UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

Design of Policy Engine to Prevent Malware Propagation in Advanced Metering Infrastructure

Younghee Park
Computer Engineering Department
San Jose State University

illinois.edu

SAN JOSÉ STATE UNIVERSITY

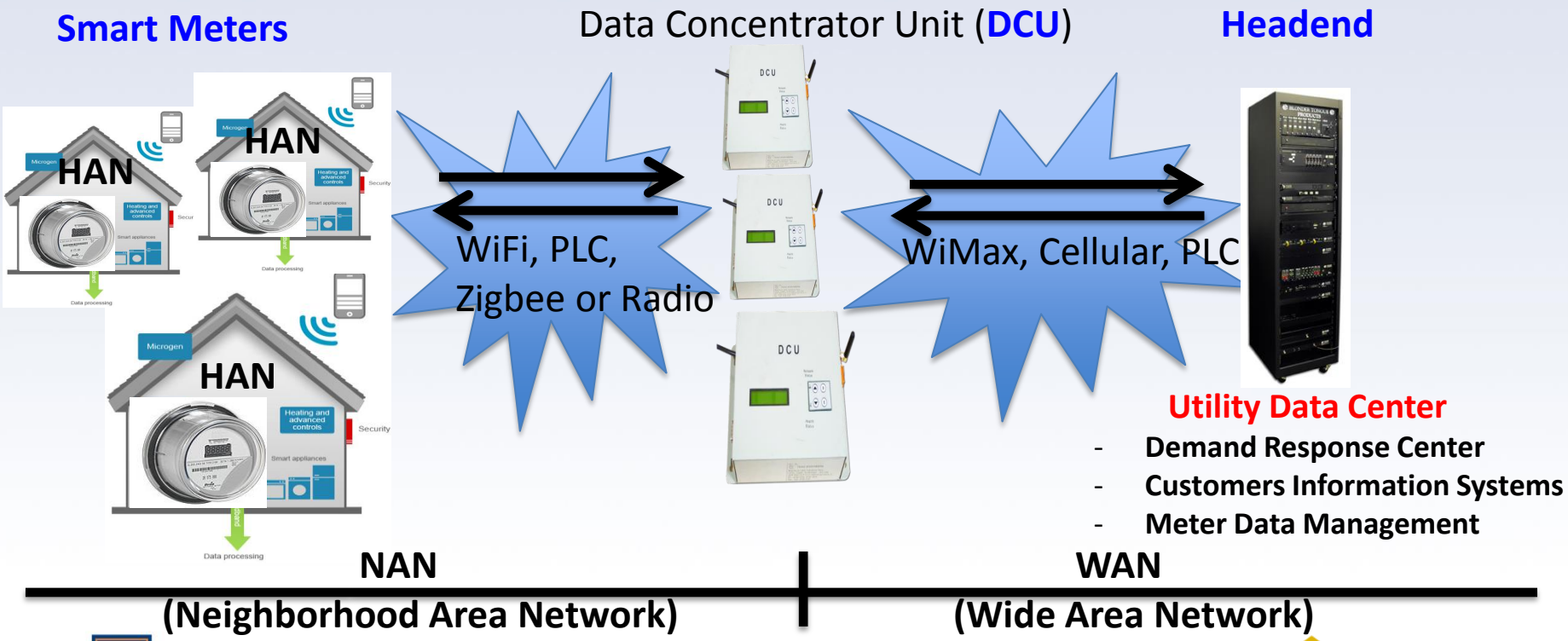
Introduction

illinois.edu

SAN JOSÉ STATE UNIVERSITY

Introduction

- Advanced Metering Infrastructure (AMI)



Introduction

- Goals of Advanced Metering Infrastructure (AMI)
  - Reliability
    - Smart metering
    - Automated real-time monitoring
  - Efficiency
    - Demand response and control
    - Active energy management
    - Remote connect/disconnect
  - Security
    - Improved by monitoring and reliability

Cyber Attacks in AMI

- Network availability attack
  - Overwhelm the communication and computational resources
- Data attacks
  - Insert, alter or delete data in the network traffic
- Privacy attacks
  - Learn users' private information by analyzing electricity usage data
- Device attacks
  - Compromised or control a device

