A Framework for Sensor Networks with Multiple Owners

Satyasree Muralidharan

Department of Electrical Engineering & Computer Science
Master's Thesis Defense
December 6, 2007

Committee
Dr. Victor S. Frost (Chair)
Dr. Gary J. Minden
Dr. Joseph B. Evans

Acknowledgment

- Defense Committee:
  - Dr.Victor Frost
  - Dr. Gary Minden
  - Dr. Joseph Evans
- SensorNet Team:
  - Ed Komp
  - Pradeep Mani, Andrew Boie, Daniel Fokum, Supriya Vasudevan, Jim Stevens

Outline

- Introduction
- Research Goals
- Multi-Owner Architecture
- Available Infrastructure
- Proposed Access Control Mechanism
- Prototype Implementation
- Testing and Results
- Conclusion
- Future work

- Introduction
- Research Goals
- Multi-Owner Architecture
- Available Infrastructure
- Proposed Access Control Mechanism
- Prototype Implementation
- Testing and Results
- Conclusion
- Future work

Introduction

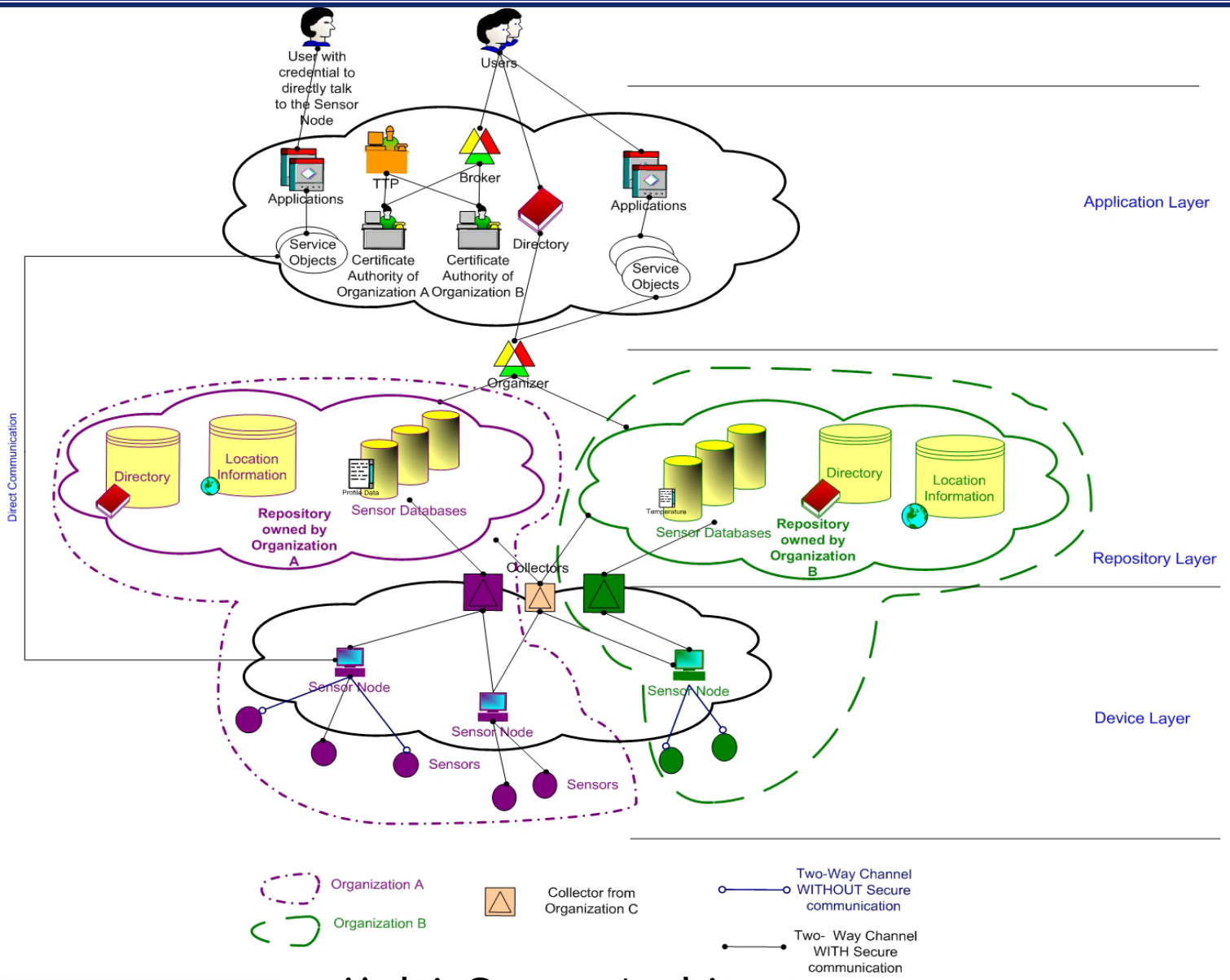
- Current sensor network technologies involve:
  - Assets owned and maintained by multiple disparate organizations
  - Huge pool of data generation and dissemination points
  - Providing secured distribution of data is critical
- Lack
  - well defined interfaces between different component layers
  - sophisticated, reliable, robust authentication and authorization mechanisms
- A unified architecture for sensor networks was proposed at ITTC to address many of these issues.

