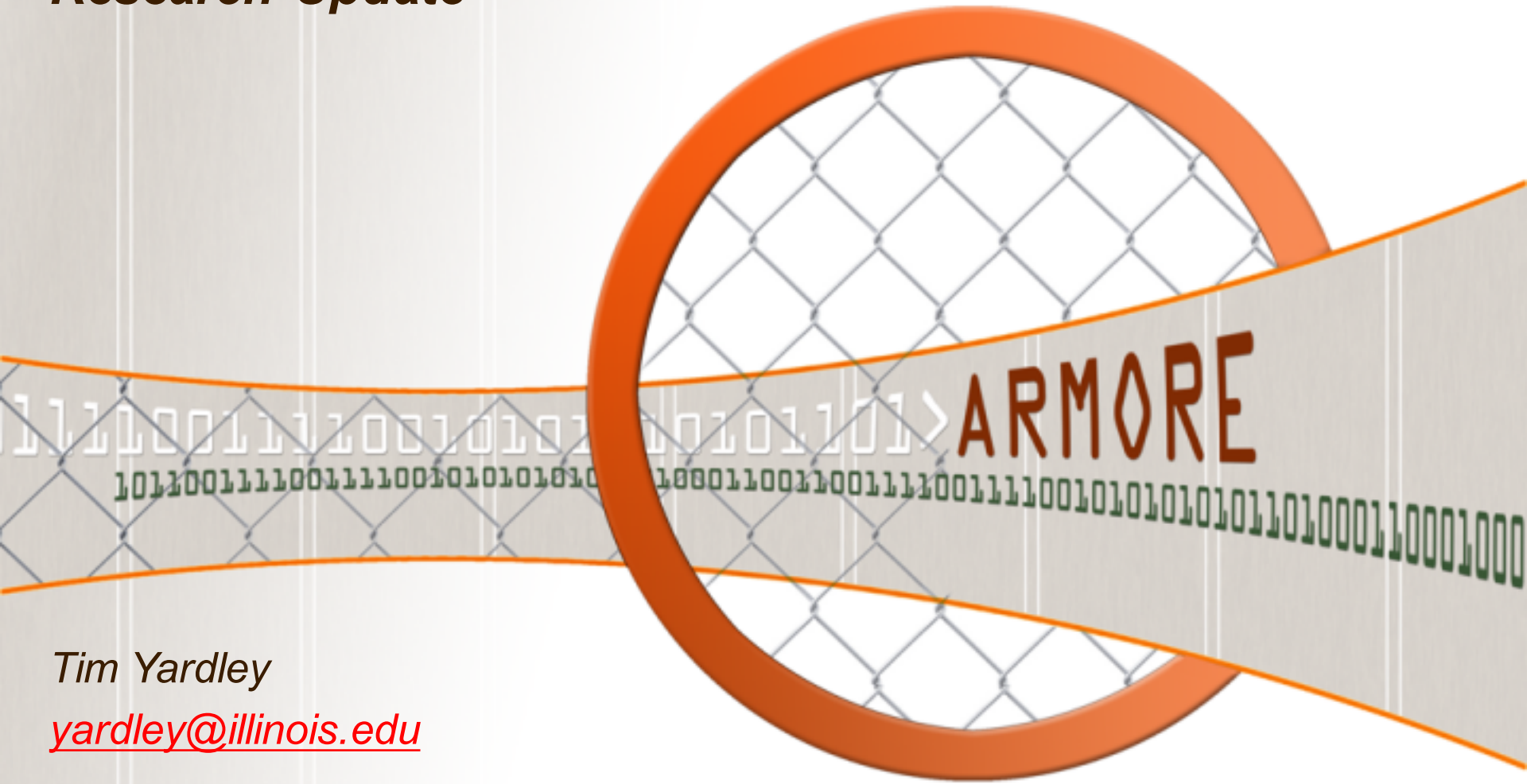
ARMORE

Applied Resiliency for More Trustworthy Grid Operation

Research Update



Tim Yardley

yardley@illinois.edu

About Me

- Tim Yardley, Associate Director of Technology
- Information Trust Institute, University of Illinois Urbana-Champaign

- Old school hacker, Long time practitioner, Current researcher

- @timyardley, yardley@illinois.edu

UIUC's Information Trust Institute

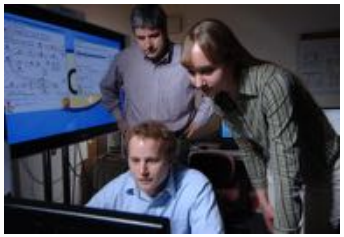
Providing World-Wide Excellence in Information Trust and Security

Institute Vision:
Trust in Complex Systems

Institute Personnel:
Core faculty from CS and ECE
90+ faculty, 28 departments, 11 colleges

Primary Research Themes

- Power Grid
- Evaluation
- Data Science
- Systems and Networking



Background

- Since 2004 startup ITI has won $100M+ in research funding
- Solutions for societal and industrial problems
- Major corporate partnerships
- Led by the University of Illinois College of Engineering

Smart Grid Security Efforts @ Illinois

Centers

TCIPG

Trustworthy Cyber Infrastructure for the Power Grid
~$26.3M effort across 10 years

- Drive the design of a more secure, resilient, and safe electric power infrastructure
- $7.5M NSF center (2005 – 2010), $18.8M DOE-OE (CEDS) & DHS center (2010 – 2015)
- University of Illinois, Washington State, Dartmouth, Arizona State

ADSC ADVANCED DIGITAL SCIENCES CENTER

Smart Grid Subprogram (~$15M effort across 5 years)

