A Component Security Infrastructure

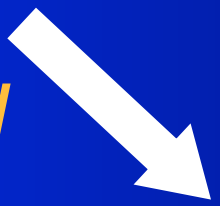
Y. David Liu
Scott Smith
http://www.jcells.org

Motivation

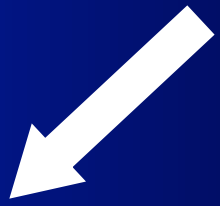
Secure software systems

Build software systems using components

SDSI/SPKI



Cells



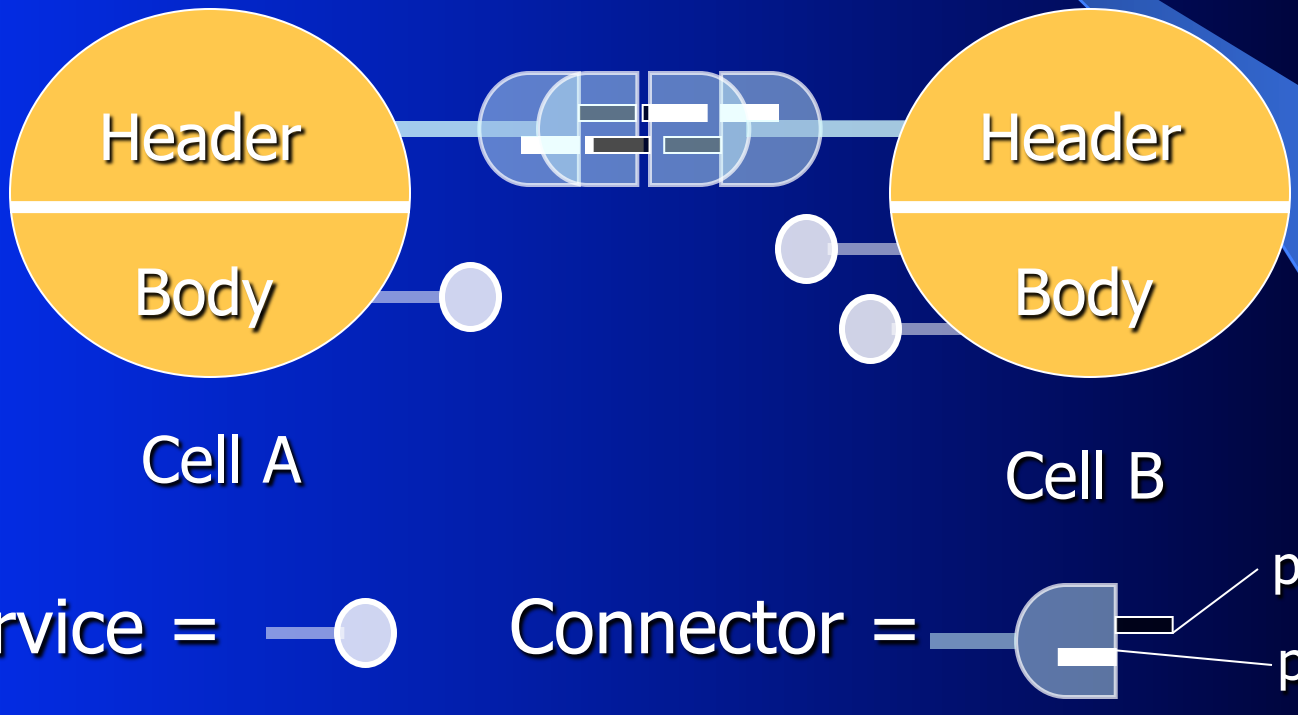
Secure software systems at the component level

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]



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

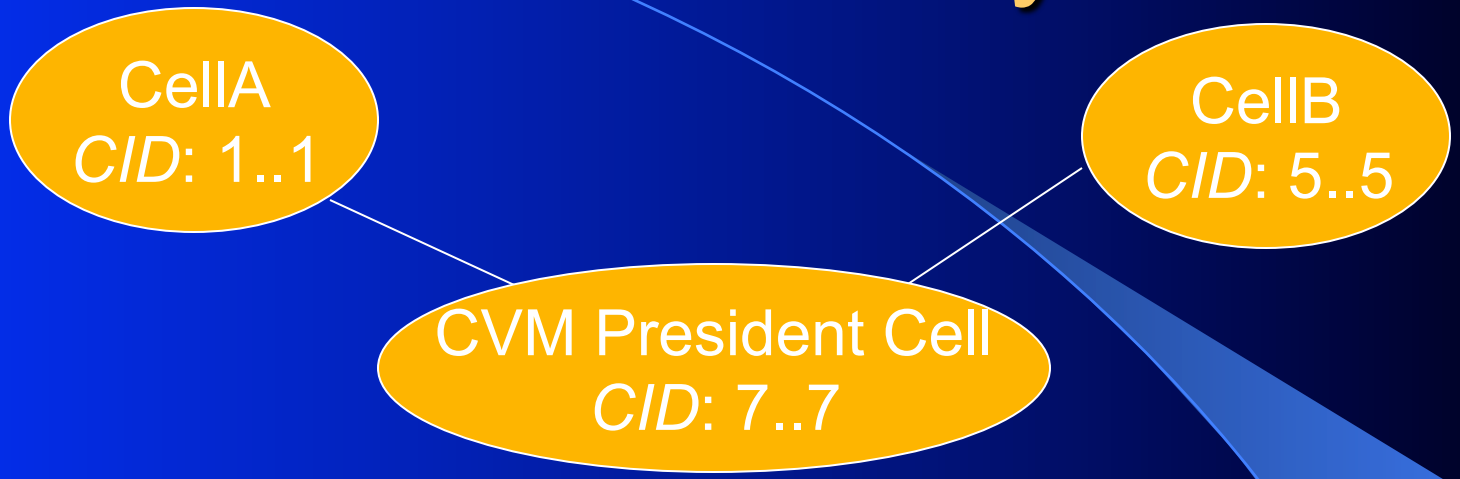
Principles of Component Security

- 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: *CID*

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

Identity/Key

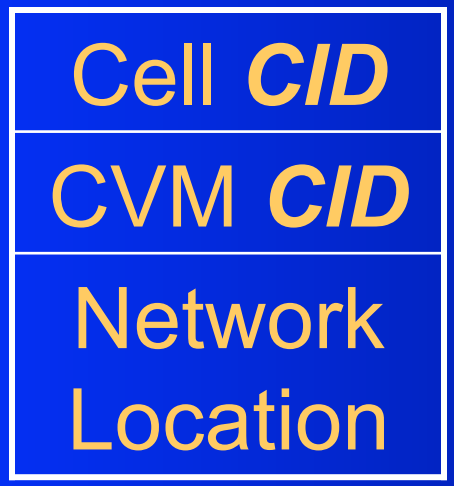
Naming Lookup Table

Security Policy Table

Certificate Store (Delegation)

Cell Reference

Unifies many notions in one concept:

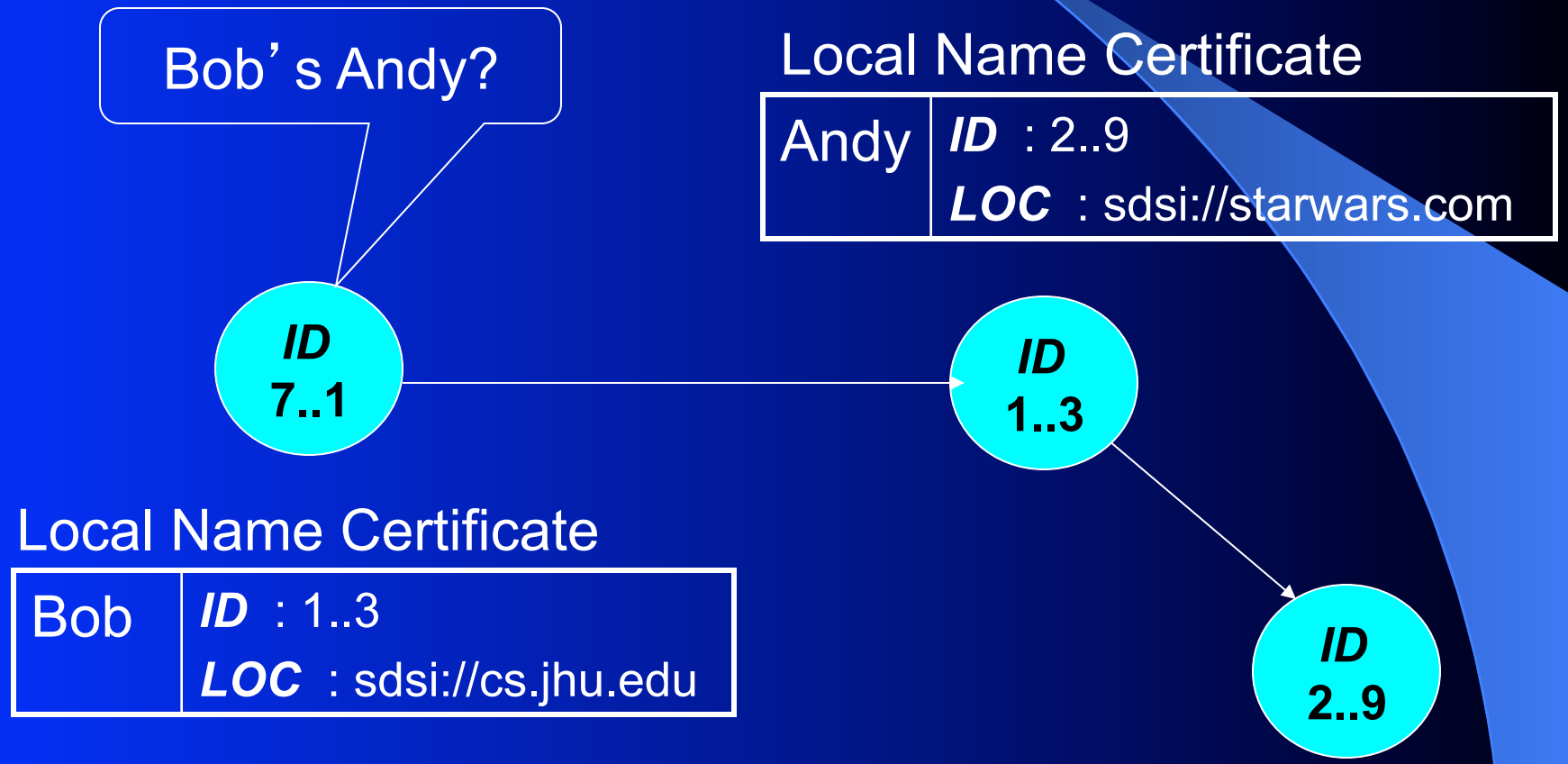


- A locator of cells
- A capability to a cell
  - No cell reference, no access
- A programming language construct: *reference*
- Corresponds to a SDSI/SPKI principal certificate