- Introduction
- Research Goals
- Multi-Owner Architecture
- Available Infrastructure
- Proposed Access Control Mechanism
- Prototype Implementation
- Testing and Results
- Conclusion
- Future work

Research Goals

- Providing assured and controlled access to data objects is central to the architecture.
- Some of the goals in realizing this requirement were:
  - To design an access control framework to ensure right information is provided to right people at right time
  - To design a framework to enable authentication and authorization mechanisms for users from multiple organizations
  - To address the issues involved in realizing the required security functions of privacy, integrity, authentication and authorization into a usable prototype

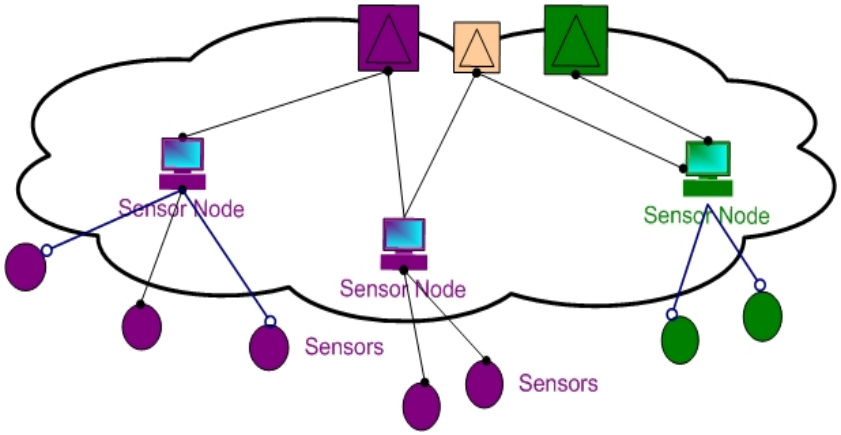
- Introduction
- Research Goals
- **Multi-Owner Architecture**
- Available Infrastructure
- Proposed Access Control Mechanism
- Prototype Implementation
- Testing and Results
- Conclusion
- Future work



Multi-Owner Architecture

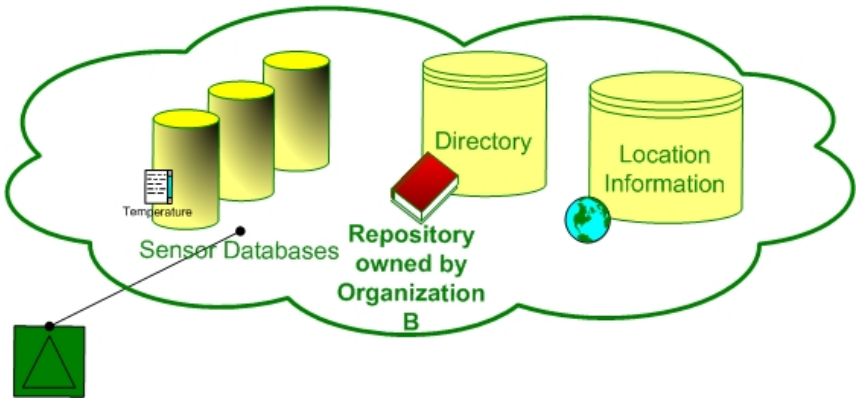
Multi-Owner Architecture - Device Layer

- Composed of all the physical sensor endpoints.
- Generation of sensor data.
- Consists of:
  - Sensors – hardware devices
  - Sensor nodes – computer that manages one/more sensors
  - Sensor services - programs that control the sensors attached to the node
  - Collectors - programs that collect data from these services and transport them to the repository layer for further processing.



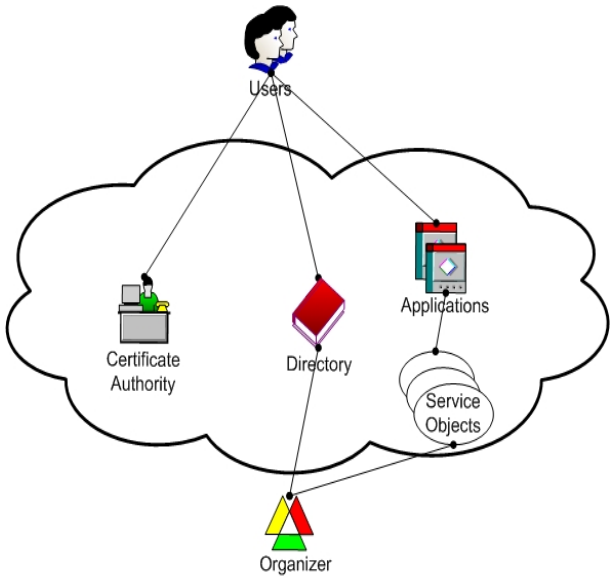
Multi-Owner Architecture - Repository Layer

- Storage of sensor information.
- Essentially composed of databases:
  - Sensor Databases that store and retrieve sensor data.
  - Infrastructural databases that store other information to support the system



Multi-Owner Architecture - Application Layer

- Provides a unified view of the various components of the architecture.
- Composed of:
  - Organizer – collects and transports data from the repository layer
  - Applications - programs that can be either used to talk to the Organizer to get the processed data or to the Sensor Service directly.
  - Certificate Authority - an entity that:
    - Signs the public keys of the users
    - creates credentials for users to talk to devices



Multi-Owner Architecture – Direct Communication

- User can talk directly to a device without having to pass through this layered architecture.
- An out-of-band communication.
- Example: A situation where the user takes control over the sensors of all organizations and may wish to control them without having to talk to the organizer or the collector.

- Introduction
- Research Goals
- Multi-Owner Architecture
- **Available Infrastructure**
- Proposed Access Control Mechanism
- Prototype Implementation
- Testing and Results
- Conclusion
- Future work

Available Infrastructure

- **Ambient Computing Environment** (ACE) , a system previously implemented at ITTC, provides an infrastructure for:
  - Authentication using TLS,
  - Encryption using AES,
  - Integrity using SHA1,
  - Authorization using KeyNote Trust Management.
- It has certain limitations when applied to multi-owner architecture.

Available Infrastructure - KeyNote

- Provides a simple language for describing and implementing security policies, trust relationships and digitally-signed credentials to control potentially dangerous actions over untrusted networks.
- Terminology:
  - Assertions - Describe the conditions under which a principal authorizes actions requested by other principal.
    - Policy - one or more unsigned assertions.
    - Credential: - a signed assertion. A credential can be securely transmitted over untrusted networks.
  - Action Attributes - (name, value) pairs, and the primary objects on which KeyNote assertions operate.