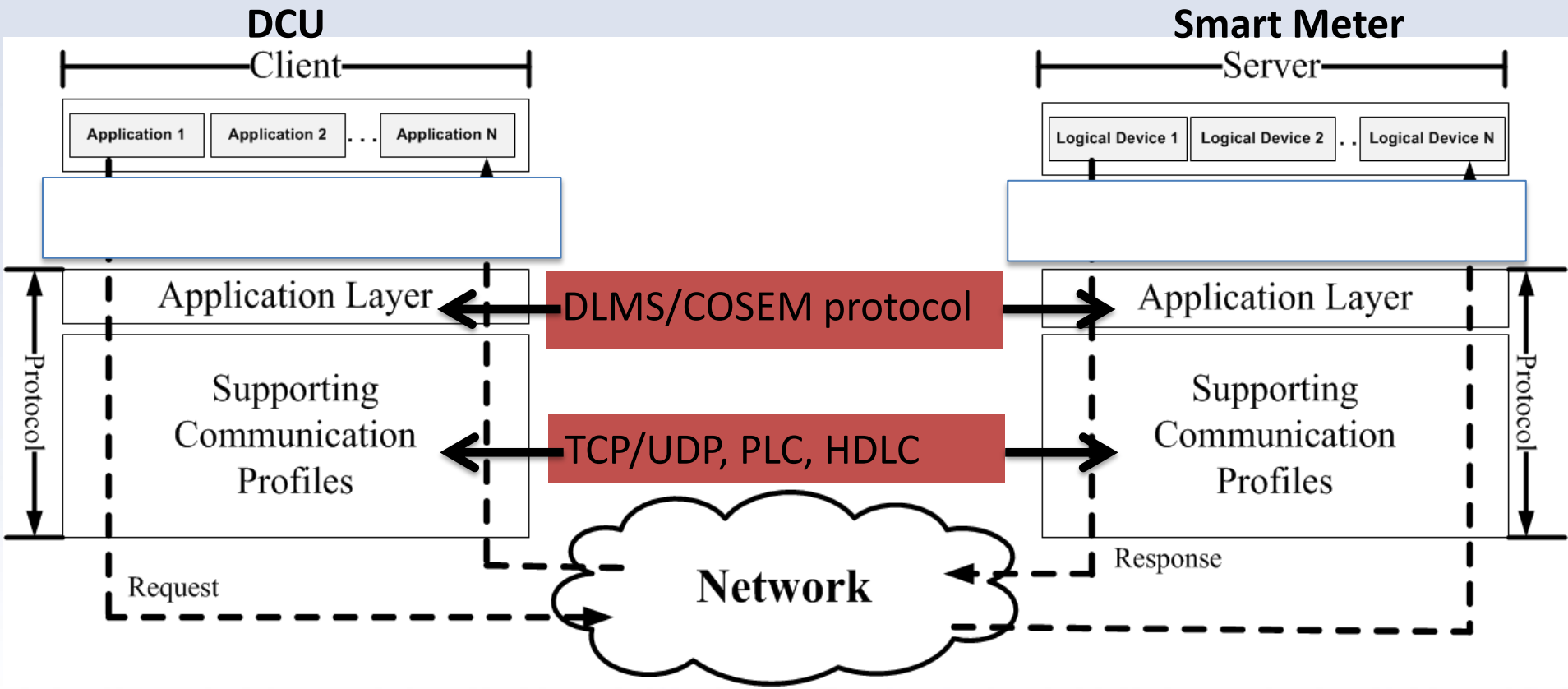
Cyber Attacks in AMI

- Malware (Malicious Software)
  - Stuxnet, Auora worm in industrial systems
  - Warning from Blackhat, McAfee
  - Control systems and disrupt operations
  - Launch malicious commands
    - Blackout, remote disconnects
  - Infect other systems in AMI

Policy Engine

- What is Policy Engine?
  - A set of policy rules
  - Detection/Prevention of Malware carried by an application layer protocol
    - Malware propagation during communications
  - Monitor bi-directional communications
    - Client request (DCU, Headend)
    - Server response (Smart meters)

Policy Engine



illinois.edu

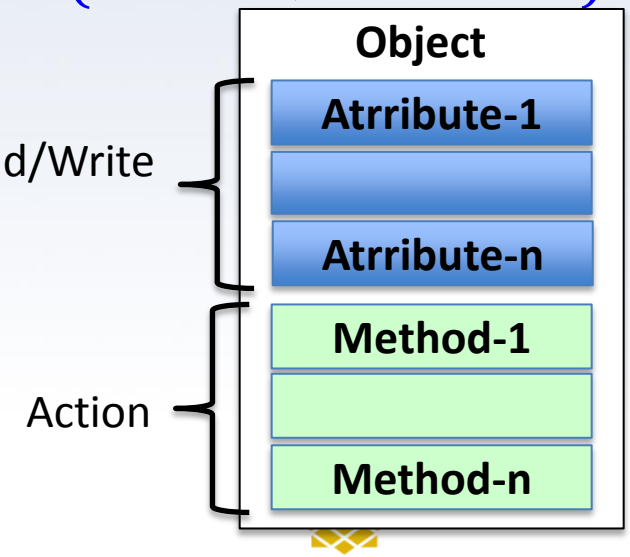
SAN JOSÉ STATE UNIVERSITY

DLMS/COSEM Protocol

- An standard application protocol for AMI
- DLMS (Device Language Message Specification)
  - define objects found in meters
- COSEM (Companion Specification for Energy Metering)
  - Define communication between meters and devices needing meter information

DLMS/COSEM Protocol

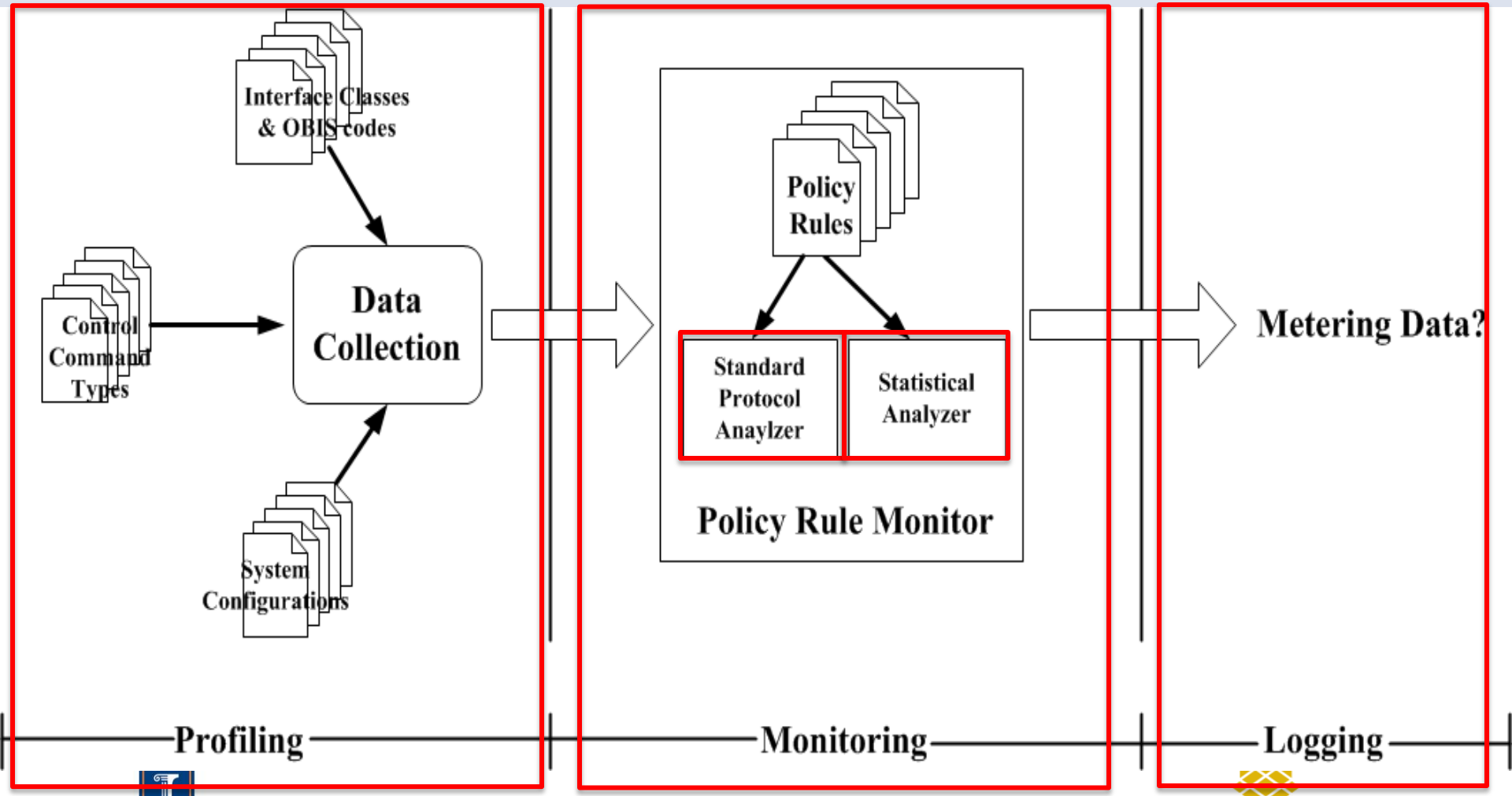
- Interface classes (Class ID)
  - All meter data as objects
  - Object Identification system (OBIS)
    - OBIS code is called a logical name (1st attribute value)
    - [EX] Reading energy value
      - **Register (class_id : 3)**
      - **Obis code is "1.1.1.8.0.255"**
      - Reading second attribute value
  - 70 standard interface classes