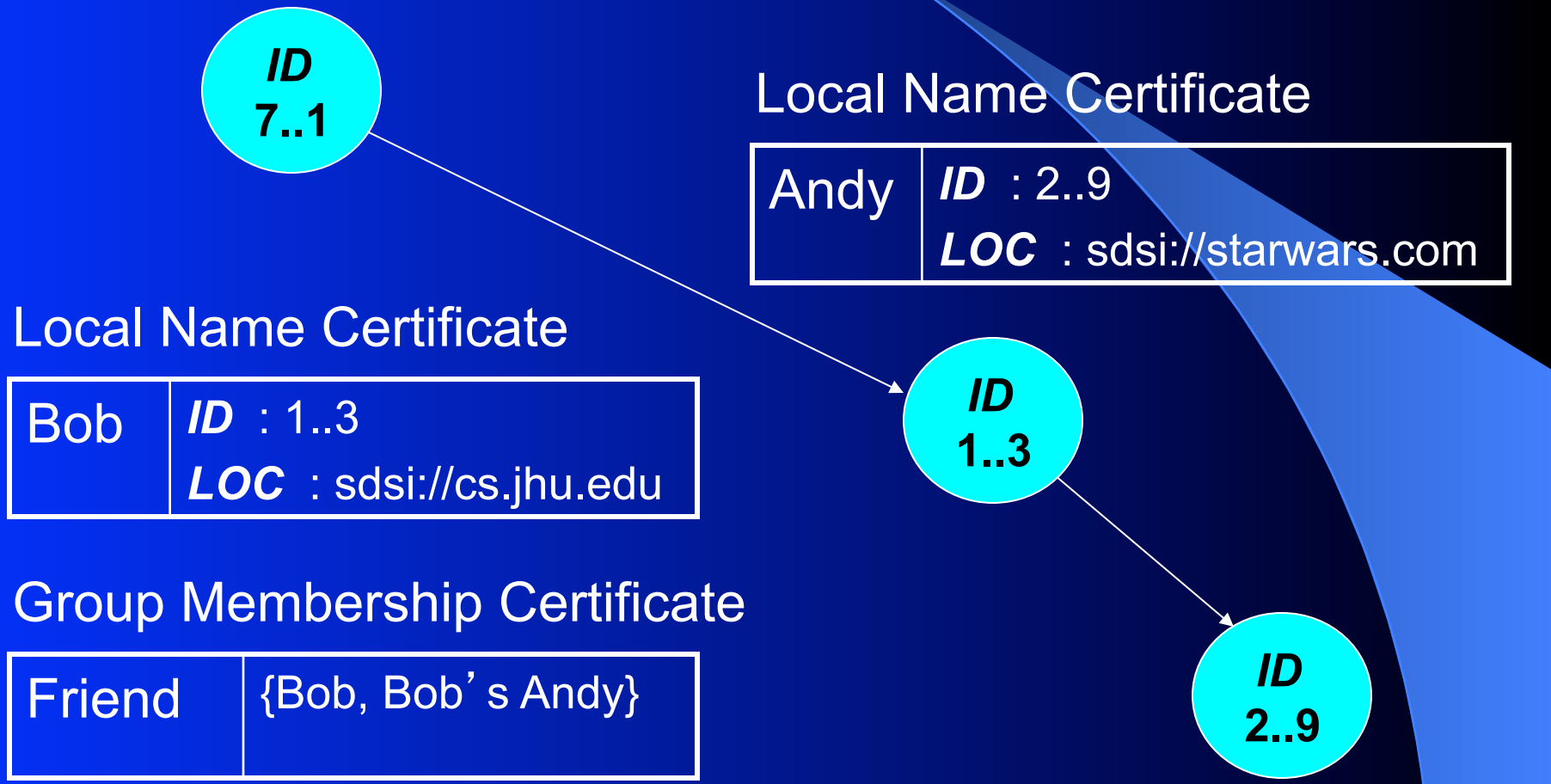
Name Services

- *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

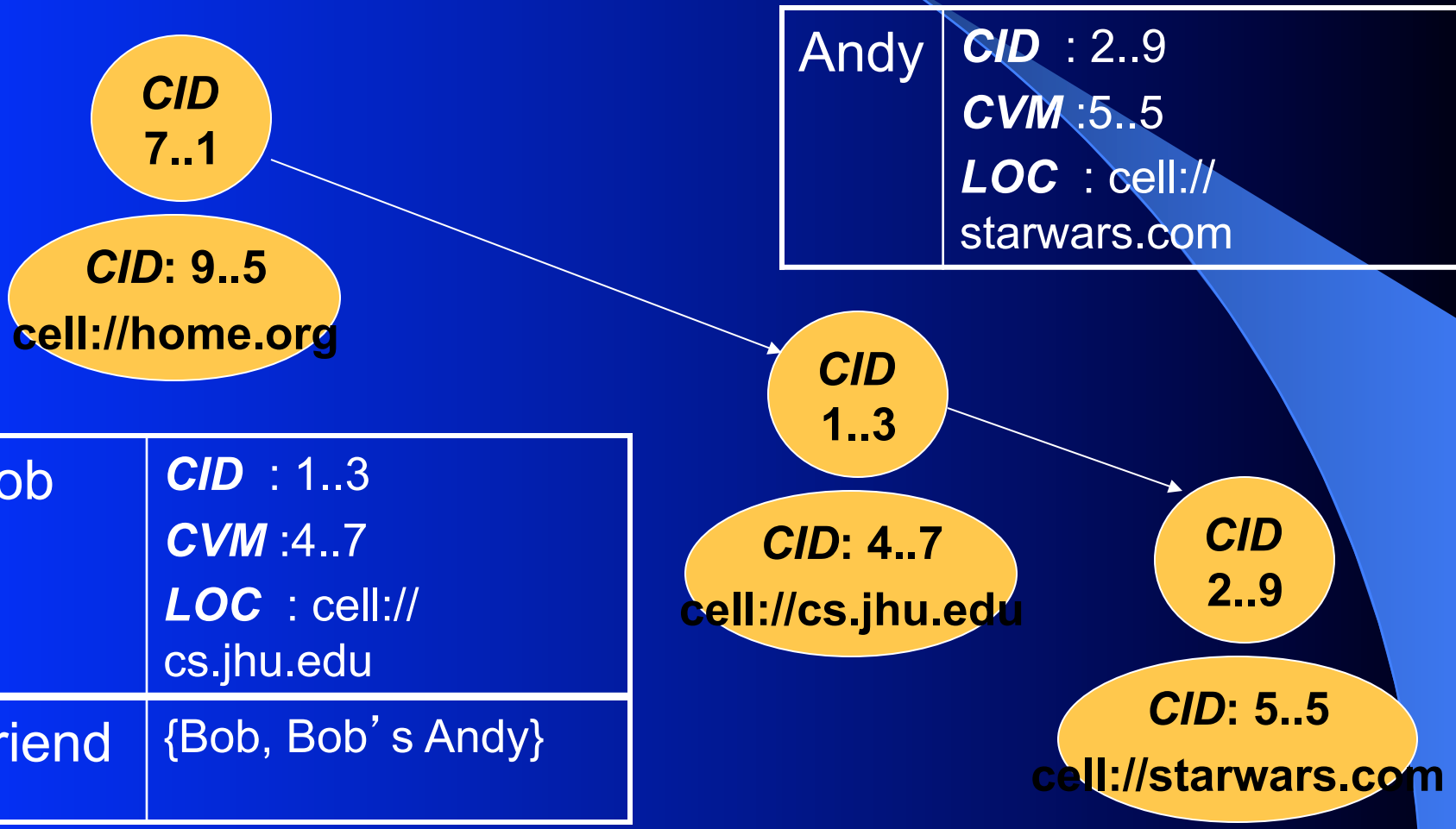
SDSI/SPKI Extended Names



SDSI/SPKI Groups



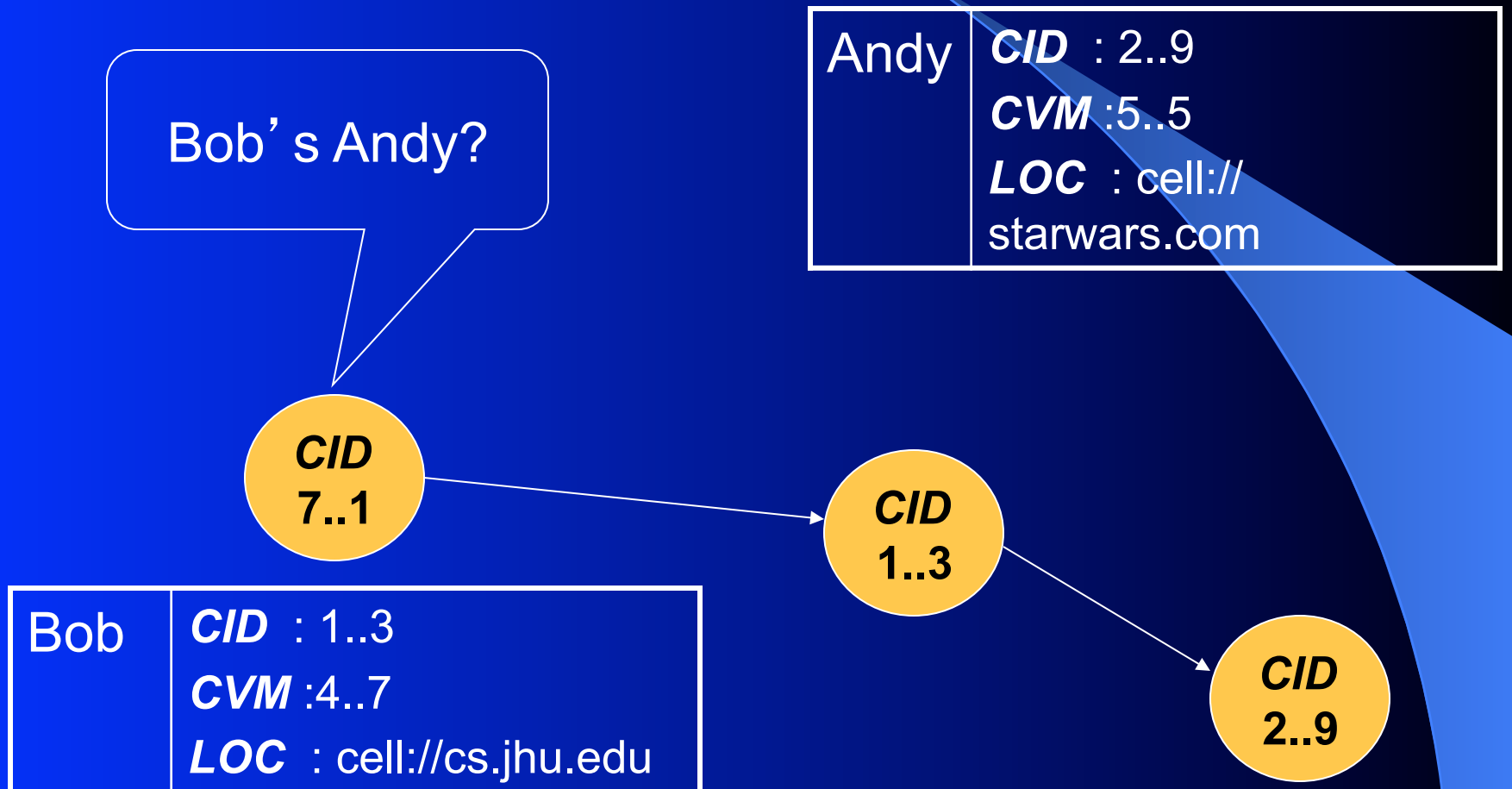
Cell Naming Lookup Table



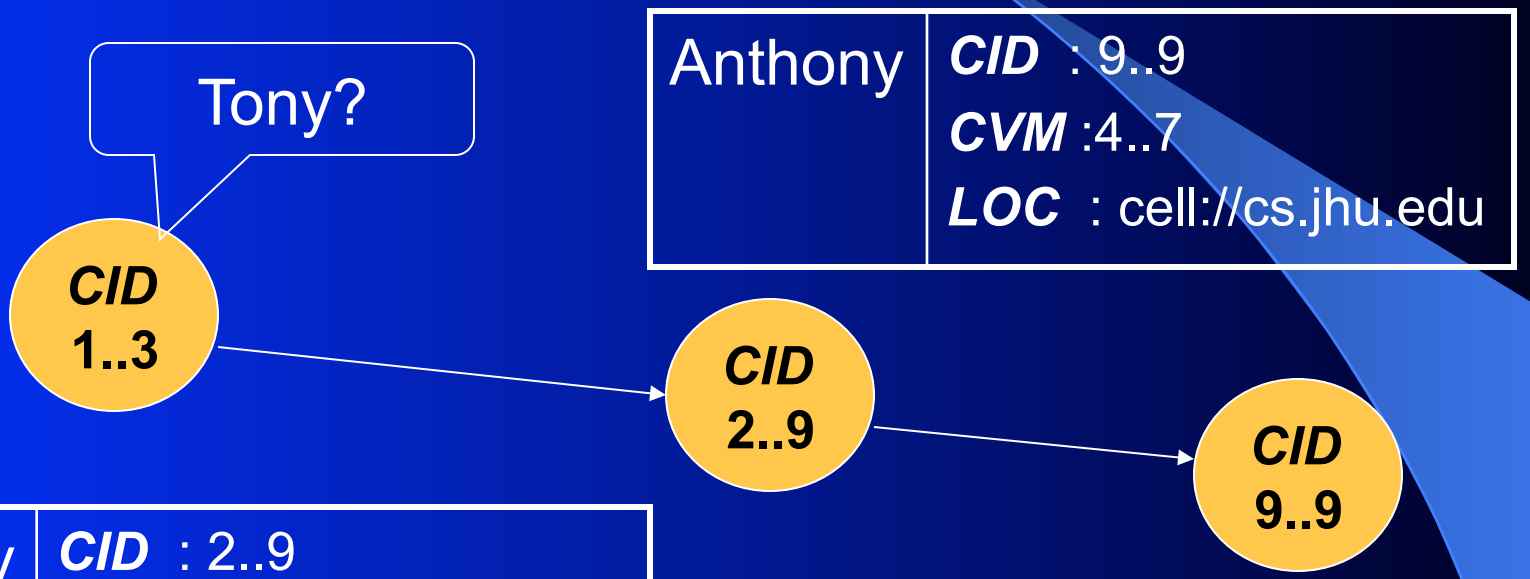
Cell Naming Lookup Table

- 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



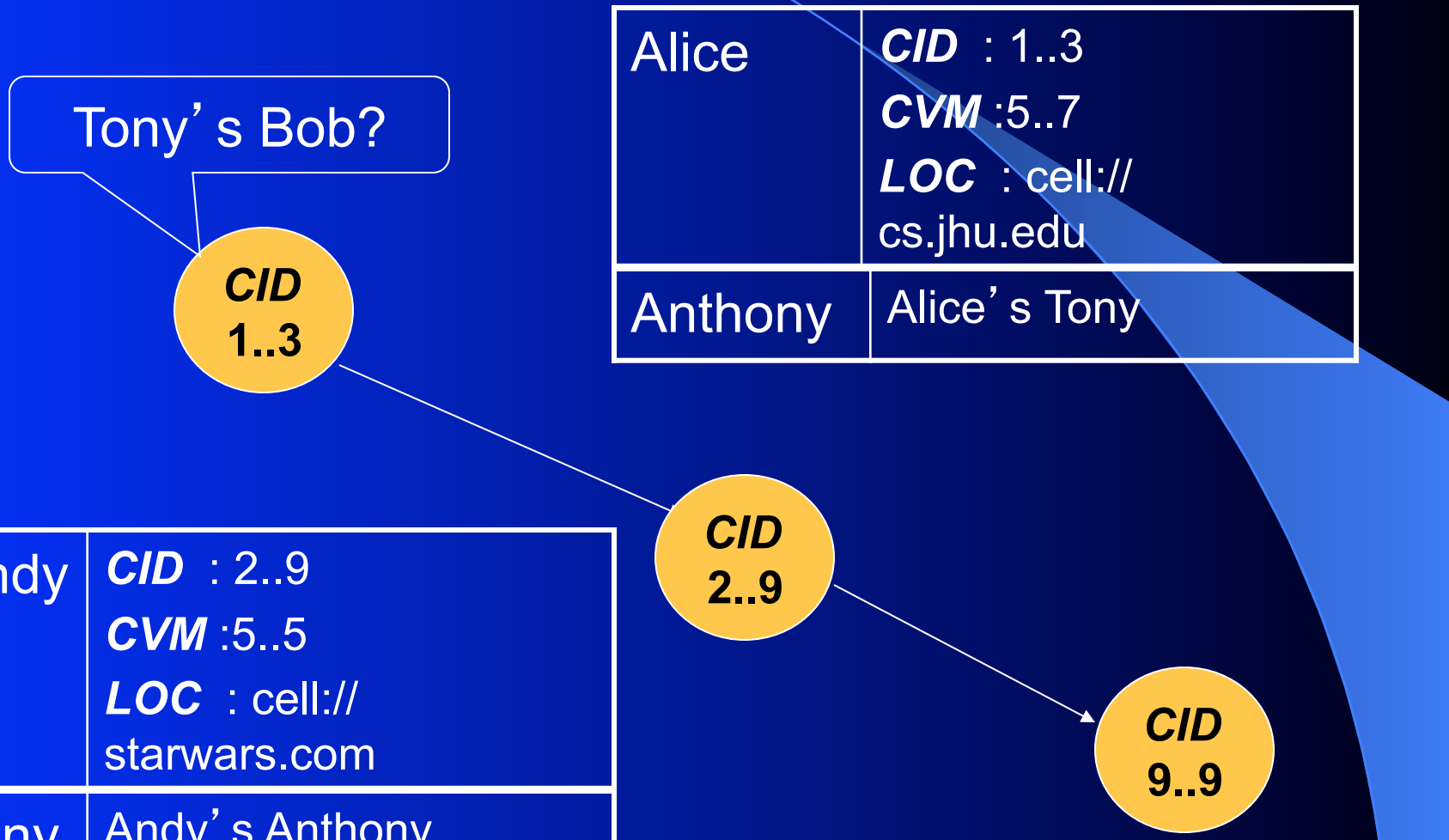
A More Expressive Algorithm



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


Tony's Bob?
CID
1..3
Alice
CID : 1..3
CVM :5..7
LOC : cell://
cs.jhu.edu
Anthony
Alice's Tony
CID
2..9
Andy
CID : 2..9
CVM :5..5
LOC : cell://
starwars.com
Tony
Andy's Anthony
CID
9..9

Cycle Detection Sketch

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.

subject	resource		access right	hook	deleg bit
	owner	unit			
Bob	thiscell	connector1	connect	NULL	0
Group1	thiscell	service1	invoke	NULL	1
Alice	Tim	service1	invoke	h	0