Available Infrastructure – Assertion Examples

POLICY:

KeyNote-version: 2
Comment: Policy assertion authorizing the administrator
Authorizer: POLICY
Licensees: "x509-base64:MIIEZzCCA.."
Conditions: (APP_DOMAIN == "ACE") -> _MAX_TRUST;

CREDENTIAL:

KeyNote-version: 2
Authorizer: "x509-base64:MIIEZzCCA9CgAw...LCSG0N2ICh"
Licensees: "x509-base64:MIIEZnb53...ighfkRT4523k"
Conditions: ((APP_DOMAIN == "ACE") &&
(time >= 1082390980610) &&
(time <= 1082390980628)) -> "true";
Signature: "sig-rsa-sha1-base64:Nt4+XIP...soP+mgjjTXWA=="

Available Infrastructure - Limitations when applied to Multi-Owner environment

Credential granularity

- Lowest level of granularity that ACE supports is method
- But, Multi-Owner architecture involves situations requiring credentials with different levels of specificity, ranging from parameter of a method to access of group of devices in a single credential.

Limited Set of Action attributes

- ACE cannot differentiate between different service objects and their instances.
- Grouping of devices and service objects results in enormous length of credential

Low level embedding

- Attributes in ACE are directly embedded at a low level in the program infrastructure
- Cannot externally control, add or delete the set of action attributes.

Scope of application

- ACE was primarily designed for single domain environment.

- Introduction
- Research Goals
- Multi-Owner Architecture
- Available Infrastructure
- **Proposed Access Control Mechanism**
- Prototype Implementation
- Testing and Results
- Conclusion
- Future work

Proposed Access Control Mechanism

- The access control mechanism for the architecture overcomes these limitations and meets the security requirements.
- Extension of action attribute set
  - The new set extends from parameter level to groups of services/devices.
  - External control of action attributes
- Group level authorization achieved by Role attribute
- Cross Domain authentication and authorization achieved by introducing TTP and Broker

Proposed Access Control Mechanism Core Attributes

Existing Set:

APP_DOMAIN
- Domain for which these credentials are used.

Method
- Java method name the client is attempting to execute.

Time
- Time of client's request to access the service.

Extended Set:

ServiceClassHierarchy
- Java class hierarchy path for the service performing the action.

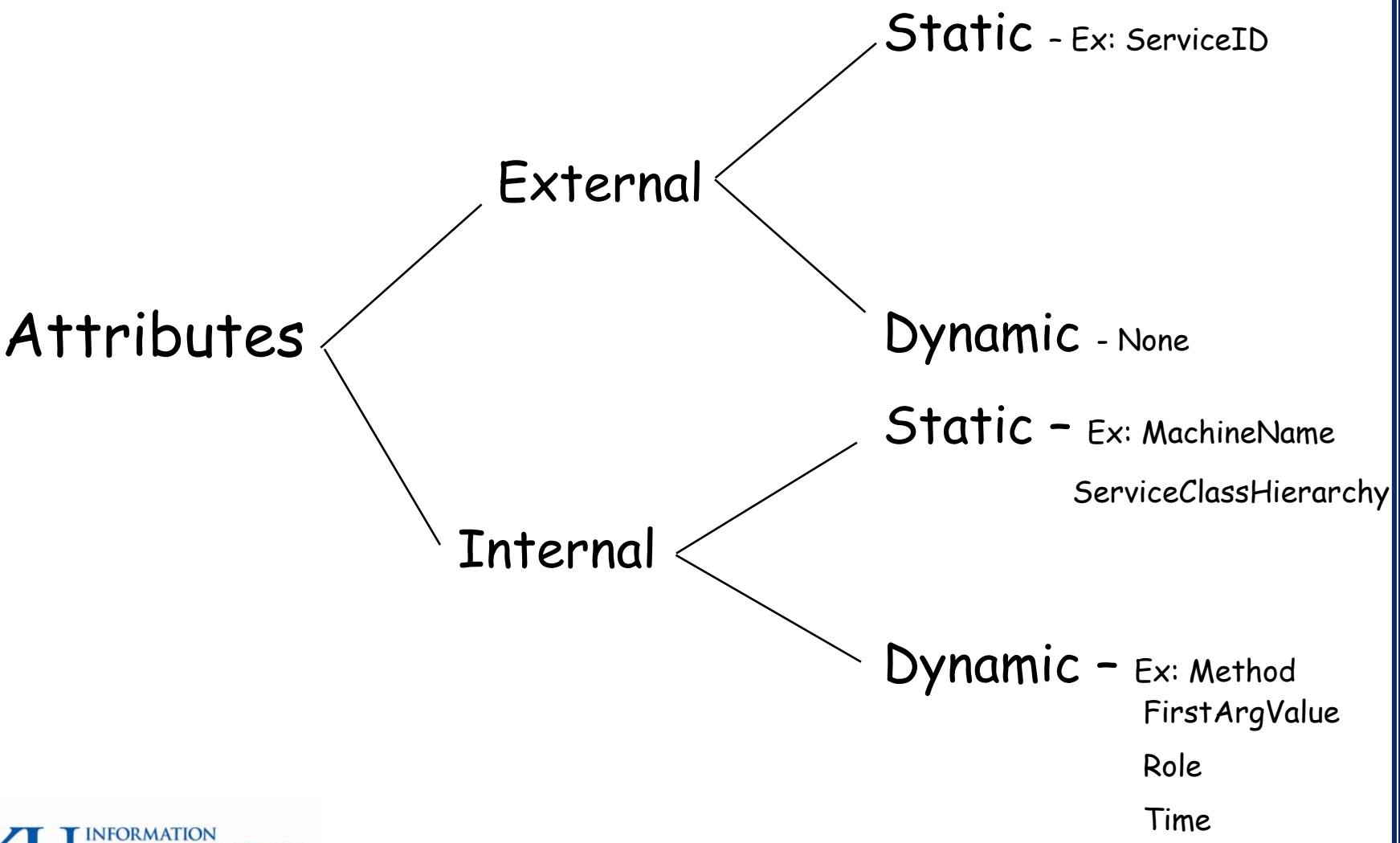
MachineName
- The name of the machine in which the service is running.