Cybersecurity, Microgrids, DERs, and HANs

Illinois Center for a Smarter Electric Grid (~$5M effort across 5 years)

- Validation of IT and control aspects of the Smart Grid
- Operates facilities equipped with HW/SW to aide in the validation of emerging smart grid systems
- Focus on both power and cyber related issues

Assured Cloud Computing (~$6M effort across 6 years)

- Leveraging trustworthy cloud computing for critical infrastructure

Science of Security Systems (~$8.5M effort across 4 years)

- Resiliency, security, and trust in complex engineered systems

Highlighted Projects

Policy Based Configuration (PBCONF)

EPRI ELECTRIC POWER RESEARCH INSTITUTE Ameren

Software Defined Networking

SEL Ameren

Applied Resiliency for More Trustworthy Grid Operation (ARMORE)

GRID PROTECTION ALLIANCE Pacific Northwest NATIONAL LABORATORY

Collaborative Defense for T&D Devices Against Attack (CODEF)

ABB

Cyber-Physical Modeling and Analysis for a Smart and Resilient Grid

PowerWorld CORPORATION U

... and many more

ARMORE

Overview

Motivation

- Industrial Control Systems (ICS) protocols lack security protection
- Security bolt-ons are typically implemented via firewalls and VPNs
- Little if any visibility as to what these systems are actually doing
- Any security extensions have a long-tail implementation path (or never at all)
- Deployments are often much more costly than the capital expenditures

What is ARMORE?

- Security appliance that aims to
  - *Increase visibility and awareness on ICS networks*
  - *Augment insecure protocols with security features*
  - *Inspect and (optionally) enforce defined policies*
  - *Minimize deployment costs while creating a feasible adoption path*

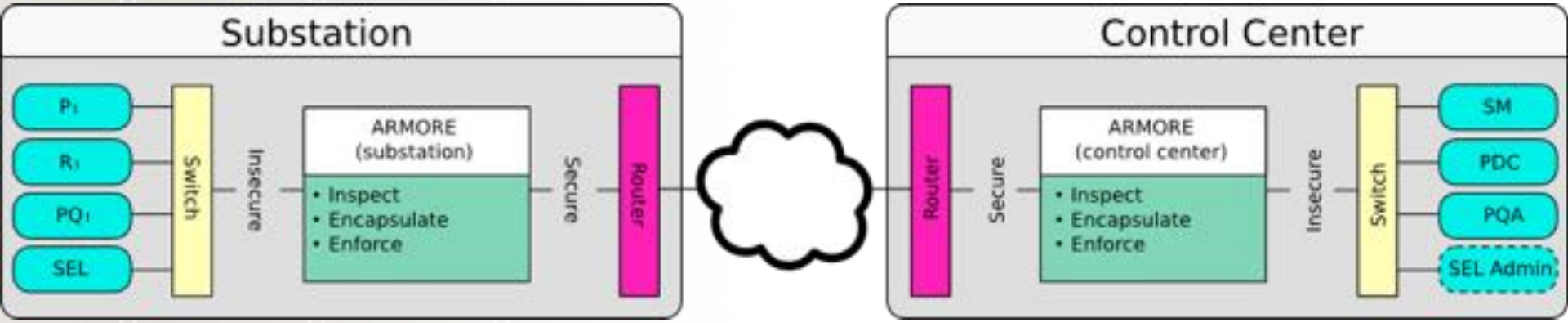
How ARMORE Works

- Passive
  - *Span port*
- Transparent
  - *Inline inspection, optional enforcement*
- Encapsulated
  - *Inline inspection, encapsulated transfer with optional encryption, optional enforcement*

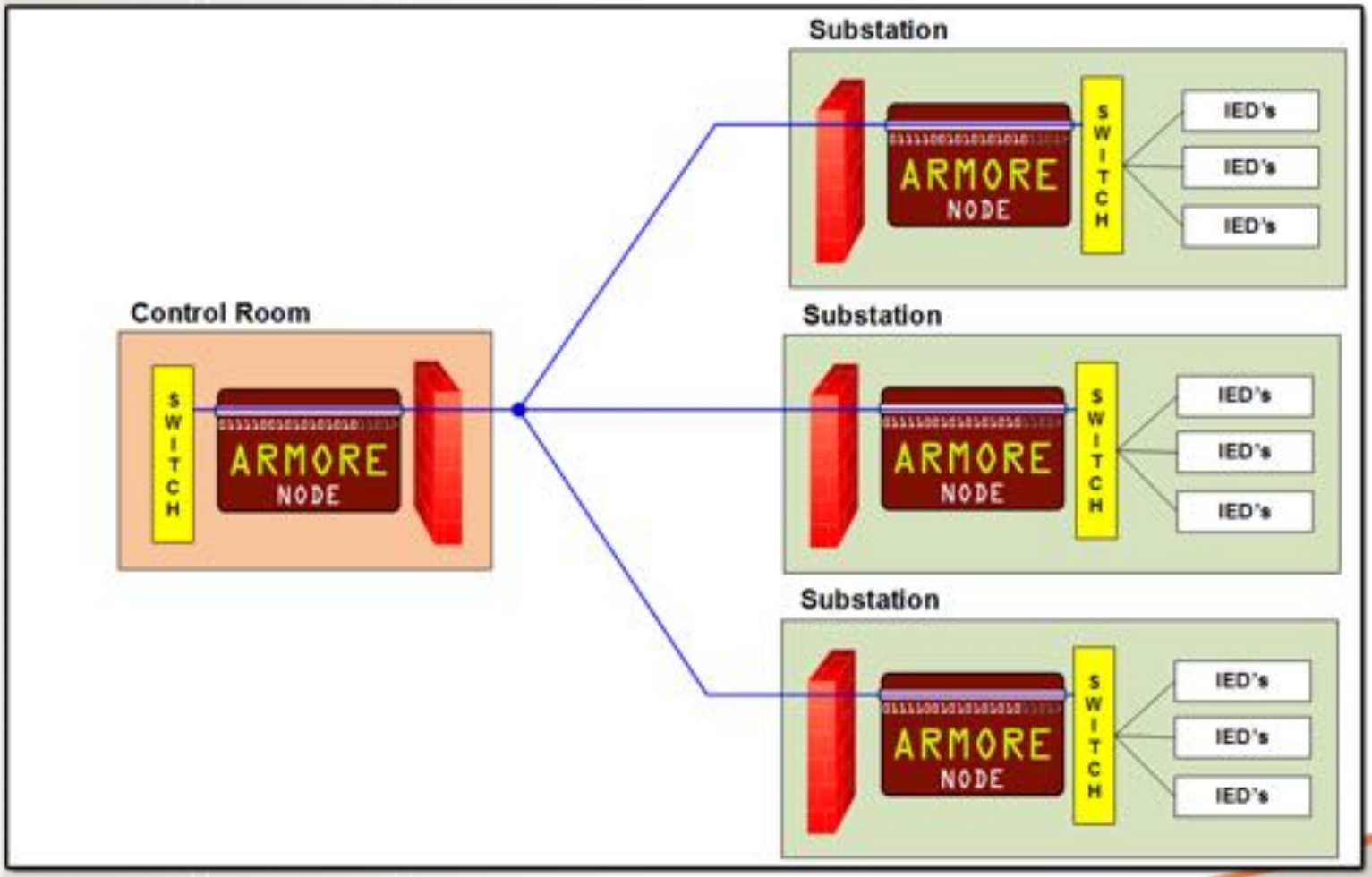
What do you get?