Subjects, Resources, Access Rights

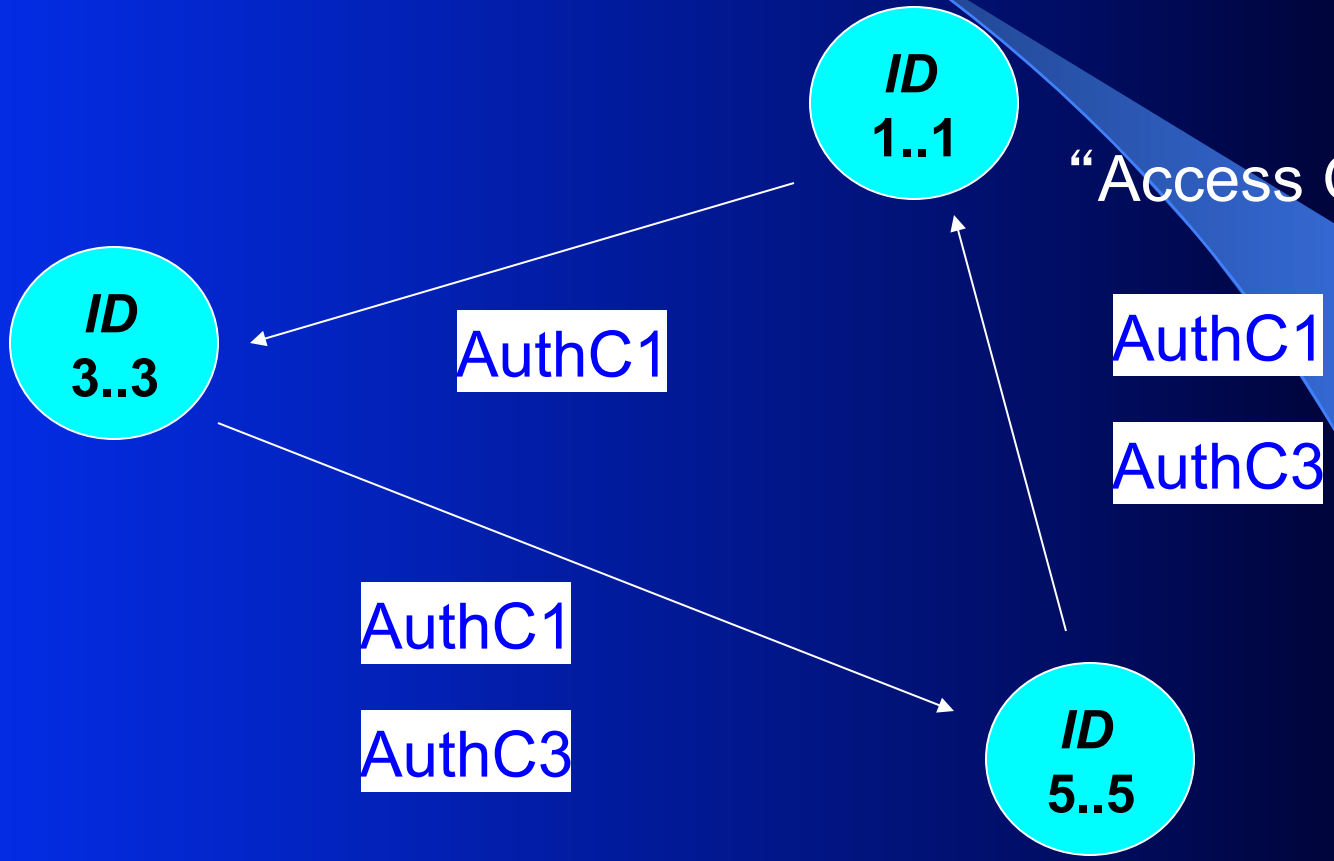
- Subjects
  - Cells and a group of cells
  - Local names, extended names, cell references
- Resources
  - Services, connectors, operations
  - Partial order relations among them
- Access rights
  - Connect and invoke
- Application level protection: meaningful services and meaningful connections.

Hooks

- Designed for fine-grained access control
  - Protect a naming lookup entry: *lookup("Tony")*
  - Protect a specific file: *read("abc.txt")*
- Associated with operations
- Operation parameters verified via a predicate
- Predicate checked when the associated operation is triggered
  - Example:

    $Hook_{lookup}(arg1) = \{ arg1 = \text{"Tony"} \}$

SDSI/SPKI Delegation



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 *thiscell*
  - 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