FirstArgValue
- Specialized attribute assigned the value of the first argument supplied to the method to be executed

Role
- Attribute identifies a general level of device control that the client must be authorized to perform an action.

ServiceID
- A unique identification for each instance of a service.

Proposed Access Control Mechanism
Classification of Action Attributes



Proposed Access Control Mechanism
Action Attribute File

- External Static attributes are placed in a file and read when the service starts
- A module to verify the syntax validity of the action attribute file
- Complete control by the Organization hosting the service.
- Organization can decide on the attributes names and values.

Proposed Access Control Mechanism
Role Attribute

1. Reader – one who can retrieve information from the device
   Example: getSensorReading().
2. Writer – one who can load information on to the device
   Example: setSensorProfile(newProfile).
3. Administrator – one who can perform (almost) any action on the device.
   Supersets the role of "Reader" and "Writer".
   Example: resetSensor().
4. No role attribute is acknowledged for the specialized action.

Proposed Access Control Mechanism
Role Attribute

- Every method will be associated to one of the four categories.
- Who decides this mapping?
  - Author of the service, on a method by method basis.
- Example:

  ((app_domain == "SensorNet") && (Provider == "ITTC") &&
  (Time <= "1151465644580" && Time >= "1161465647580") &&
  (ServiceID == "ChemicalSensor001") && (Role == "Reader")) → "allow";

  → The user has access to all methods that has "Reader" role provided by this service.
- Methods which DO NOT have a role attribute associated with them, requires explicitly mentioning of the method name in a user's credential.
- To have privileged methods that needs explicit access.
- Example:

  (ServiceClassHierarchy == "<full-hierarchy-spec>" &&
  Method == "shutdown") → "allow".

Proposed Access Control Mechanism
Advantages - Expressiveness

- Granular access to a particular service
- Example: 1

KeyNote-Version: 2

Authorizer: CA

Licensees: Alice

Conditions:

((app_domain == "SensorNet") && (Time <= "1151465644580" && Time >= "1161465647580") &&

(ServiceID == "TemperatureSensor350") && (MachineName == "sentinel.ittc.ku.edu") && (Method == "getSensorReading()") && (Role == "Reader")) → "allow";

Signature: "sig-rsa-sha1-base64:XQZopw.."

Proposed Access Control Mechanism Advantages - Expressiveness

- Access to different services in a single credential
- Example: 2

KeyNote-Version: 2
Authorizer: CA
Licensees: Alice
Conditions:
((app_domain == "SensorNet") &&
(Time <= "1151465644580" && Time >= "1161465647580") &&
((ServiceID == "ChemicalSensor001") && (Role == "Writer")) ||
((ServiceID == "SensorDatabase002") && (Role == "Reader"))
→ "allow";
Signature: "sig-rsa-sha1-base64:XQZopw.."

Proposed Access Control Mechanism Advantages - Expressiveness

- Access to group of services in a single credential
- Example: 3

  Conditions:

  ((app_domain == "SensorNet") &&

  (Time <= "1151465644580" && Time >= "161465647580") &&

  (ServiceClassHierarchy== ^.*Camera$ ") &&

  (Role == "Reader")) → "allow";

Proposed Access Control Mechanism Advantages- Flexibility

- Flexibility – Add/Delete a new attribute
- Example: 4
  - Add the Provider attribute to the action attribute file

Comment: An Organizer collects sensor data from two different providers and maintains a database of his own.

Conditions:

((app_domain == "SensorNet") && (Provider == "EECS") &&
(Time <= "1151465644580" && Time >= "1161465647580") &&
(ServiceID == "TemperatureSensorDatabase") && (Role == "Reader")) → "allow";
((app_domain == "SensorNet") && (Provider == "ITTC") &&
(Time <= "1151465644580" && Time >= "1161465647580") &&
(ServiceID == "ProfileDatabase") && (Role == "Reader")) → "allow";
((app_domain == "SensorNet") && (Provider == "KU") &&
(Time <= "1151465644580" && Time >= "1161465647580") &&
(ServiceID == "SensorDatabase") && (Role == "Writer")) → "allow";
Signature: "sig-rsa-sha1-base64:XQZopw.."