Motivation for Policy Engine

- Compromised applications or devices
- Propagate malware by hiding itself into DLMS/COSEM traffic
  - Violation of Packet formats
    - Specification of DLMS/COSEM application layer
  - No metering data
  - Containing malware features

System Architecture of Policy Engine



illinois.edu

SAN JOSÉ STATE UNIVERSITY

System Architecture of Policy Engine

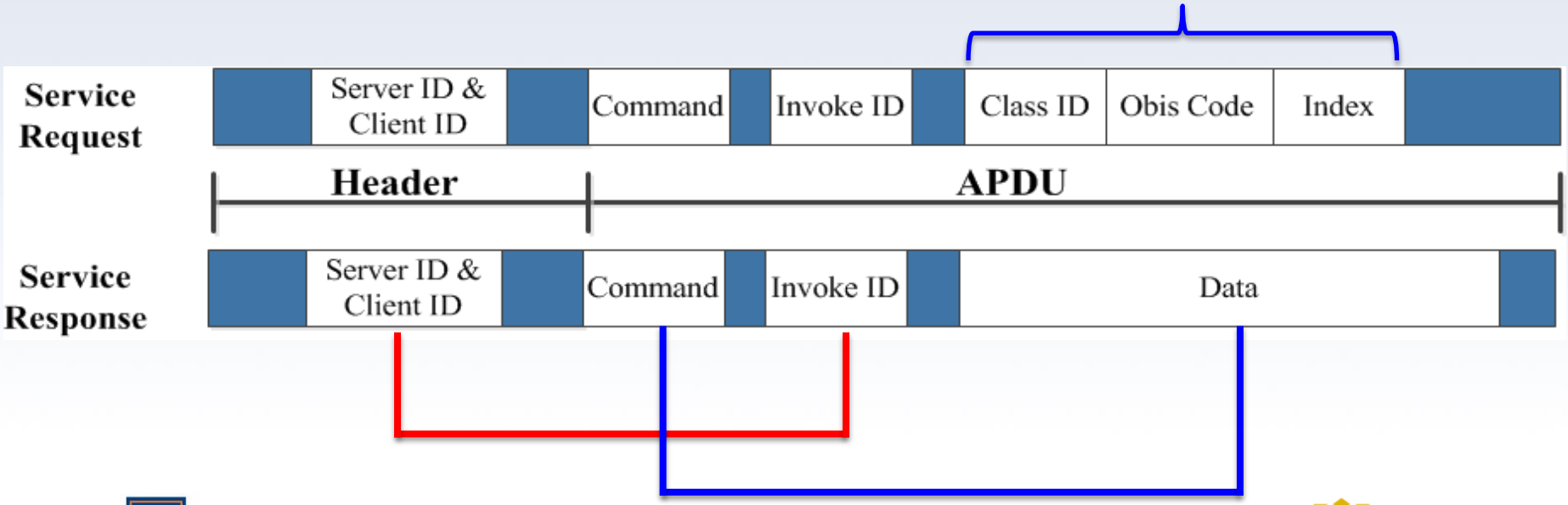
- Data collection system
  - OBIS, Command types, system configurations
- Policy Rule Monitoring system
  - Policy rules from the standard protocol
  - Policy rules from two statistical features
- Logging system
  - Recording malicious data

Standard Protocol Analysis

- DLMS/COSEM protocol
  - It creates application data
    - APDU (application protocol data unit)
  - Policy rules
    - Syntactic policy rules
    - Semantic policy rules
    - Access policy rules
    - Communication policy rules

Standard Protocol Analysis

- DLMS/COSEM Protocol
  - Syntactic/Semantic/Access/Communications
    - Malware overwrite the APDU



Statistical Analysis

- Two features of malware
  - Packed or encrypted binaries are around 90%
  - Executable format
- Smart meters
  - Low-computational power
  - Limited bandwidth and resourses
- Statistical analysis tools
  - Entropy analysis
  - N-gram analysis

Statistical Analysis

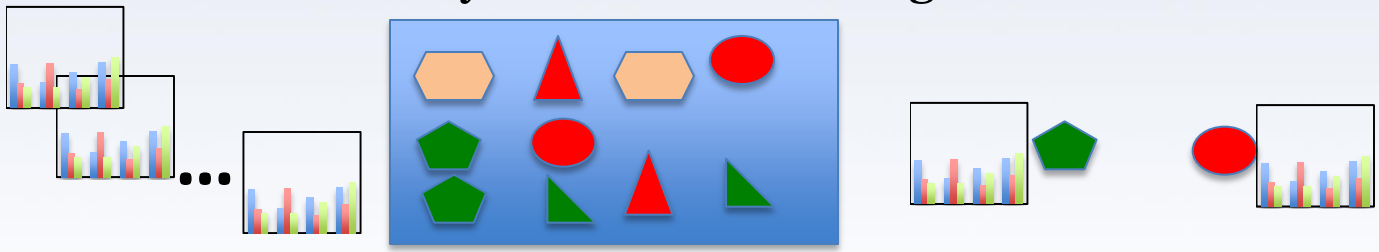
- Entropy analysis
  - Bintropy
    - Shannon entropy
    - Detection method for packing and encrypted binaries
    - Packed and encrypted binaries have greater than 6.67 entropy
  - Metering data have low entropy less than 6.
    - Evaluating the frequency of each byte in fixed block size