- Passive
  - *Network visibility and intelligence*
- Transparent operation
  - *Passive plus…*
  - *Communication endpoints operate without any changes*
  - *Optional policy enforcement*
- Encapsulated
  - *Transparent plus…*
  - *Encapsulation and Encryption*
  - *Security augmentation (access control/filtering)*
  - *Optional policy enforcement*
  - *Fault tolerance and resiliency options*
- Other value adds
  - *Enhanced access control*
  - *Payload inspection*
  - *Data processing and analysis*

ARMORE Conceptual Diagram

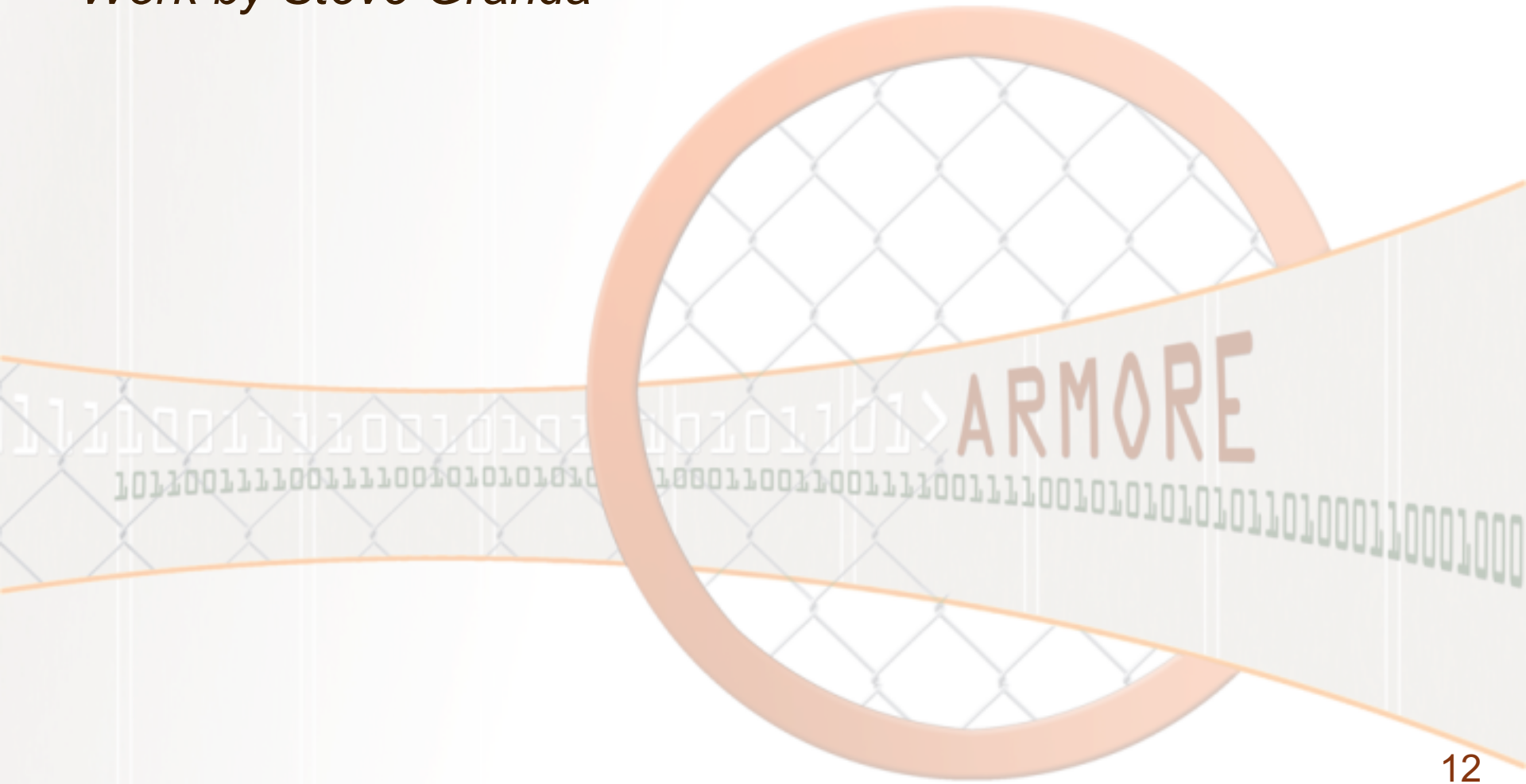


In deployment…



System Realization

Work by Steve Granda



ARMORE Software