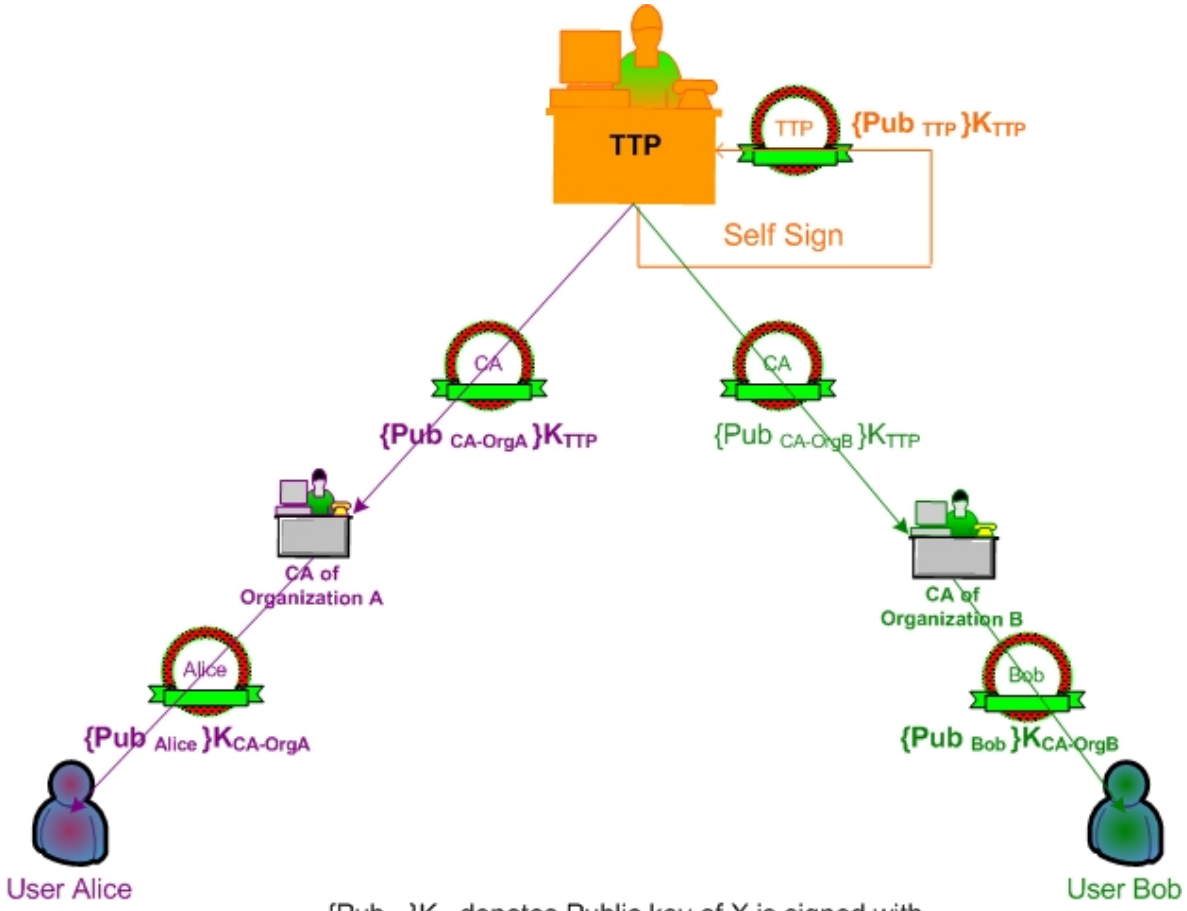
Proposed Access Control Mechanism
Cross Domain Authentication/Authorization

- Use intermediaries
  - Trusted Third Party (TTP)
    - Provide cross-domain authentication
  - Broker to provide authorization
    - Provide cross-domain authorization

Proposed Access Control Mechanism - TTP

- An entity who facilitates authentication between organizations that trust it.
- Issue public key certificates to CAs of the trusting organizations.
- Establish a Chain of Trust provided by the Public Key Infrastructure.
- Must maintain a trusted relationship with the participating organizations.

Proposed Access Control Mechanism
Chain of Trust

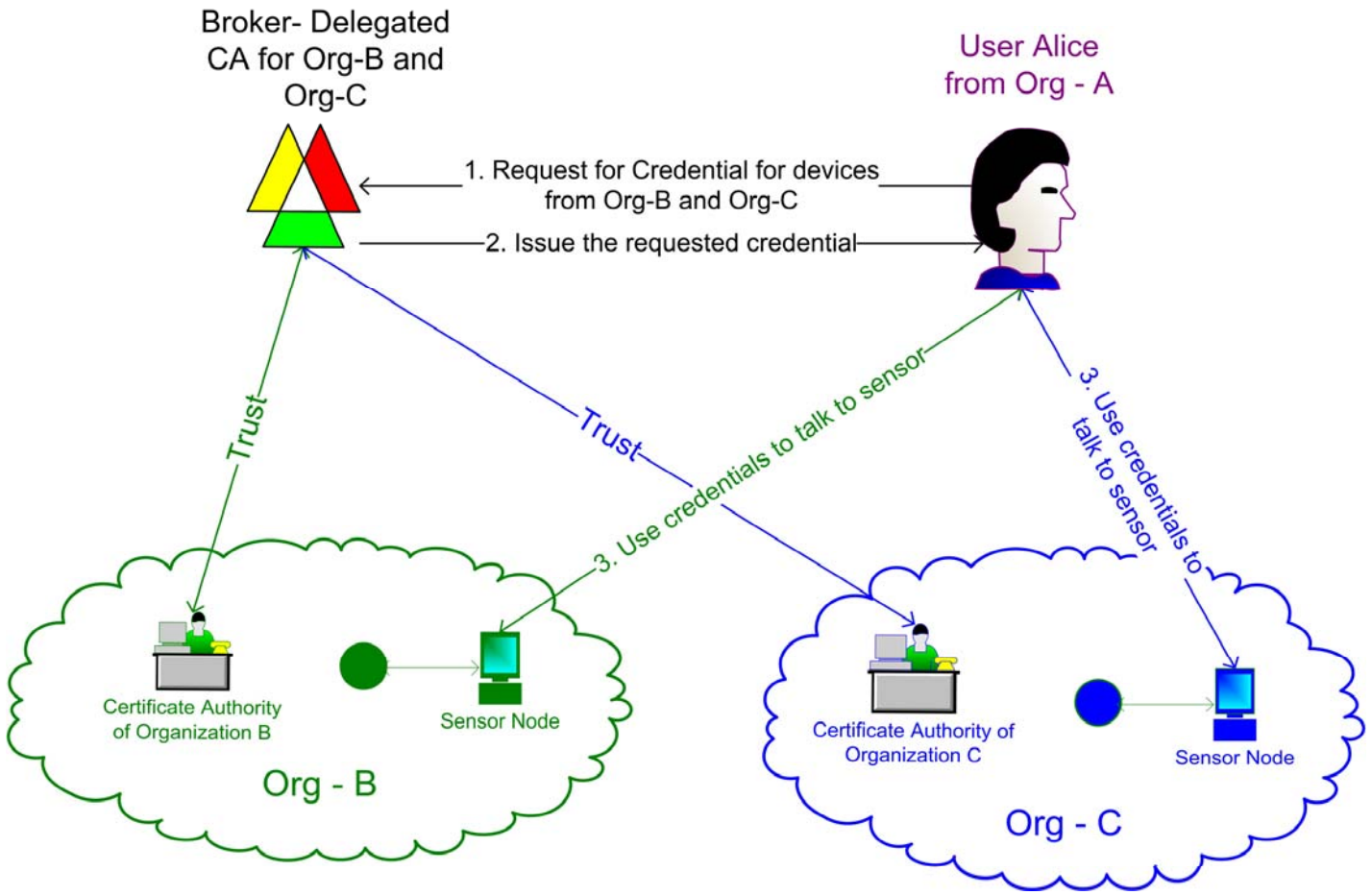


{Pub X}KY denotes Public key of X is signed with Private Key of Y

Proposed Access Control Mechanism - Broker

- An entity that can issue credentials on behalf of a CA.
- Act as the "delegated CA".
- Single point of contact.
- Must maintain a trusted relationship with the participating organizations.