Statistical Analysis

- N-gram analysis
  - FilePrint
    - Each file type has its own distinct 1-gram distribution
  - 1-gram distribution of metering data is different from the one of malware
  - Measuring dissimilarity(distance) between two distributions
    - Manhattan distance and Mahalanobis distance

Statistical Analysis

- N-gram analysis
  - Trained Model for malware
    - K-mean clustering & Manhattan distance
    - EX) 10 malware files, K=2 (2 Trained models)
      - Pick 2 random files -> evaluate distributions -> update distribution by K-mean clustering and manhattan distance

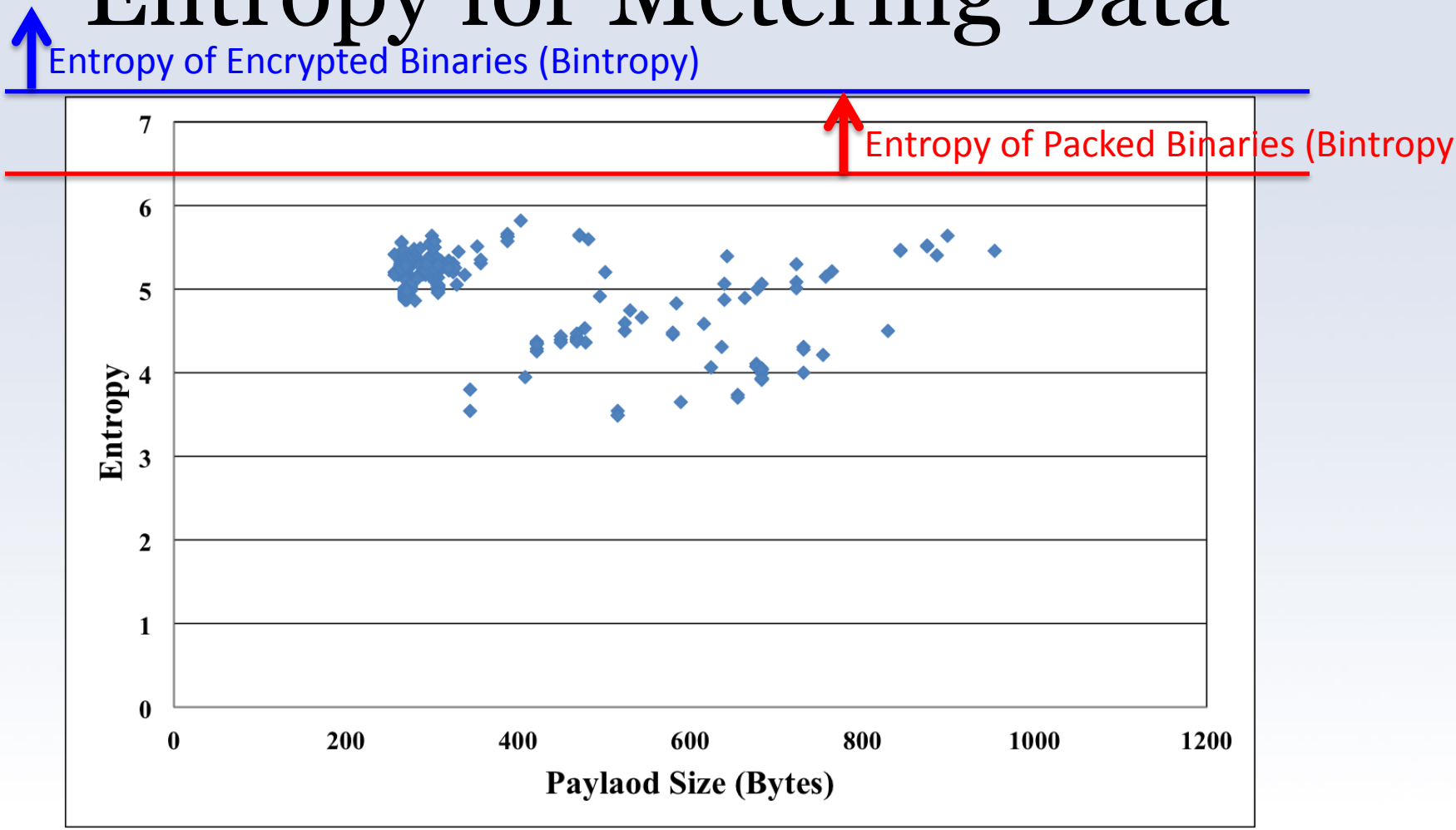


  - Testing Model for metering data
    - Mahalanobis distance to compare to the trained model
    - *Distance(testing file, K-model) > threshold*

Experiments

- Data samples
  - Trained model for 200 malware samples
  - Real 100 metering data files from the TCIPG testbed
- Policy engine implementation
  - Open source GuruX for DLMS/COSEM
  - Intercept API calls of GuruX
- Experiments
  - Entropy & 1-gram distribution & false positive rate & overhead

