systems using components

SDSI/SPKI

Cells

Secure software systems at the component level
Goals for a Component Security Infrastructure

- **Simplicity**
  - Complex protocols will be misused

- **Generality**
  - Applicable across a wide range of domains

- **Interoperability**
  - Security policies shared between components, others

- **Extensibility**
  - Evolves as component architecture evolves
Background: Cells

- A new distributed component programming language [Rinat and Smith, ECOOP2002]

![Diagram showing Cells A and B with header and body, services, and connectors. The diagram includes symbols for 'plugout' and 'plugin'].

Service =  
Connector =
Background: SDSI/SPKI

- Basis of our security infrastructure
- **Features**
  - Principal with public/private key pair
  - Decentralized name service
    - Extended names, name certificate
    - Group membership certificate
  - Access control
    - Principal with ACL
    - Delegation model: authorization/revocation certificate
Principles of Component Security

● Each component should be a principal
  – Traditional principals: users, locations, protection domains, …
  – New idea: Components as principals

● Components are known to outsiders by their public key

● Components each have their own secured namespace for addressing other components

● Components may be private
Cell Identifiers: \textit{CID}

- $\textit{CID}$ = the public key in the key pair generated by public key cryptosystem
  - $\textit{CID}$ is a secured cell identity
- Universally unique
  - No two cells share the same $\textit{CID}$
- Outgoing messages signed by $\textit{CID}^{-1}$ and verified by $\textit{CID}$
CVM Identity

With President Cells:

- Universe is homogeneously composed of cells
- Locations are also principals
  - Locations are represented by cells and each cell is a principal
- Unique CVM identity via its President
Cell Header Security Information

<table>
<thead>
<tr>
<th>CID</th>
<th>CID⁻¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLT</td>
<td></td>
</tr>
<tr>
<td>SPT</td>
<td></td>
</tr>
<tr>
<td>CertSTORE</td>
<td></td>
</tr>
</tbody>
</table>

Identity/Key
Naming Lookup Table
Security Policy Table
Certificate Store (Delegation)
Cell Reference

Unifies many notions in one concept:

- A locator of cells
- A capability to a cell
  - No cell reference, no access
- A programming language construct: reference
- Corresponds to a SDSI/SPKI principal certificate
Name Services

- **CIDs vs. Names**
  - *CIDs* serve as universal identifiers, but names are still necessary
  - Extended name mechanism enables a cell to refer to another cell even if its CID is unknown

- **Our name service is based on SDSI/SPKI**

- **Improvements:**
  - Fewer certificates needed due to on-line nature
  - More expressive lookup algorithm
SDSI/SPKI Extended Names

Bob’s Andy?

Local Name Certificate

### Andy

- **ID**: 2..9
- **LOC**: sdsi://starwars.com

Local Name Certificate

<table>
<thead>
<tr>
<th>Bob</th>
<th><strong>ID</strong>: 1..3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>LOC</strong>: sdsi://cs.jhu.edu</td>
</tr>
</tbody>
</table>

- **ID**: 7..1
- **ID**: 1..3
- **ID**: 2..9
# SDSI/SPKI Groups

**Local Name Certificate**

<table>
<thead>
<tr>
<th>Name</th>
<th>ID</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>1..3</td>
<td>sdsi://cs.jhu.edu</td>
</tr>
</tbody>
</table>

**Group Membership Certificate**

<table>
<thead>
<tr>
<th>Group</th>
<th>Member(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Friend</td>
<td>{Bob, Bob’s Andy}</td>
</tr>
</tbody>
</table>

**Andy**

<table>
<thead>
<tr>
<th>ID</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>2..9</td>
<td>sdsi://starwars.com</td>
</tr>
</tbody>
</table>
Cell Naming Lookup Table

<table>
<thead>
<tr>
<th>Bob</th>
<th>CID : 1..3</th>
<th>CVM : 4..7</th>
<th>LOC : cell://cs.jhu.edu</th>
</tr>
</thead>
<tbody>
<tr>
<td>Friend</td>
<td>{Bob, Bob’s Andy}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Andy</th>
<th>CID : 2..9</th>
<th>CVM : 5..5</th>
<th>LOC : cell://starwars.com</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CID : 1..3</td>
<td>CID : 4..7</td>
<td>CID : 5..5</td>
</tr>
</tbody>
</table>

CID: 7..1
CID: 9..5
cell://home.org

cell://cs.jhu.edu

cell://starwars.com
Cell Naming Lookup Table

- Online nature makes local name certificates unnecessary, unlike SDSI/SPKI
  - More suited for mobility
- Maintained by naming lookup interface, a concept closer to programming languages
- Naming entries can be effectively secured by using hooks
- Compatible with SDSI/SPKI
Name Lookup Process

Bob's Andy?