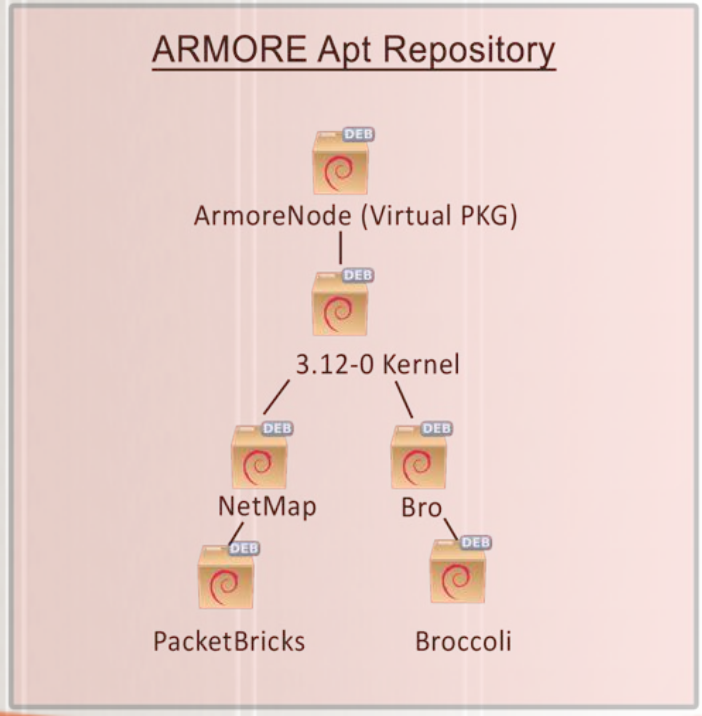
- OS: Debian Wheezy 7.8 x64
  - *Modified 3.12.0 Linux Kernel*
- ARMORE Proxy
  - *Abstracted middleware encapsulator*
- Bro
  - *Intrusion Detection System*
- NetMap
  - *Kernel Module for High Speed Packet I/O*
- Management/Configuration
- ZMQ
  - *Middleware layer*
  - *CurveZMQ*
    - *Authentication and Encryption protocol for ZMQ*

Other ARMORE Support

- BrocolliSharp
- Bro-statsd
- Rsyslogd
- Etckeeper

ARMORE Node installation

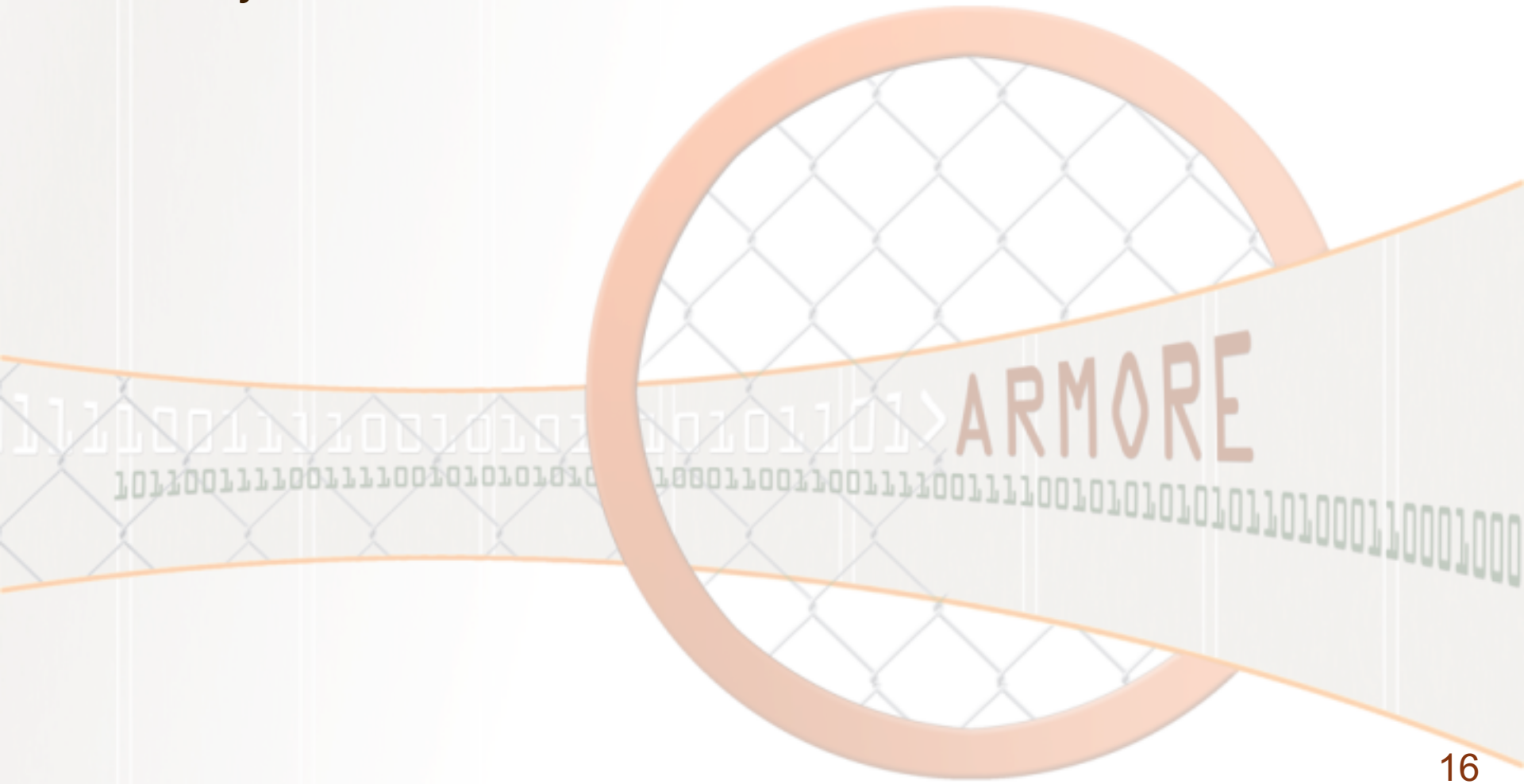
- Original installation was via a large shell script which compiled and installed software from source.
- Current installation is with our debian repository