jcells.org

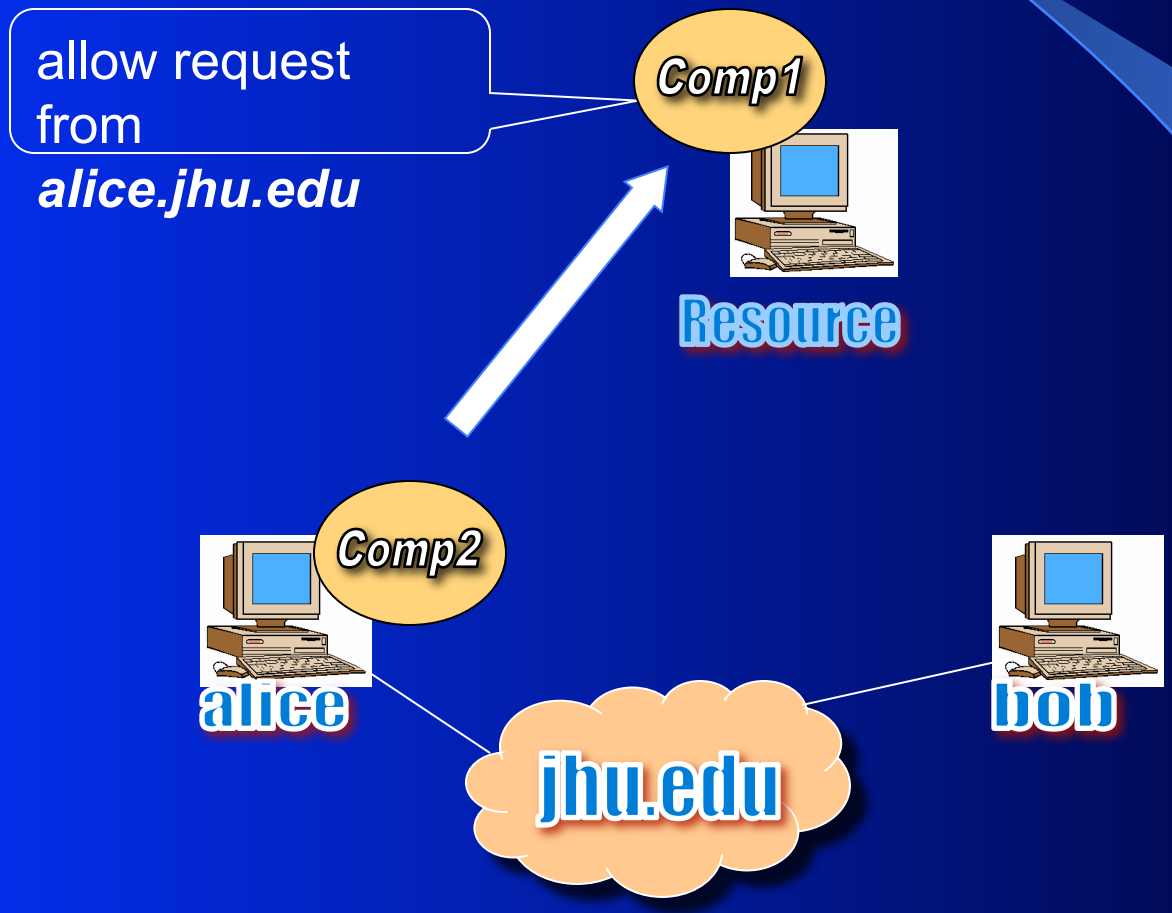
Dynamic Component

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

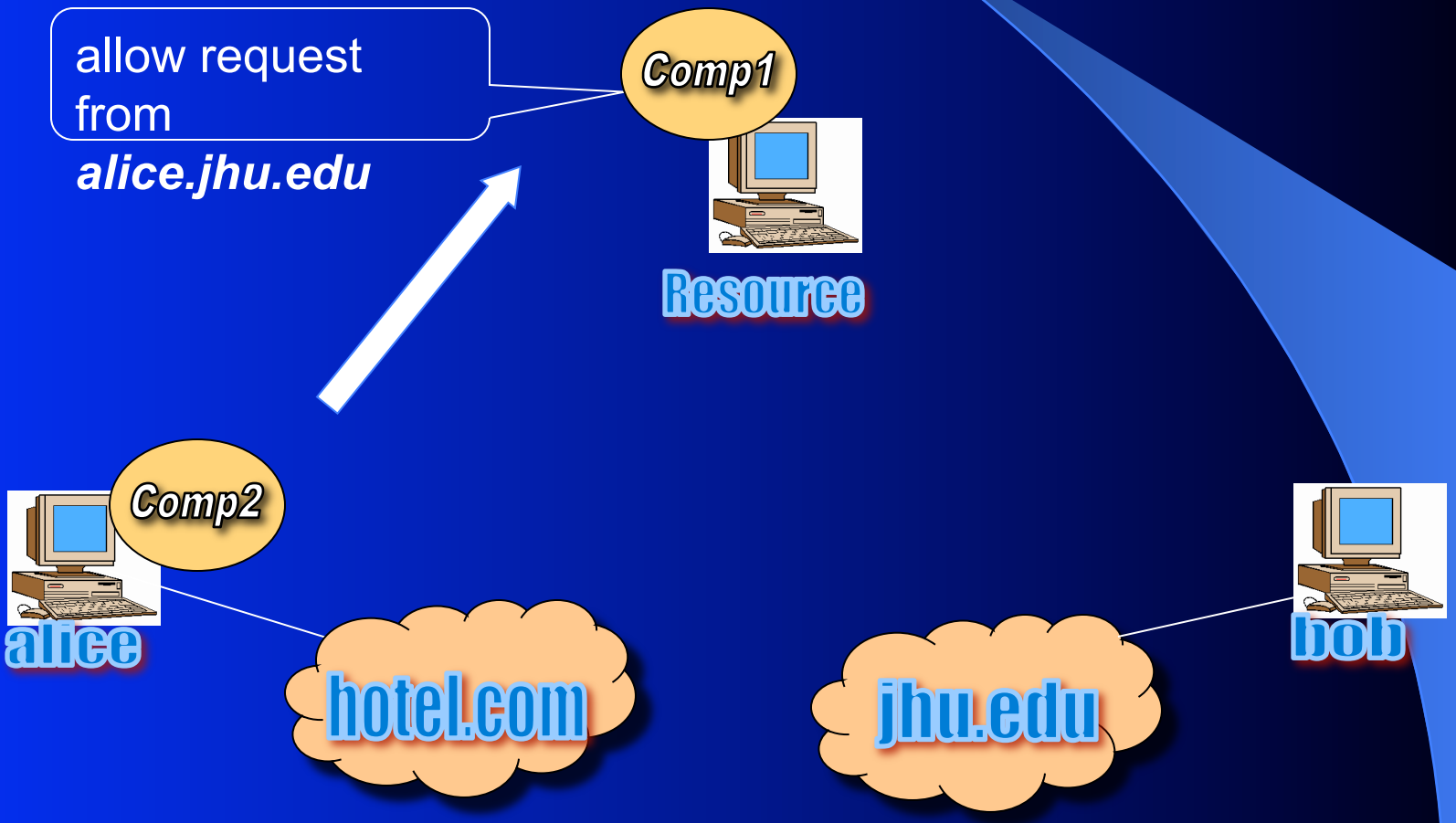
Summary

- 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

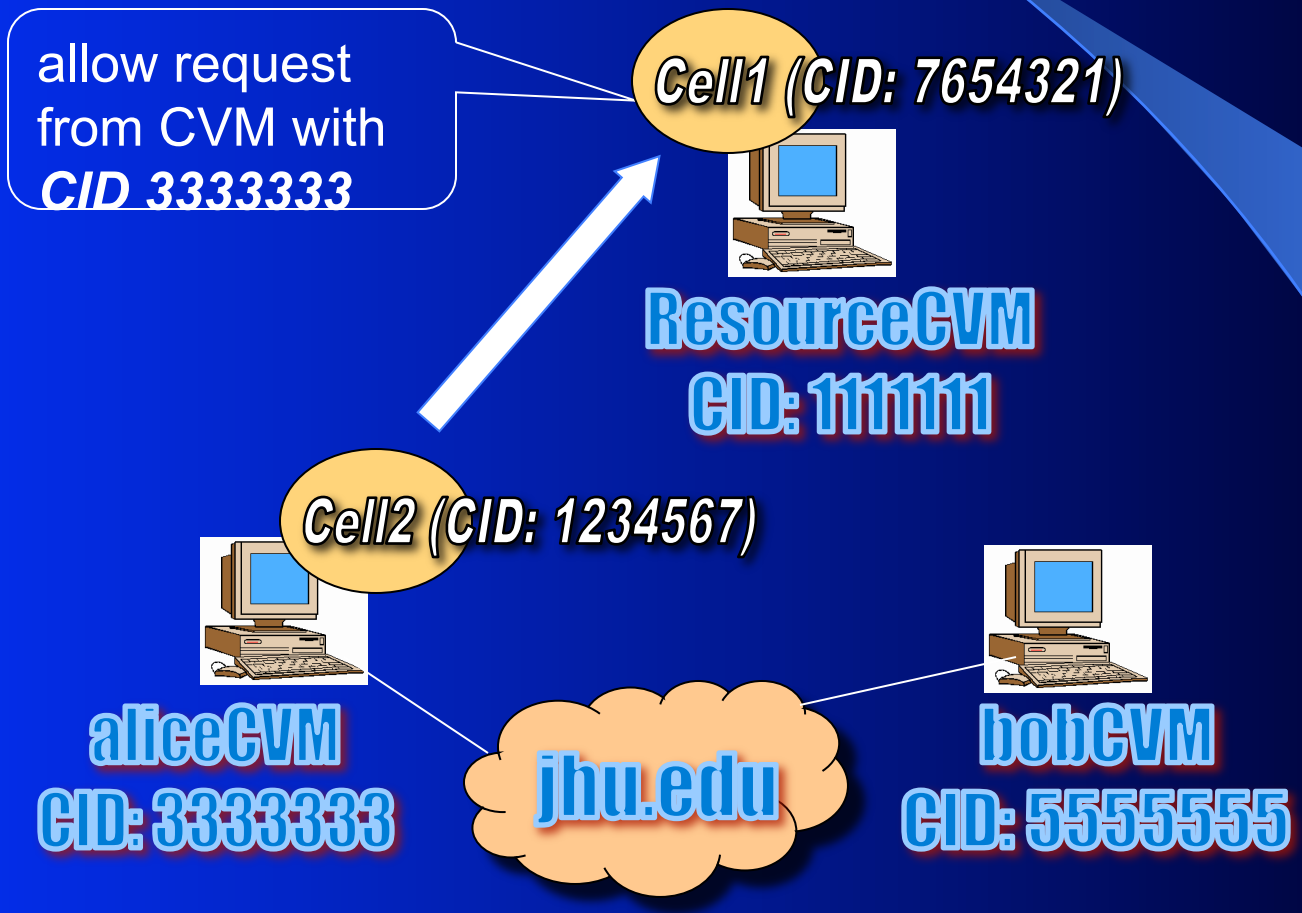