Proposed Access Control Mechanism Authorization using Broker



Proposed Access Control Mechanism Advantages

- Scalable
  - Broker issues only one credential when the user requests for access to devices from multiple organizations
- Controllable
  - The organizations can decide the services that are advertised and used by the users of other organizations.
  - The CA of the organization issues credentials to the Broker only for these services, which can be delegated to users of other organization.
  - Organization has control on what services are external and internal to the organization.

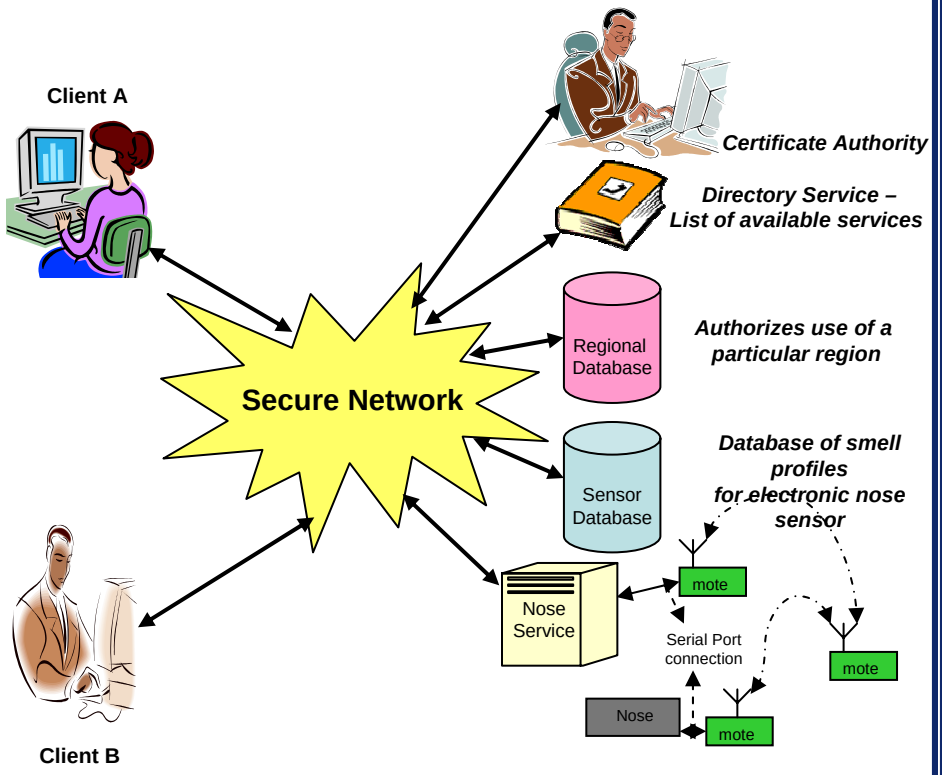
Proposed Access Control Mechanism Limitations

- Current delegation mechanism makes Broker a valid user of the system
- Broker delegates this authority as a valid user to other principals
- Some situations require separate "delegation" and "usage" rights
- Such a delegation right demands a provision of meta-credentials i.e., credential to provide other credentials
- KeyNote delegation system:
  - Does not have this provision.
  - Does not have the provision to check the validity of the credential during distribution time
  - Faulty credentials are captured during runtime.

- Introduction
- Research Goals
- Multi-Owner Architecture
- Security Requirements
- Available Infrastructure
- Proposed Access Control Mechanism
- **Prototype Implementation**
- Testing and Results
- Conclusion
- Future work

Prototype Implementation

- Device Layer
  - Cyranose
  - Nose Service
    - Load Profile : Load a new smell profile from the sensor database to the nose
    - Start Identification : Start a new identification in the nose
    - Fetch Results: Retrieve the results of the last identification from the nose.
  - IEEE 1451 Compliant
- Repository Layer
  - Profile Database Service
  - Directory Service
  - Regional Database
- Application Layer
  - Nose Client Program
  - Nose Client Program is developed to comply with Model View Controller Paradigm