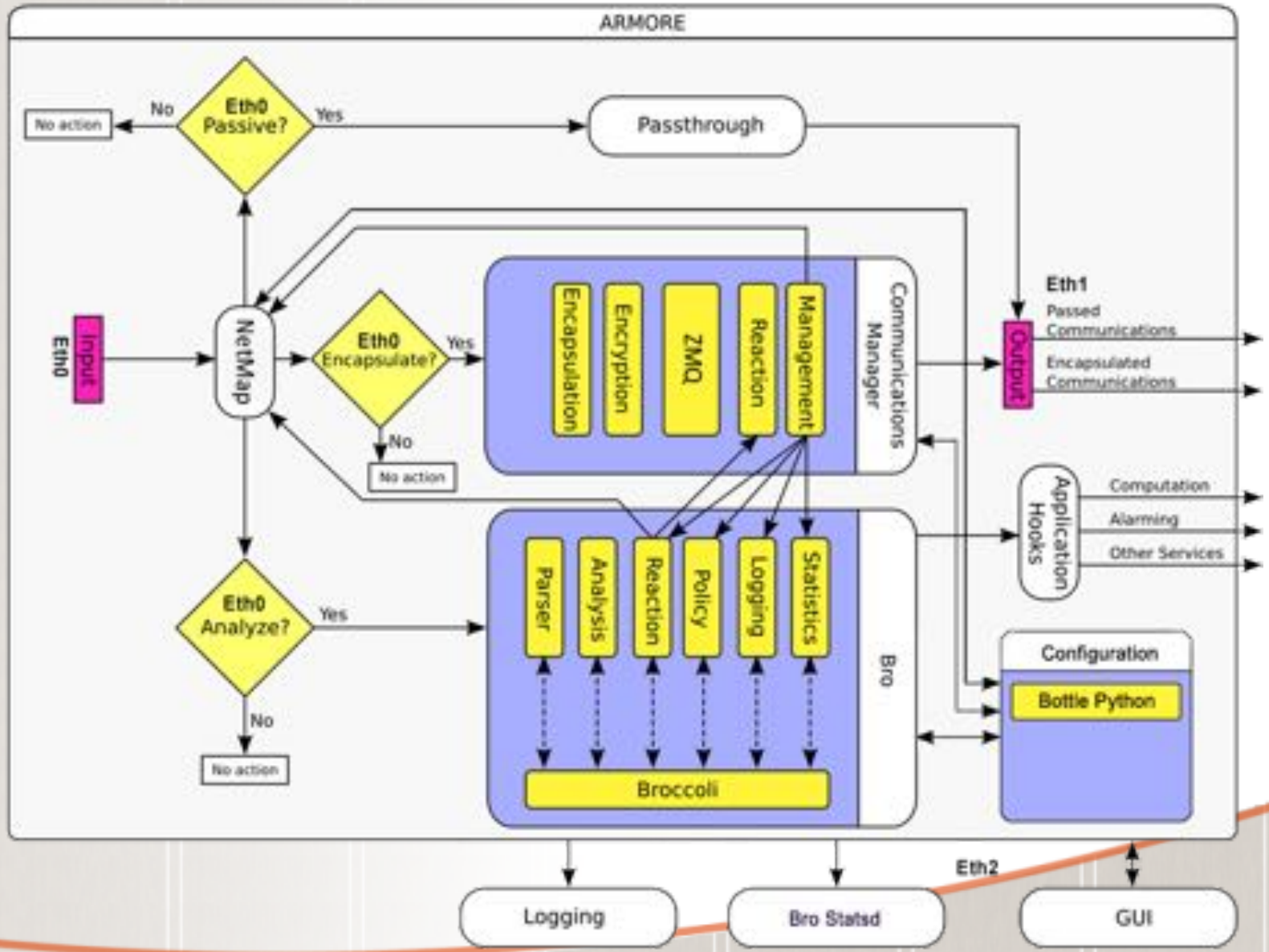
- Allows easier dependency checking and updating of individual components.
  - *apt-get install armorenode*
  - *apt-get update armorenode*