Traditional Security Model



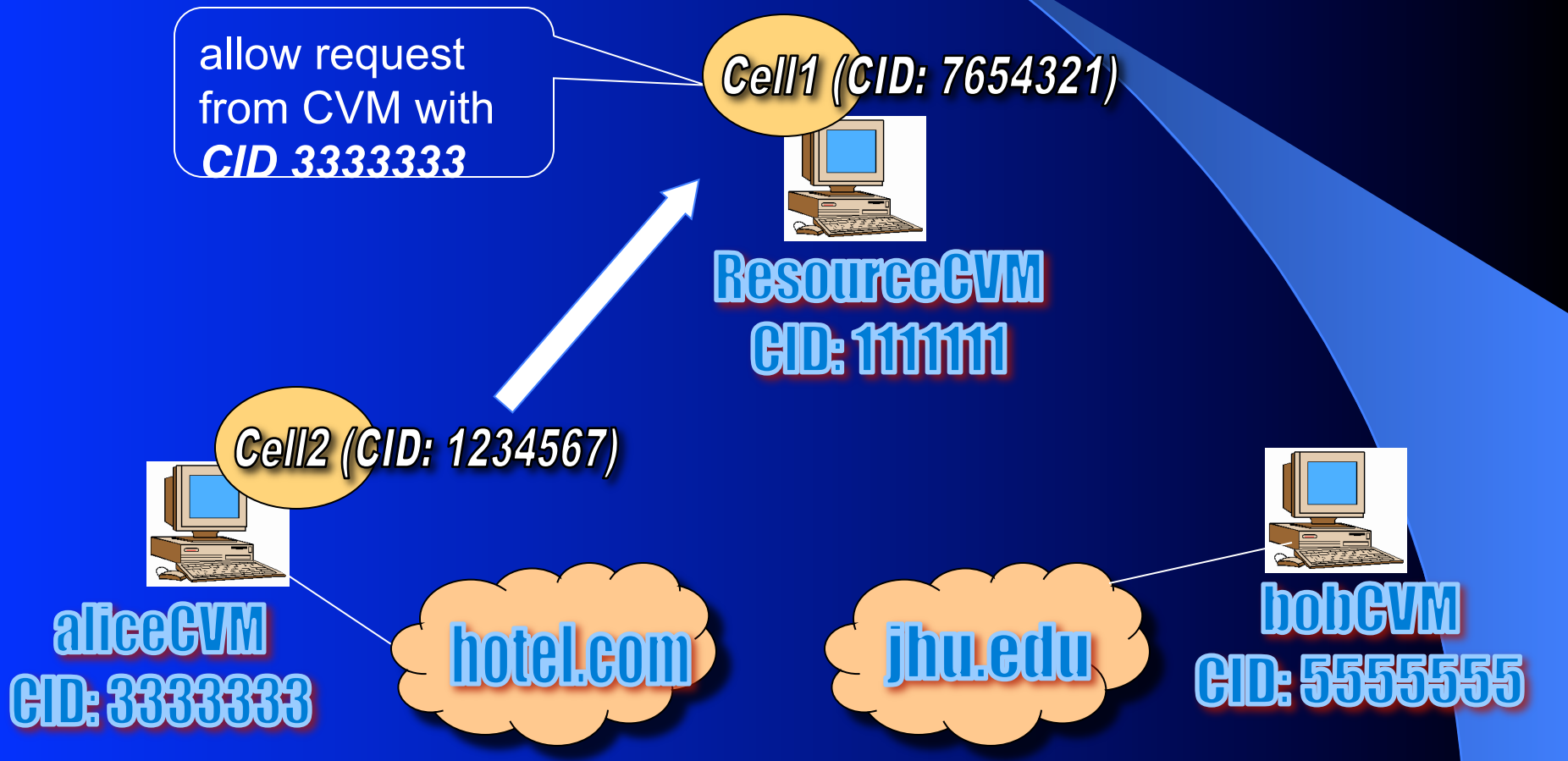
…Fails for Mobile Devices



Cell Security Infrastructure



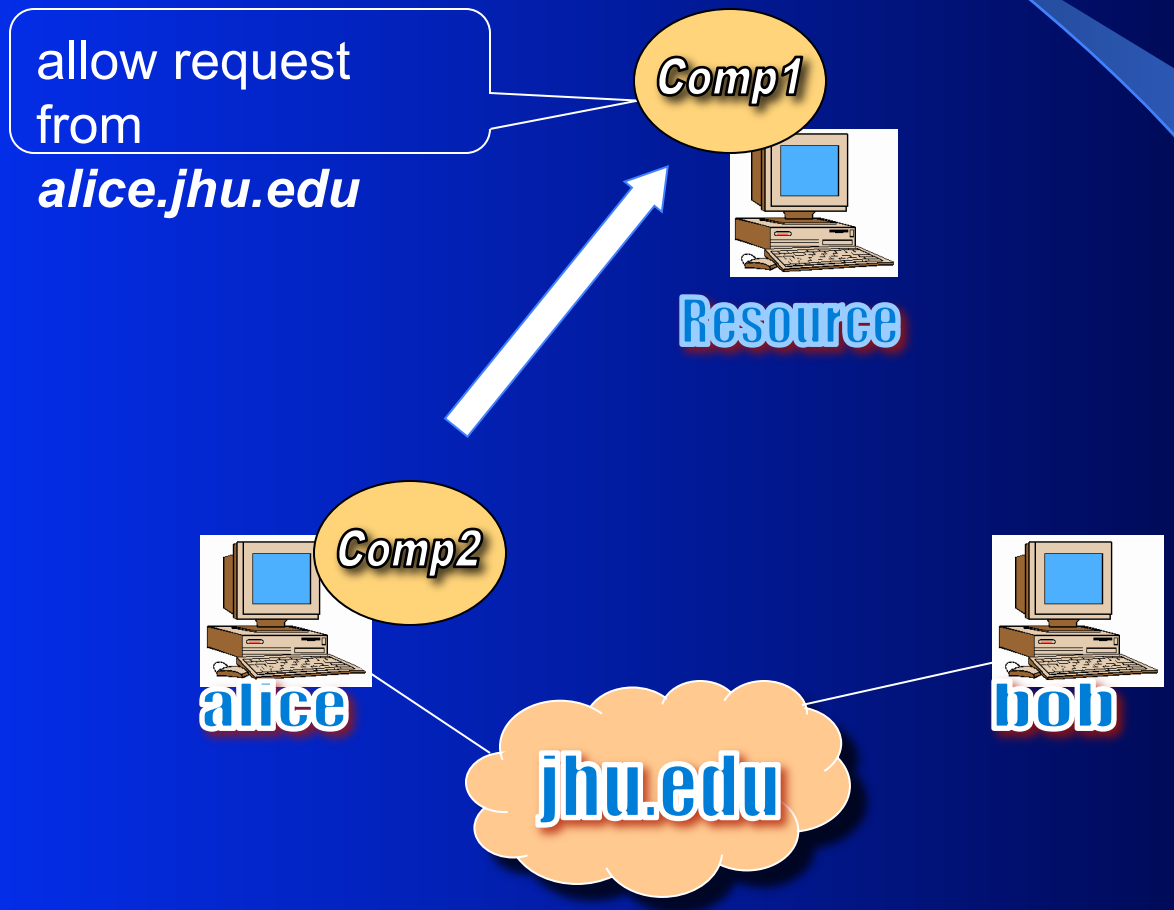
…Adapts Well with Mobile Devices



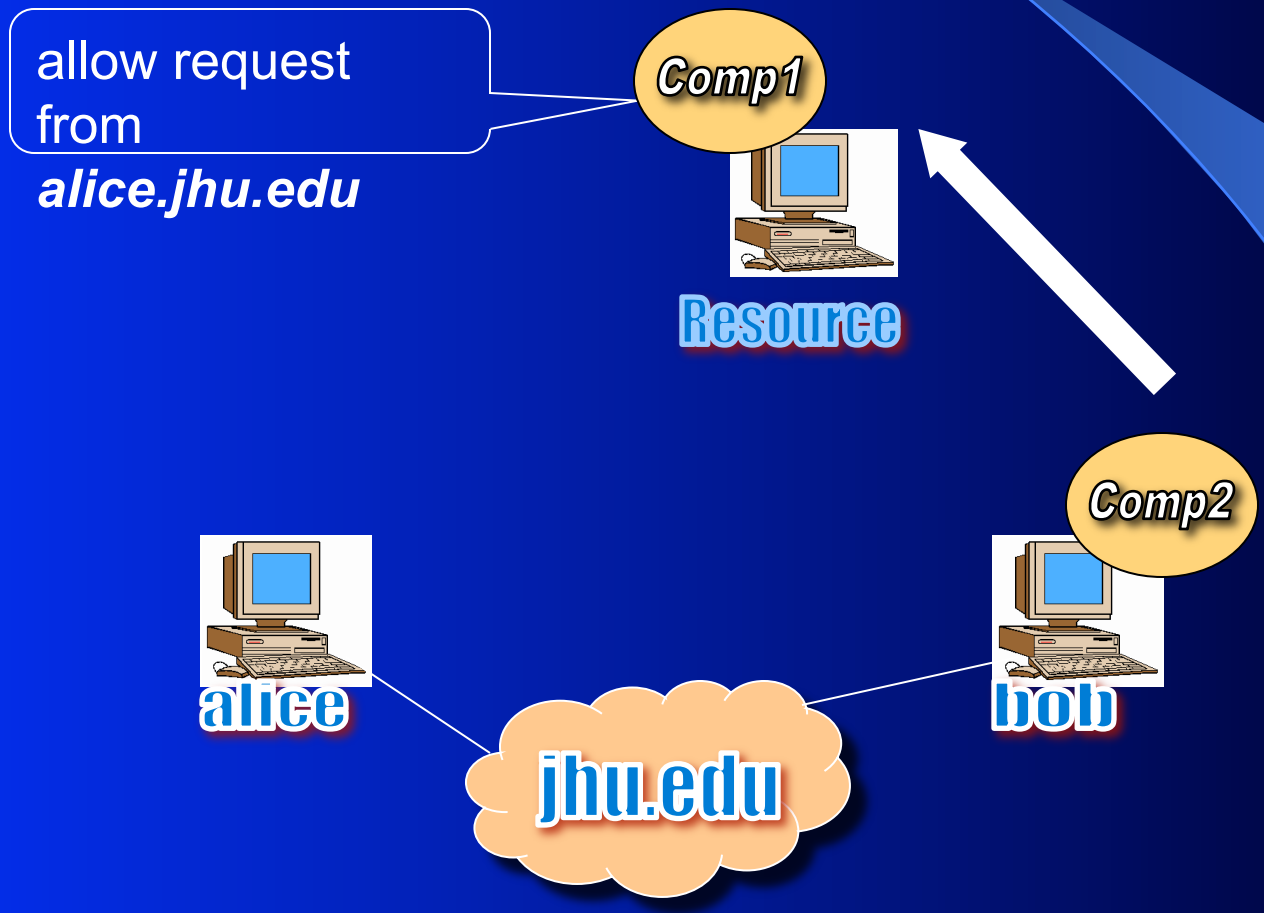
Extended Name