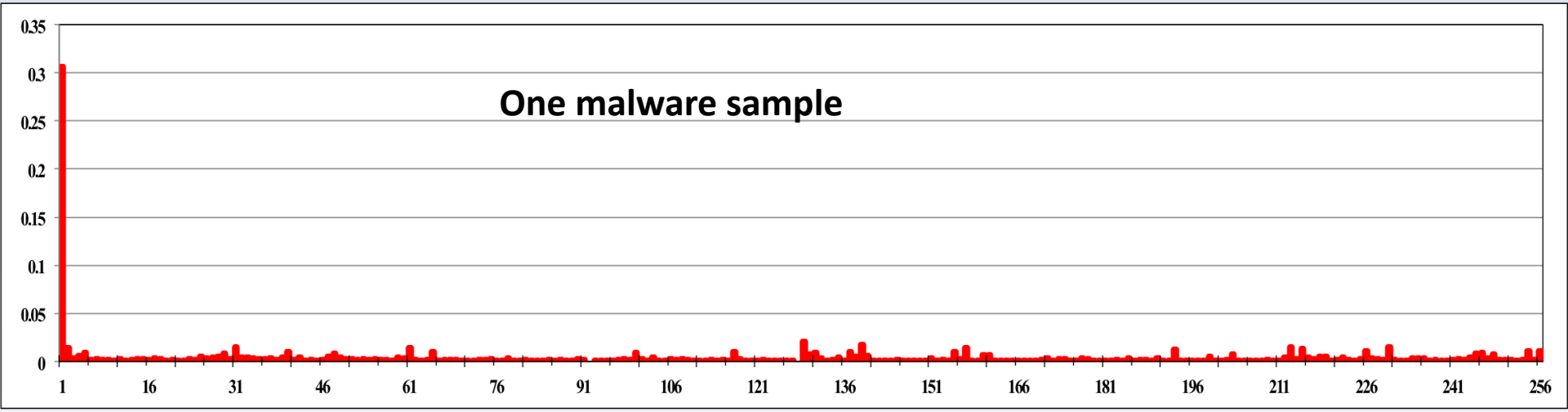
Entropy for Metering Data

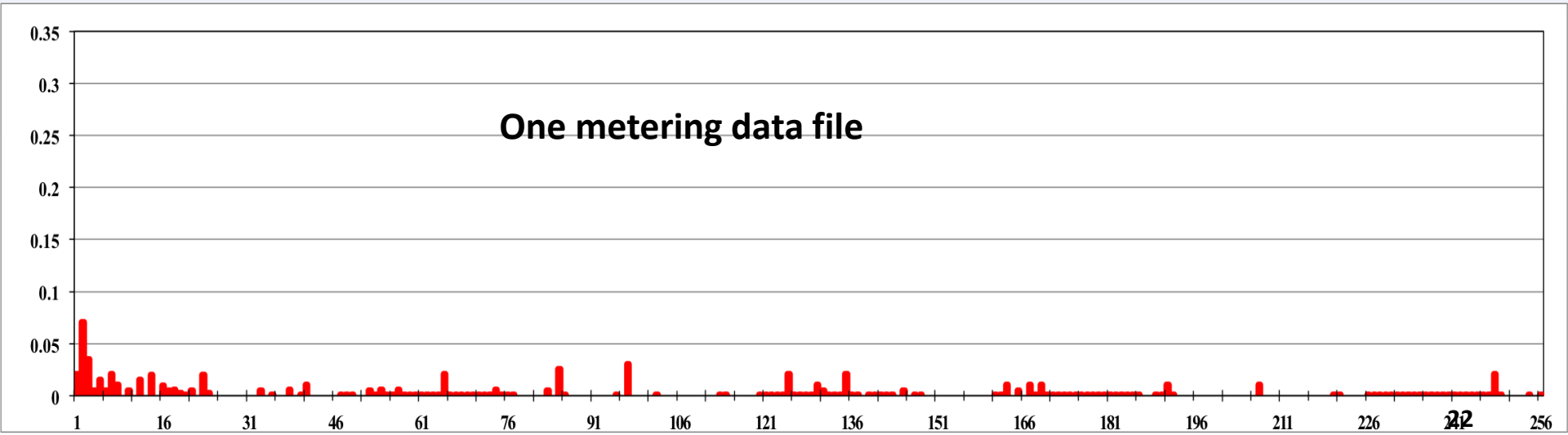


illinois.edu

SAN JOSÉ STATE UNIVERSITY

1-gram Distribution





False Positive Rate

- Metering data are recognized as malware
- 0.17% performance overhead

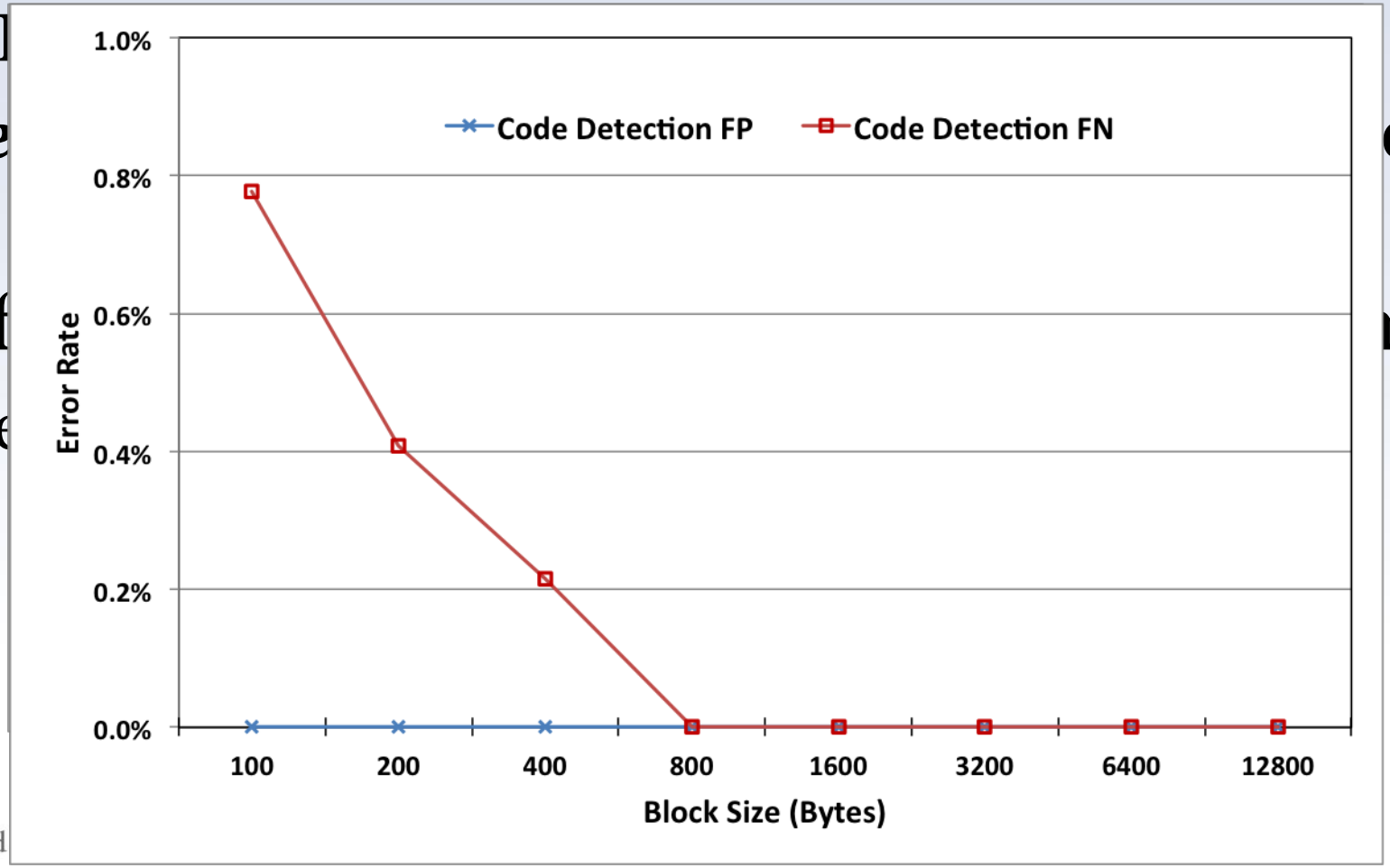
Truncation	50 Bytes	100 Bytes	200 Bytes	500 Bytes	1 KB	2KB
K=1	0%	0%	0%	0%	0%	1%
K=2	0%	0%	0%	0%	1%	1%
K=5	0%	0%	0%	0%	1%	1%
k=10	0%	0%	0%	0%	1%	1%
k=20	0%	0%	0%	1%	1%	2%