Middleware

Work by Chris Drew and Steve Granda



Scope in ARMORE



ARMORE Proxy

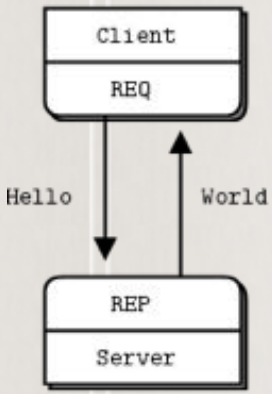
- Abstract class for middleware library inclusion
  - *ZeroMQ implemented with Curve security*
  - *DDS stubbed but not implemented*
    - *Reason: Open source libraries are currently lacking security extensions*
- Abstract packet capture interface
  - *PCAP*
  - *Netmap*
- Many options for logging
- MAC address translation mode

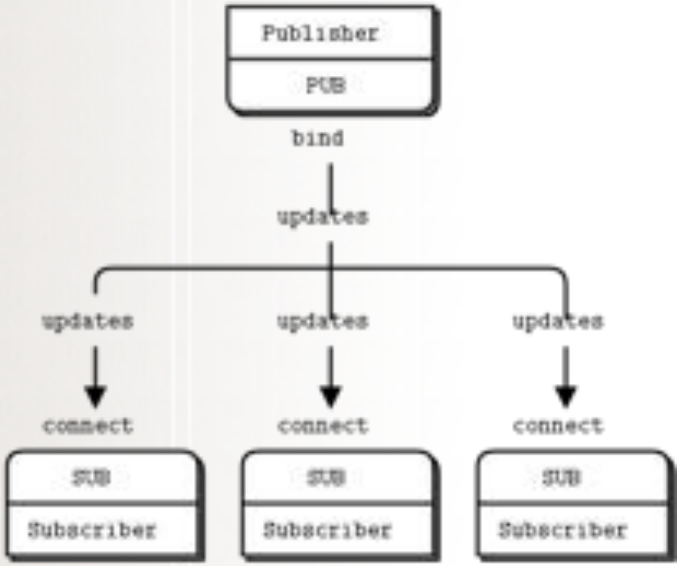
ZMQ

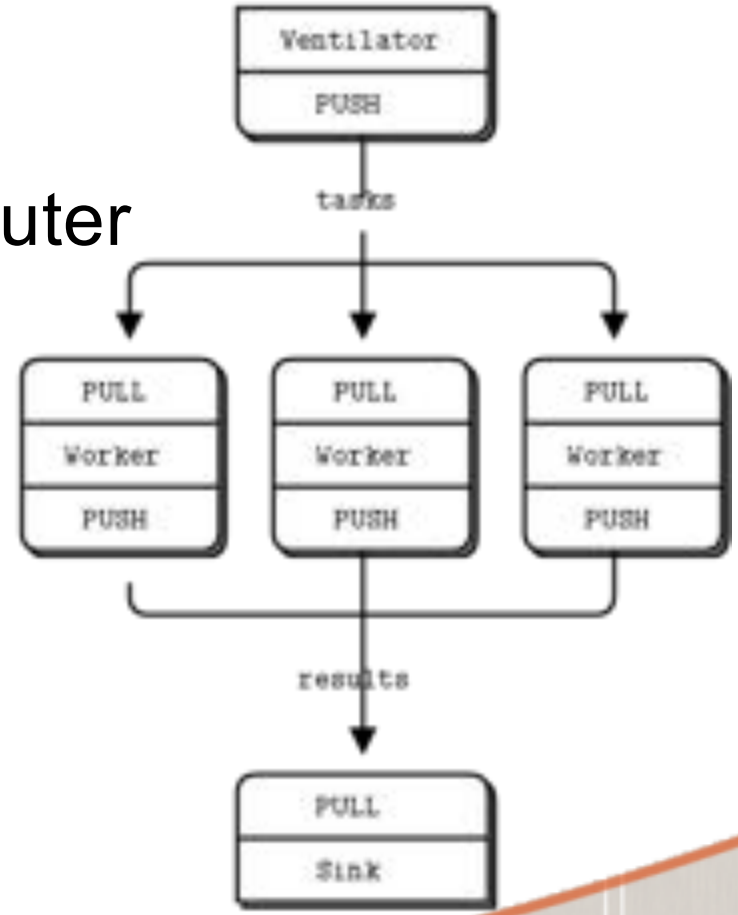
- Asynchronous messaging library
- Allows many types of communication from intra-process to WAN
- Removes need for message broker
- API values simplicity over functionality
- Encourages user to implement functionality as needed
- Available in over 30 languages on multiple platforms
- Open source
- Very active community provides extensive support for developing and debugging
- Existing documentation provides extensive instruction on various communication patterns