An extended name is a sequence of local names [n_1, n_2, …, n_k], where each n_{i+1} is a local name defined in the name space of the cell n_i.

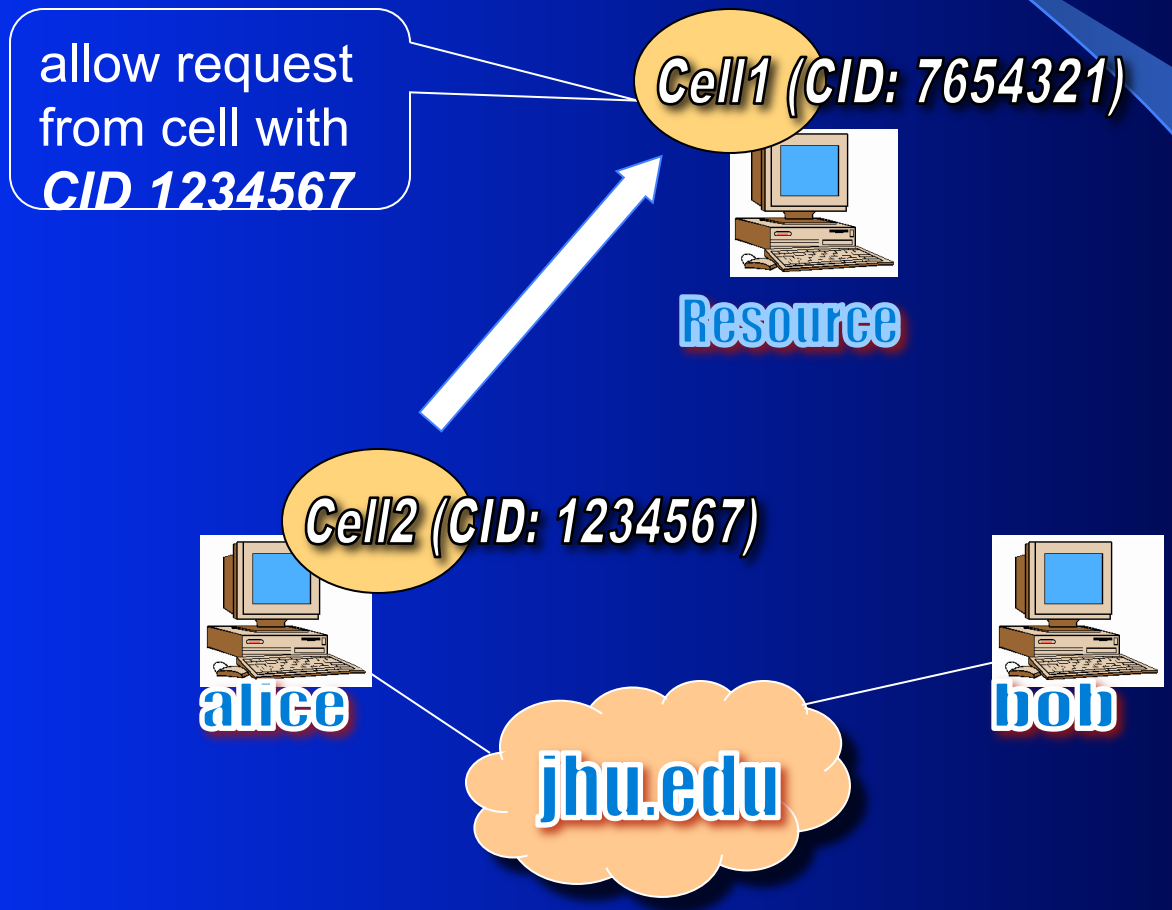
Example: Traditional Security Model



…Fails in Cell Migration



Example: Cell Security Infrastructure



…Adapts Well in Cell Migration