<table>
<thead>
<tr>
<th>Bob</th>
<th>CID</th>
<th>CVM</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1..3</td>
<td>4..7</td>
<td>cell://cs.jhu.edu</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Andy</th>
<th>CID</th>
<th>CVM</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2..9</td>
<td>5..5</td>
<td>cell://starwars.com</td>
</tr>
</tbody>
</table>
A More Expressive Algorithm

CID 1..3

CID 2..9

CID 9..9

Anthony
CID : 9..9
CVM : 4..7
LOC : cell://cs.jhu.edu

Andy
CID : 2..9
CVM : 5..5
LOC : cell://starwars.com

Tony
Andy’s Anthony
### Cycles

<table>
<thead>
<tr>
<th>Tony’s Bob?</th>
<th>Alice</th>
<th>Anthony</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CID</strong> 1..3</td>
<td><strong>CID</strong> 1..3\n<strong>CVM</strong> 5..7\n<strong>LOC</strong> cell://cs.jhu.edu</td>
<td>Alice’s Tony</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Andy</th>
<th><strong>CID</strong> 2..9\n<strong>CVM</strong> 5..5\n<strong>LOC</strong> cell://starwars.com</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tony</td>
<td>Andy’s Anthony</td>
</tr>
</tbody>
</table>
A cycle exists

Same local name expansion entry encountered twice

Solution:
- Keep track of the path
- Raise an exception if the same name encountered twice
Security Policy

- Each cell holds a security policy table, SPT.
- Each policy is a 5-tuple.

<table>
<thead>
<tr>
<th>subject</th>
<th>resource</th>
<th>access right</th>
<th>hook</th>
<th>deleg bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>thiscell</td>
<td>connect</td>
<td>NULL</td>
<td>0</td>
</tr>
<tr>
<td>Group1</td>
<td>thiscell</td>
<td>invoke</td>
<td>NULL</td>
<td>1</td>
</tr>
<tr>
<td>Alice</td>
<td>Tim</td>
<td>invoke</td>
<td>h</td>
<td>0</td>
</tr>
</tbody>
</table>
Subjects, Resources, Access Rights

- **Subjects**
  - Cells and a group of cells
  - Local names, extended names, cell references

- **Resources**
  - Services, connectors, operations
  - Partial order relations among them

- **Access rights**
  - Connect and invoke

- **Application level protection:** meaningful services and meaningful connections.
Hooks

- Designed for fine-grained access control
  - Protect a naming lookup entry: $\text{lookup}(\text{"Tony"})$
  - Protect a specific file: $\text{read}(\text{"abc.txt"})$

- Associated with operations

- Operation parameters verified via a predicate

- Predicate checked when the associated operation is triggered
  - Example:
    $$\text{Hook}_{\text{lookup}}(\text{arg1}) = \{ \text{arg1=\text{"Tony"}} \}$$
SDSI/SPKI Delegation

“Access Granted”

ID 3..3
AuthC1
AuthC3

ID 1..1
AuthC1
AuthC3

ID 5..5
Cell Delegation

- Implements SDSI/SPKI delegation
- Each cell holds all certificates (both delegation and revocation) in a certificate store.
- Security policy table supports delegation
  - The owner of the resource might not be this cell
  - The delegation bit indicating whether certificates can be further delegated
- Certificates are implicitly passed for delegation chain detection
  - No need for manual user intervention
Goals Revisited

- **Simplicity**
  - No complex algorithms/data structures
  - Clearly defined principals and resources

- **Generality**
  - Not just cells, but components in general
  - Not limited to certain applications

- **Interoperability**
  - Built on SDSI/SPKI standard
  - Communicate with any infrastructure that supports SDSI/SPKI

- **Extensibility**
  - Consideration for future additions: mobility, etc
Future Work

- Security for Mobile Components
  - Cells can migrate
  - Mobile devices, PDAs
- Hierarchical Security Policy
- Interoperability
Dynamic Component

- Components are named, addressable entities, running at a particular location.
- Components have interfaces which can be invoked.
- Components may be distributed across the network.
Summary

- Security infrastructure in a component programming language
- Cell identity and CVM identity (president cell)
- Naming lookup table/interface
  - More expressive lookup algorithm and cycle detection
- Fine-grained access control
- Unification of security artifacts and programming language ones
- Formalization of SDSI/SPKI
- API from programming language perspective
Traditional Security Model

allow request from alice.jhu.edu
...Fails for Mobile Devices

allow request from
alice.jhu.edu
Cell Security Infrastructure

allow request from CVM with CID 3333333

Cell1 (CID: 7654321)

ResourceCVM
CID: 1111111

Cell2 (CID: 1234567)

aliceCVM
CID: 3333333

jhu.edu

bobCVM
CID: 5555555
...Adapts Well with Mobile Devices

allow request from CVM with CID 3333333
Extended Name

An extended name is a sequence of local names \([n_1, n_2, \ldots, n_k]\), where each \(n_{i+1}\) is a local name defined in the name space of the cell \(n_i\).
Example: Traditional Security Model

allow request from
alice.jhu.edu
...Fails in Cell Migration

allow request from alice.jhu.edu

Comp1

Resource

Comp2

alice

bob

jhu.edu
Example:

Cell Security Infrastructure

allow request from cell with CID 1234567

Cell1 (CID: 7654321)

Resource

Cell2 (CID: 1234567)

alice

bob

jhu.edu
...Adapts Well in Cell Migration

allow request from cell with CID 1234567

Cell1 (CID: 7654321)

Cell2 (CID: 1234567)

alice

jhu.edu

bob

Resource