ZMQ - Patterns

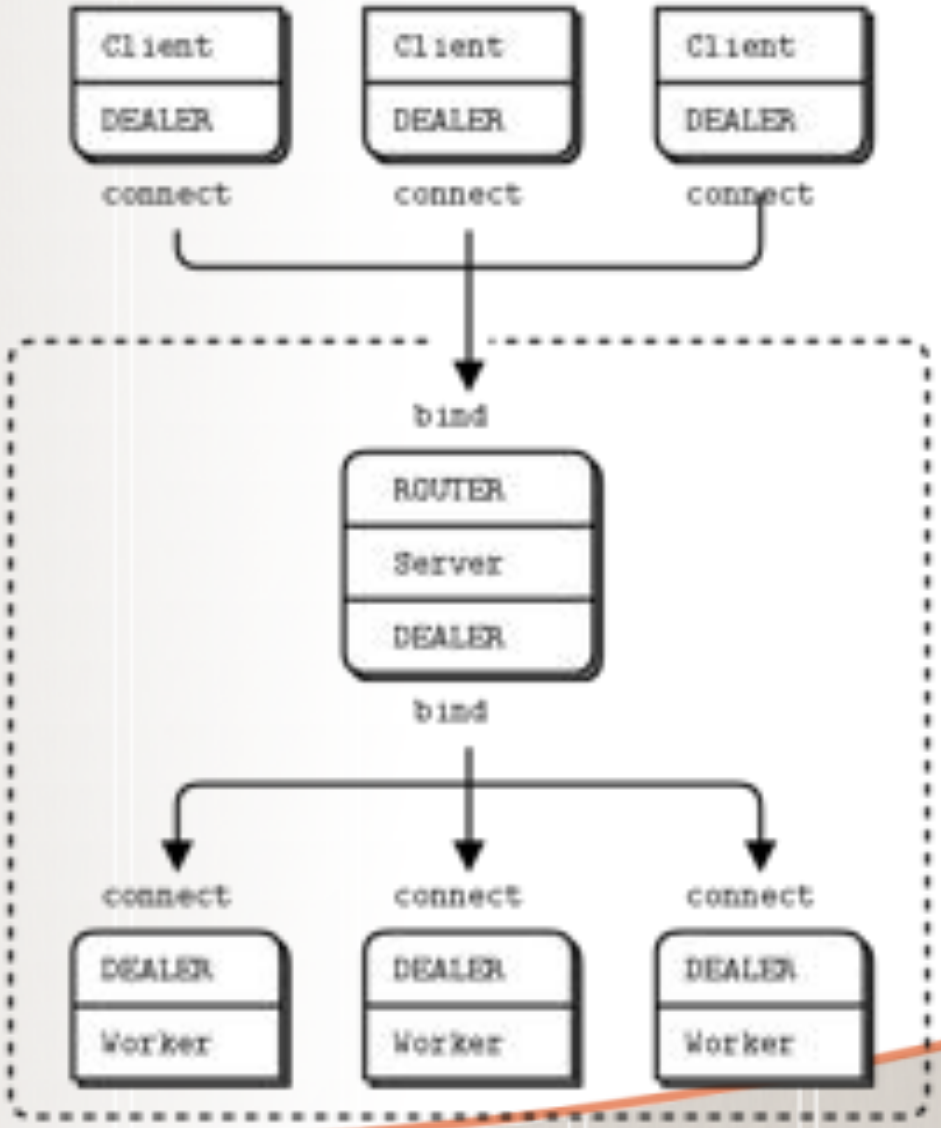
- Provides ability to create many communication patterns
- ARMORE is utilizing a dealer/router pattern