Conclusion

- Policy Engine
  - Monitor DMLS/COSEM traffic to detect malware
  - Malware must try to hide as data, but we can ask many policy rules that reveal its presence
  - Use the standard protocol specification and the statistical features of metering data

Current/Future Work

- Improve Accuracy of Malware Detection
  - Al
  - "e 013
- Perf ns
  - Be



illinois.ed

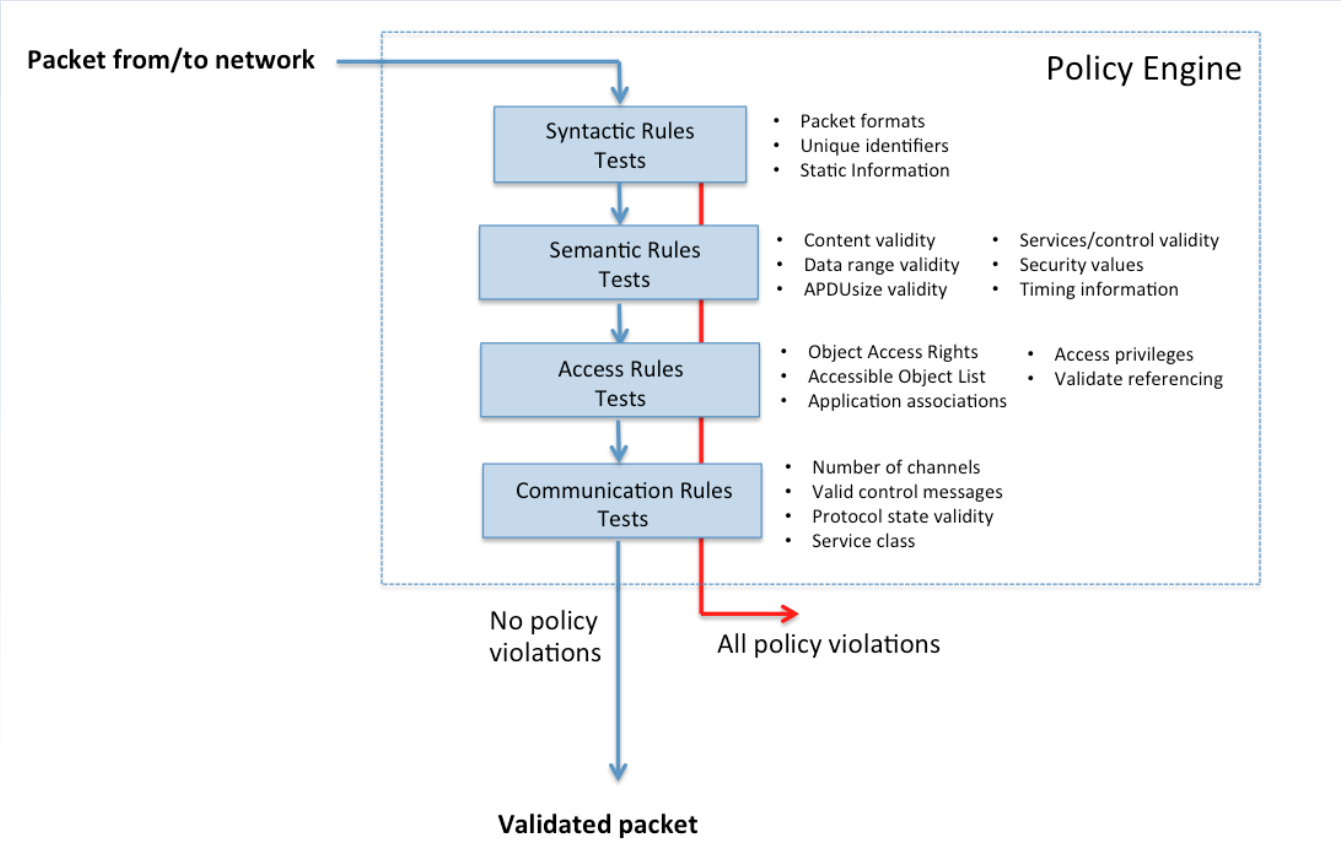
Q & A

Thank You!