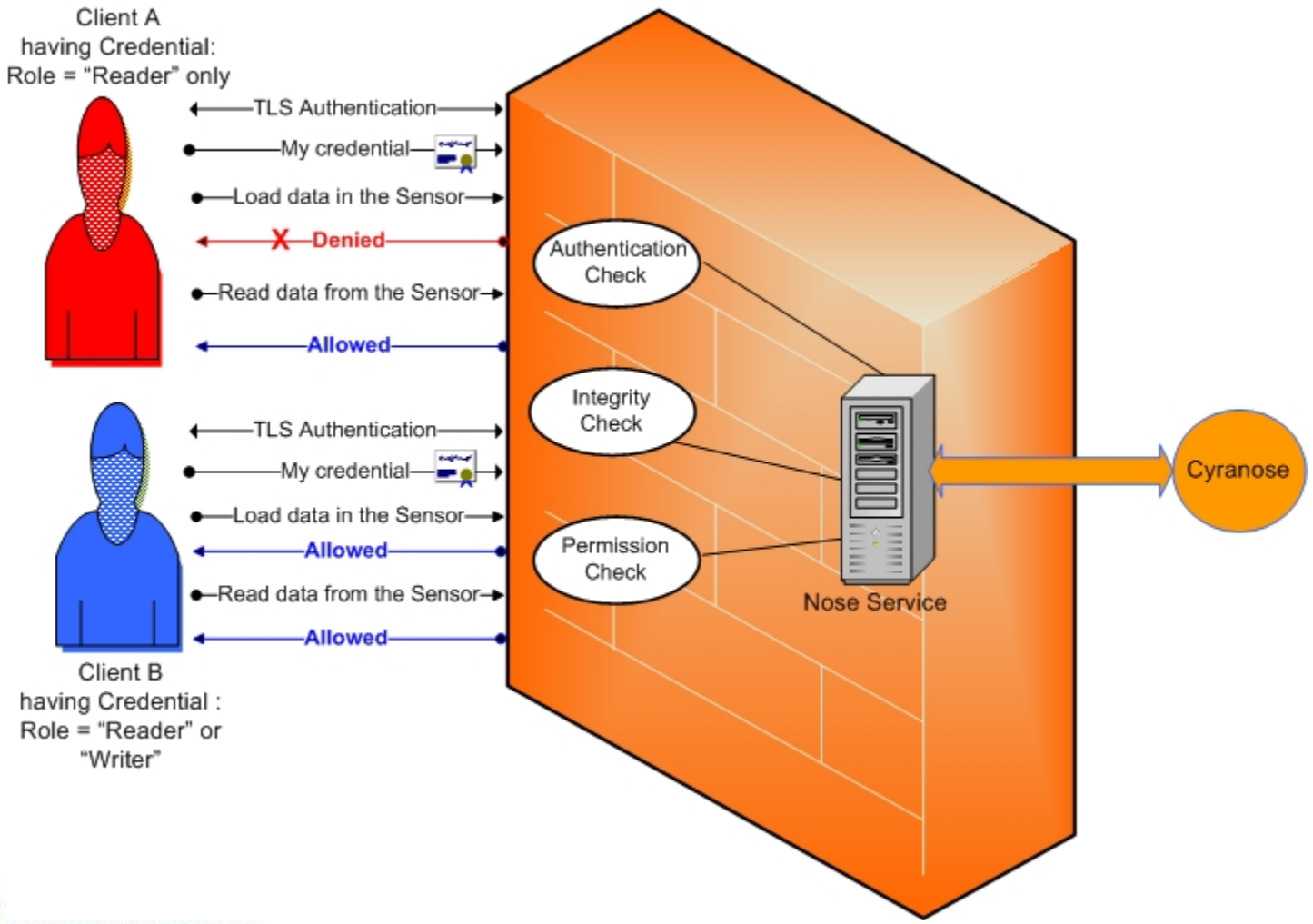


- Introduction
- Research Goals
- Multi-Owner Architecture
- Available Infrastructure
- Proposed Access Control Mechanism
- Prototype Implementation
- **Testing and Results**
- Conclusion
- Future work

