Unified Architecture for SensorNet with Multiple Owners

Electrical Engineering & Computer Science
Information & Telecommunication Technology Center
KU SensorNet Overview

• System Integration
  • Develop the technology, standards, and technical requirements for an integrated national warning and alert system from transducer through command and control

• Operational Integration
  • Incident discovery, awareness, and response capability integrated across multiple local, state, regional, and national responders in a secure and robust manner

• Technology Integration
  • Demonstrated network infrastructure suitable for rapid deployment and real-time collection, processing, and dissemination of multiple information types (e.g. telemetry, audio/video, data access)
KU SensorNet Tasks

• Architecture
  • Assured access and control over sensor nodes
  • Data architecture considering multiple owners
  • Applications support

• Integration with sensors and databases
• Distributed real-time control
• Implementation and evaluation
• Identification of new applications
KU SensorNet Architecture

- Functional description of resources
- Map SensorNet functions onto hardware and software
- Comprehensive network infrastructure
- Multiple owners and operators of infrastructure
KU SensorNet Functional Implementation

Users

Dissemination Network

Applications

Portals

Resource Network

Data Archive

Collectors

Collection Network

Sensors

Control Node

Data Archive

Control Node
KU SensorNet Functional Implementation

Dissemination Network

Resource Network

Collection Network

Data Archive

Collectors

Control Node

Sensors

Applications

Portals

Users

Applications
KU SensorNet Multiple Owners

Dissemination Network

Resource Network

Collection Network

Users
Applications
Portals
Data Archive
Collectors
Control Node
Sensors

Data Archive

Data Archive

Control Node
Distributed Real-Time Control

- Distributed sensor-actuator integration
- Control loops of collection-analysis-action
- Control loop stability influenced by end-to-end communication time
- Performance data collection and analysis
Sensors
KU SensorNet Data Collection
Range of Sensors and Network Architectures

Passive Tags
- Low cost per sensor
- Higher reader cost
- Very limited capabilities

Embedded Processors
- Medium cost per sensor
- Constrained capabilities

PC-base SensorNodes
- High-cost per node
- Complete integration
- Significant capabilities
Simple Wireless Networked Sensors

- Implemented multi-hop Mote network, connecting “E-Nose” to “Nose Server”
- Load “Nose profiles” across Mote network, read “Nose profiles” from nose, observe sample detection
- Extended Mote protocol for multiple hop, fragmented, acknowledged datagrams
- Integrated Mote network with IEEE 1451 NCAP server
- Enhanced SensorNet architecture with 1451 interface and extended network service
Databases
KU SensorNet Databases

Dissemination Network

Resource Network

Collection Network

Data Archive

Collectors

Control Node

Sensors

Applications

Portals

Users
IEEE 1451 and DB Integration

- Multiple interfaces between applications and servers
- Multiple authentication mechanisms
- Multiple authorization descriptions

==> Complexity
Typical Database Service

• Database holds base library and collected sensor data
• Database server provides search, sorting, collating, etc. functions
• Database server is accessed over network or from applications
• Applications may provide additional services
Database Security Enhancements

- Prevent direct access to database server
- Replace application and force all access through Service Daemon
- Service Daemon capabilities described in following slides
Sensor Node Security Enhancements

- Prevent direct access to sensor server
- Replace application and force all access through Service Daemon
- Service Daemon capabilities described in following slides
KU SensorNet Multiple Owners

Collection Network

Resource Network

Dissemination Network

Applications

Data Archive

Sensors

Collectors

Portals

Users

Control Node
Keynote Authentication/Authorization

- Application independent
  - Developers define application specific attributes, credentials, and policies => extensible
  - Designed to express and evaluate policies and delegations in PKI-type applications
- Service providers create credentials and distributed them to users
- Users need a credential to access a service
  - A credential identifies the owner and authorizes the owner to invoke certain service functions
  - Users must protect their credentials
Example Credential (Reader)

keynote-version: 2
comment: Credential to talk to ServiceDir, RoomDatabase and SensorNetDatabase Service. Licensee has "Reader" role on the DatabaseService

authorizer: "x509-base64:MIID+……”
local-constants: KEY1 = "x509-base64:MIIEBjCCA2+……”
licensees: KEY1

conditions:

((APP_DOMAIN == "SensorNet") && (Time >= "1185570284258") && (Time <= "1188248640000") && (ServiceID == "NicholsHall_ServiceDir") && (Role == "Reader")) -> "allow"; #ServiceDir

((APP_DOMAIN == "SensorNet") && (Time >= "1185570284258") && (Time <= "1188622740000") && (ServiceID == "NicholsHall_RoomDatabase") && (Method == "getRoom")) -> "allow"; #RoomDB

((APP_DOMAIN == "SensorNet") && (Time >= "1185570284258") && (Time <= "1188622740000") && (ServiceID == "NicholsHall_SensorDatabase010") && (Role == "Reader")) -> "allow"; #SensorNetDBService

signature: "sig-rsa-sha1-base64:tfj5RHa9L6B7i8……”
Example Credential (Reader/Writer)

keynote-version: 2
comment: Credential to talk to ServiceDir, RoomDatabase and SensorNetDatabase Service. Licensee has "Reader" role on the DatabaseService

authorizer: "x509-base64:MIID+……"
local-constants: KEY1 = "x509-base64:MIIEBjCCA2+……"
licensees: KEY1

conditions:
   ((APP_DOMAIN == "SensorNet") && (Time >= "1185570284258") && (Time <= "1188248640000") && (ServiceID == "NicholsHall_ServiceDir") && (Role == "Reader")) -> "allow"; #ServiceDir
   ((APP_DOMAIN == "SensorNet") && (Time >= "1185570284258") && (Time <= "1188622740000") && (ServiceID == "NicholsHall_RoomDatabase") && (Method == "getRoom")) -> "allow"; #RoomDB
   ((APP_DOMAIN == "SensorNet") && (Time >= "1185570284258") && (Time <= "1188622740000") && (ServiceID == "NicholsHall_SensorDatabase010") && ((Role == "Reader") || (Role == "Writer"))) -> "allow"; #SensorNetDBService

signature: "sig-rsa-sha1-base64:tfj5RHa9L6B7i8……"
Real-time Control Across Networks
Evaluation of Distributed Control Loops

- Distributed control loop application components cross network components
- *Datastreams* collects data at application and OS levels
- Global time line built using clock synchronization and local clock values
Experimental Design

- NISTNet software is used to introduce delays between the video server machine and the processing node(s).
- Datastreams software used to construct global timeline with < 10us accuracy.
- Various sub-intervals of the control loop measured, with graphs automatically generated.
- Following results were measured with a 20ms delay introduced by NISTNet.
Aggregate Results

• NETSPEC Software automates execution of experiments under different parameter settings
• Aggregate graphs show average interval time, bounded by 1 standard deviation
• Following graphs show average intervals as a function of NISTNet delay
Real-time Measurement Summary

- Clock synchronization kernel patch ensures global timeline accuracy within 30 microseconds
- Instrumented Linux network stack traces packets through kernel
- Collected data is easily filtered and converted into visual representations
- Automated execution of distributed experiments and post-processing gathered data
- Extensible Python-based post-processing framework
KU SensorNet

Unified Architecture for SensorNet with Multiple Owners