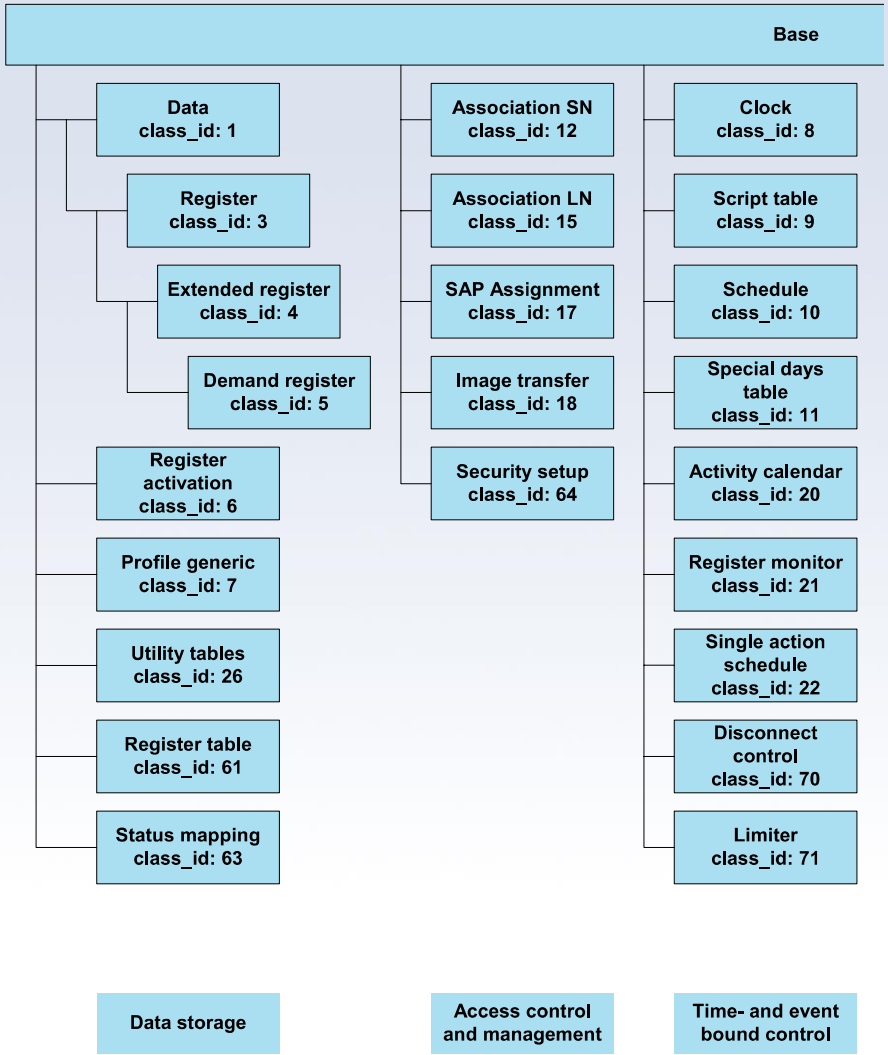
illinois.edu

SAN JOSÉ STATE UNIVERSITY

Standard Protocol Analysis



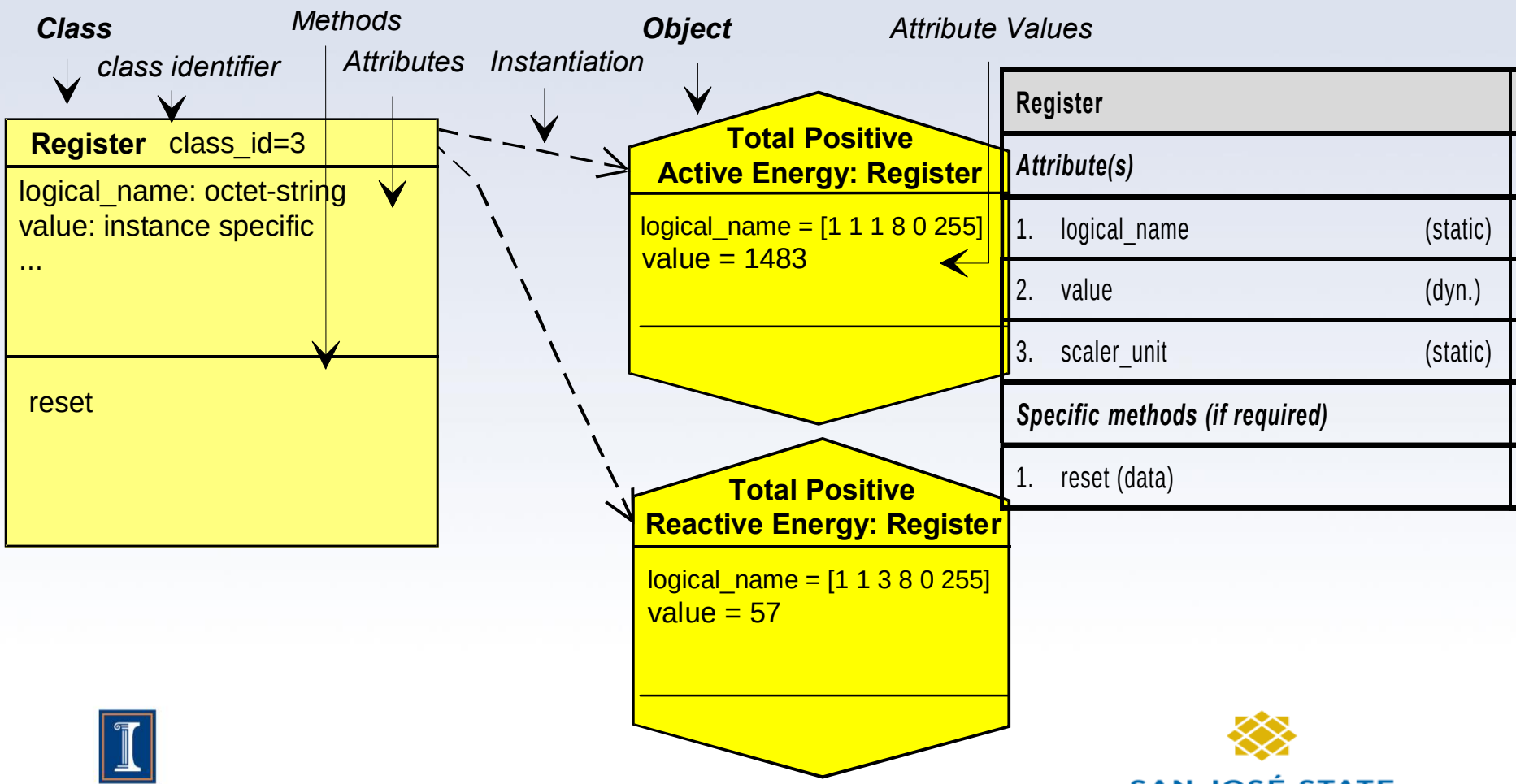
DLMS/COSEM Protocol



illinois.edu

SAN JOSÉ STATE UNIVERSITY

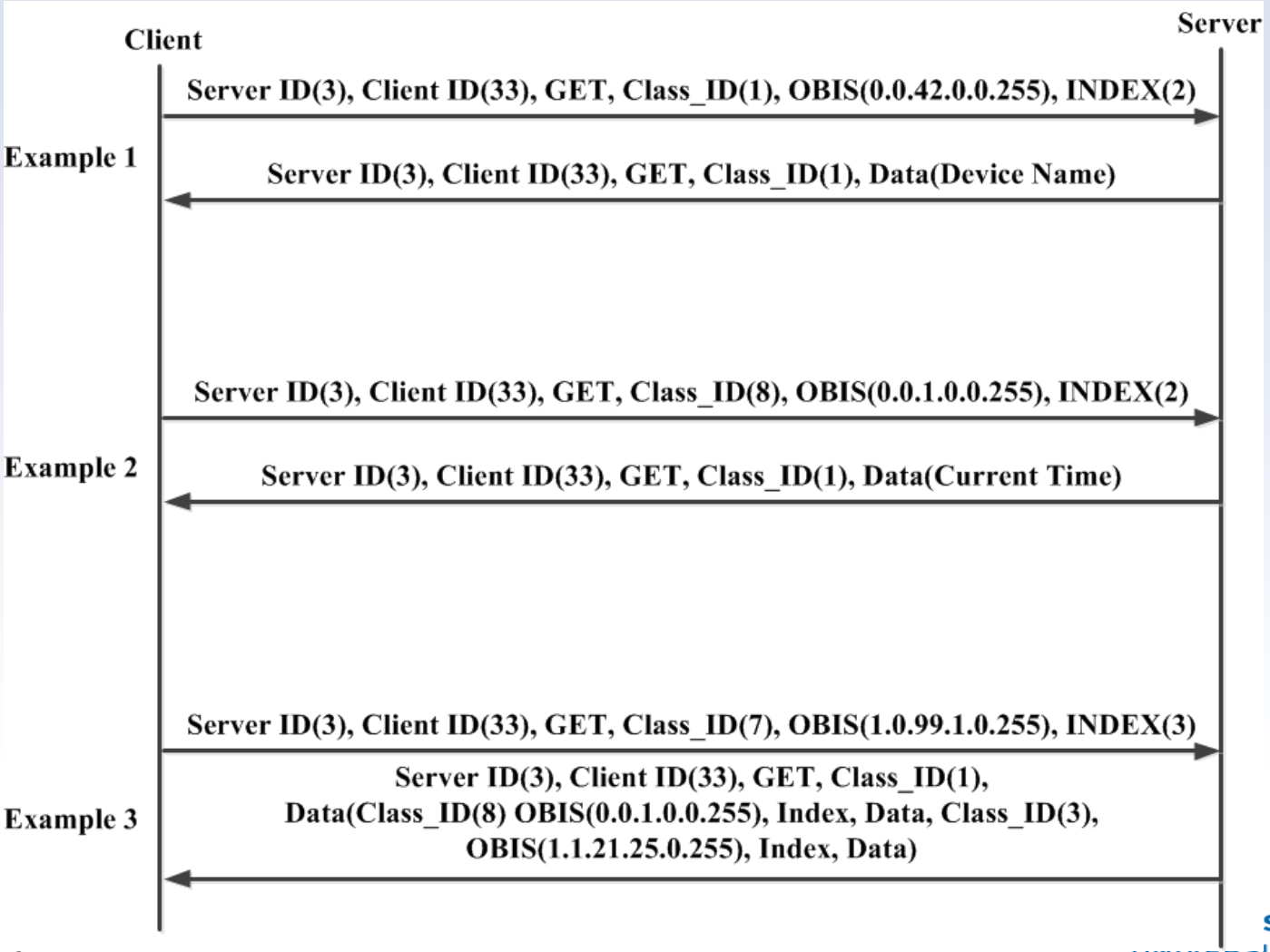
Interface Class



illinois.edu

SAN JOSÉ STATE UNIVERSITY

Example of Communication



Result of Bintropy

Table 1. Computed statistical measures based on four training sets.

DATA SETS	AVERAGE ENTROPY	99.99% CONFIDENCE INTERVALS (LOW TO HIGH)	HIGHEST ENTROPY (AVERAGE)	99.99% CONFIDENCE INTERVALS (LOW TO HIGH)
Plain text	4.347	4.066 – 4.629	4.715	4.401 – 5.030
Native executables	5.099	4.941 – 5.258	6.227	6.084 – 6.369
Packed executables	6.801	6.677 – 6.926	7.233	7.199 – 7.267
Encrypted executables	7.175	7.174 – 7.177	7.303	7.295 – 7.312

$$H(x) = -\sum_{i=1}^{n} p(i) \log_2 p(i) \ ,$$

illinois.edu

SAN JOSÉ STATE UNIVERSITY

FilePrint

- 1-gram distributions
  - 256 distinct bytes from 00000000 to 11111111
- Mahalanobis distance

n to represent a class of file types. The resul
eriments indicate that indeed 1-grams perf

- Manhatan distance

$$D(A,B) = \sum_{i=0}^{255} | A_i - B_i |$$

Modeling

- The K-means algorithm that computes multiple centroids is briefly described as follows:
  1) Randomly pick K files from the training data set. These K files (their byte value frequency distribution) are the initial seeds for the first K centroids representing a cluster.
  2) For each remaining file in the training set, compute the Manhattan Distance against the K selected centroids, and assign that file to the closest seed centroid.
  3) Update the centroid byte value distribution with the distribution of the assigned file.
  4) Repeat step 2 and 3 for all remaining files, until the centroids stabilize without any further substantive change.