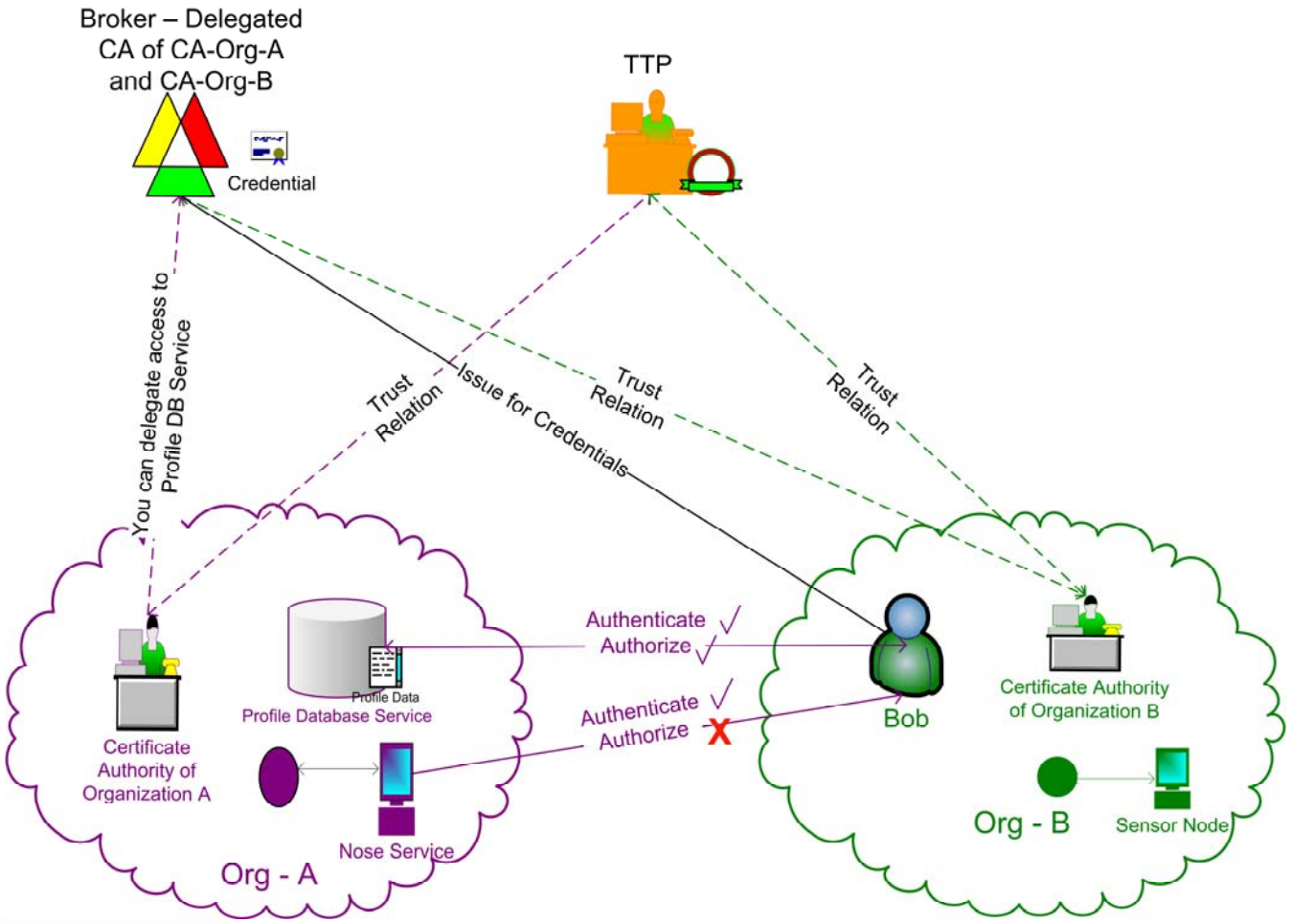
Testing

- Extensions to Action Attribute Set
  - A set of credentials was generated from a broader level to a very narrow level
    - Parameter level to test specificity
    - Group of services to test Role Based authorization
  - Variety of test cases with each one reflecting specific scenario.
- Cross-Domain authentication
  - Tests to verify "Chain of Trust" between organizations to authenticate different users from different organizations.
- Cross-Domain authorization
  - Tests to verify authorization using Broker with delegation using KeyNote.

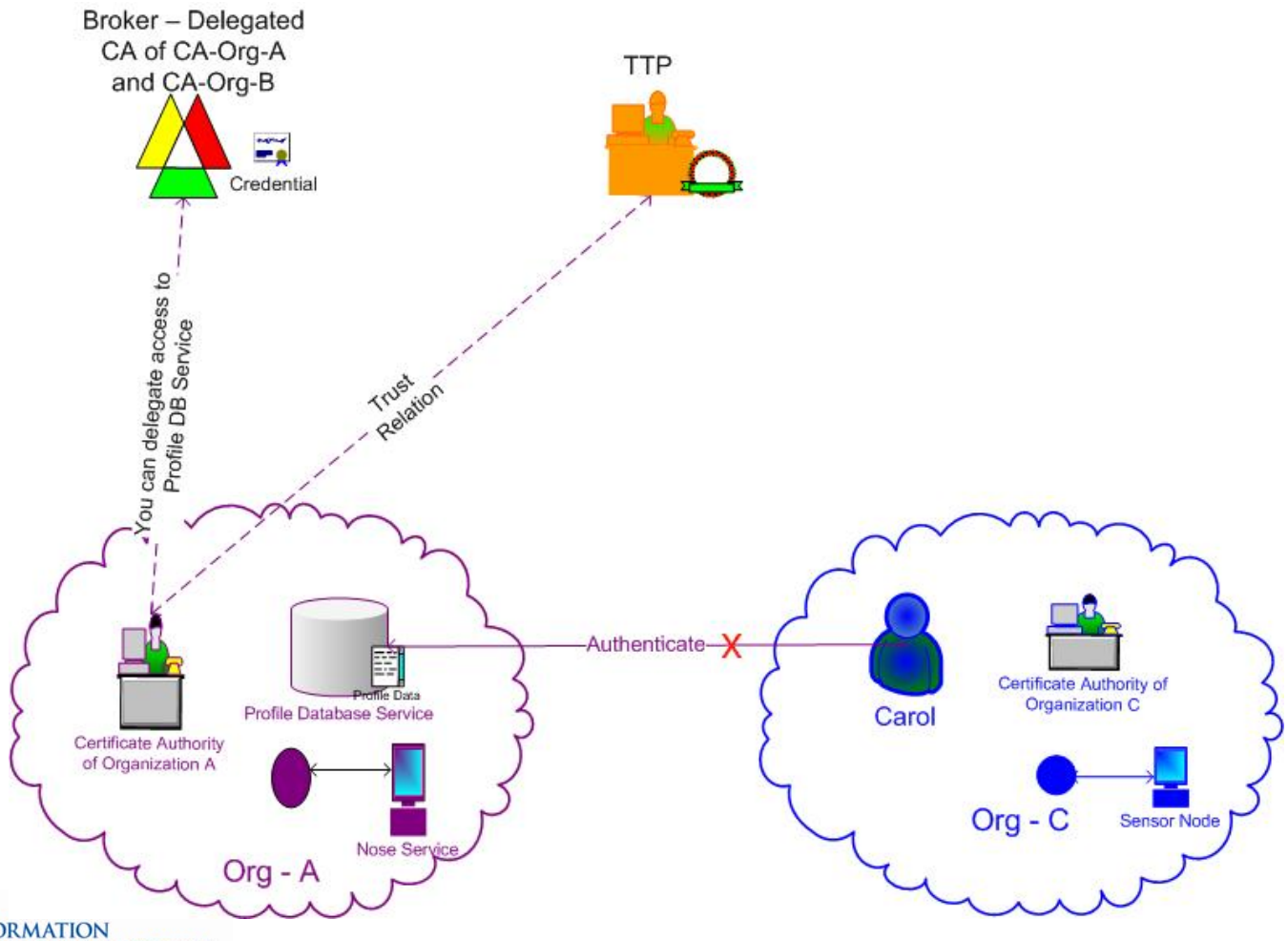
Testing and Results



Testing and Results



Testing and Results



- Introduction
- Research Goals
- Multi-Owner Architecture
- Available Infrastructure
- Proposed Access Control Mechanism
- Prototype Implementation
- Testing and Results
- **Conclusion**
- Future work

Conclusion

- The extended action attribute set provided the granularity required for Multi-Owner environment.
- The developed access control framework provides for a flexible security and policy model to support Multi-Owner environment.
- This framework has been successfully designed and tested using a prototype implementation.

- Introduction
- Research Goals
- Multi-Owner Architecture
- Available Infrastructure
- Proposed Access Control Mechanism
- Prototype Implementation
- Testing and Results
- Conclusion
- Future work

Future work

- Several manually implemented components of the prototype can be automated.
- Action attribute set can be extended to include deployment specific attributes.
- Communication using can be re-implemented using a method like XML-RPC to make the system accessible to all programming languages and across all operating systems.
- Experiments to measure performance, scalability and deployment issues to complement prototype demonstrations could be designed.

Questions or Donuts?

Choose One