ZMQ Dealer/Router Pattern

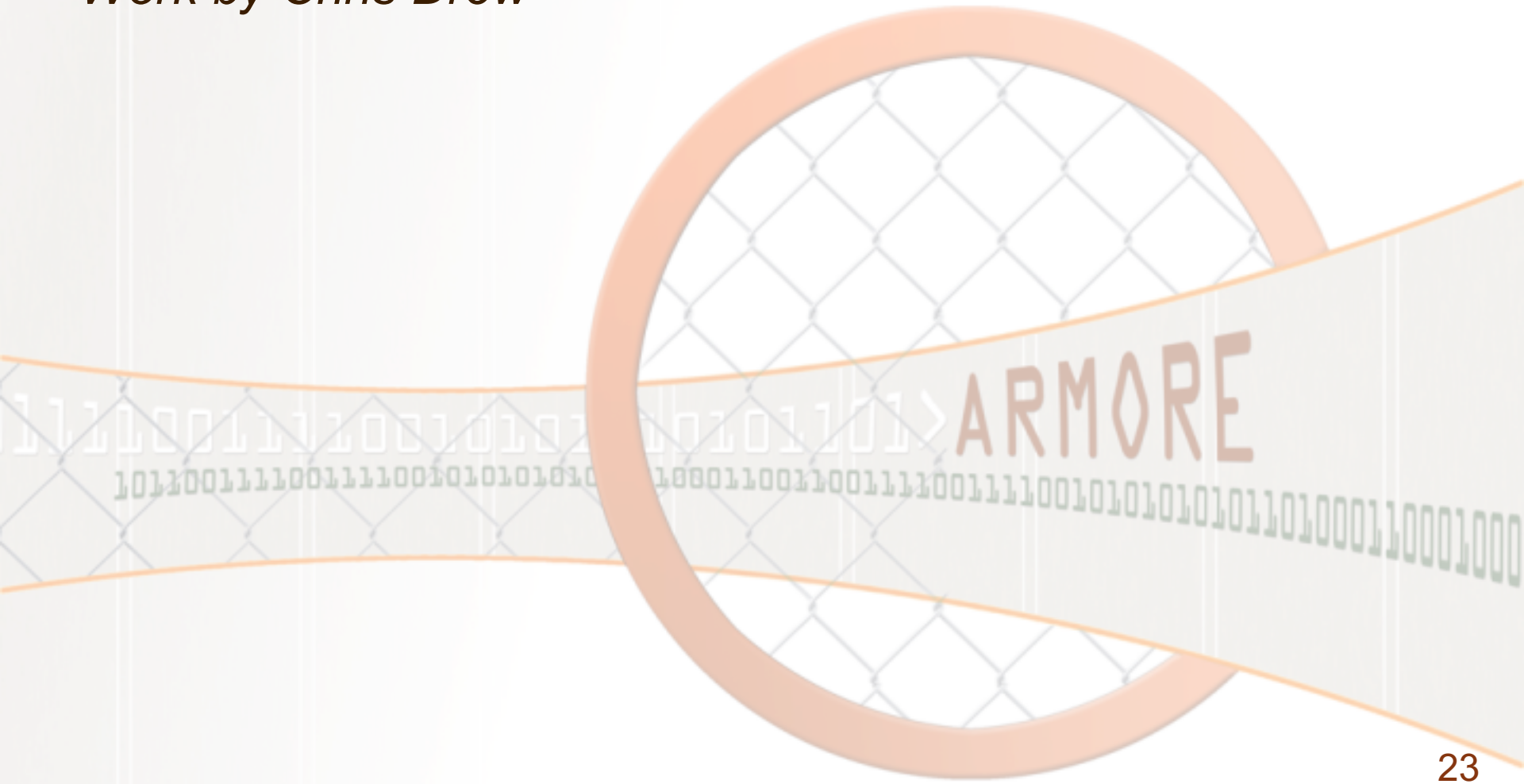


DDS vs. ZMQ

DDS	ZMQ
• Commercial Product	• Open source
• Desired functionality built in	• Some functionality may need to be written
• Steep learning curve	• Easy to learn
• Sightly more resource heavy	• Lightweight
• ~4 languages	• 30+ languages
• Restricted to pub/sub	• Flexible to multiple patterns

System Administration

Work by Chris Drew



Web API

- Front end connects UI with ARMORE node internals
  - *Read/set configuration*
    - *Subsystem status*
    - *Node topology*
  - *Display data for user*
    - *Statistics*
    - *Logs*
    - *Alerts*
- Communicate with back end via JSON messages
- Testing
  - *Janus - Rest API server*
  - *Bottle - Python Web Framework*

Example Endpoints

- armore/config/zmq/5 (NOTE: node id 5)

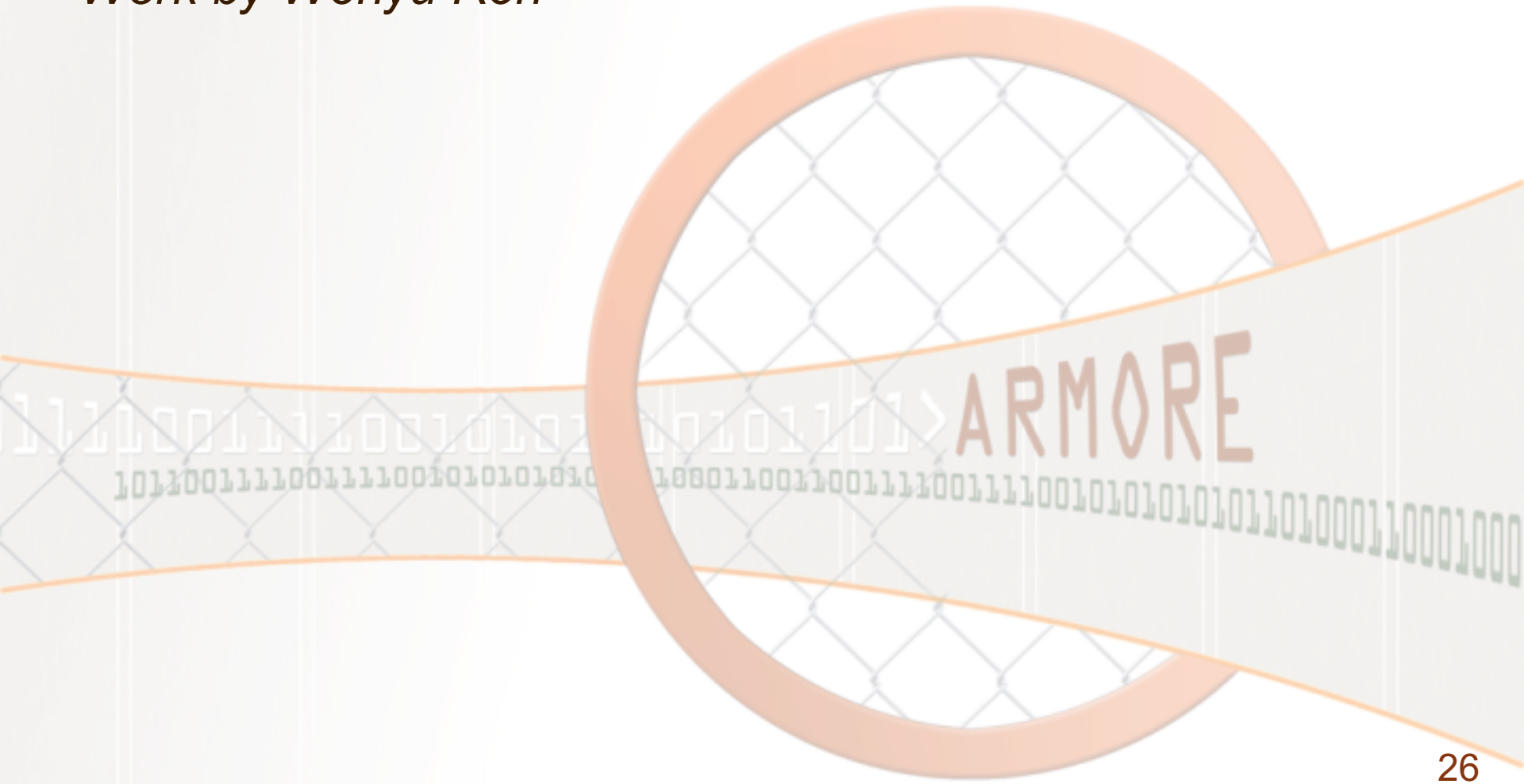
```
{
    "Encryption": True,
    "Reliability": "Best Effort",
    "Durability":"Transient Local”
}
```

- armore/notifications/bro

```
{"eventIds": [{
    12: {
        "time": "7/13/2013 12:45:01",
        "srcNode": "Node_2”,
        ....
    58: {
        "time": "9/3/2013 12:45:01",
        "srcNode": "Node_91",
        …
}]}
```

Dynamic and Smart Traffic Analyzer for Smart Grid

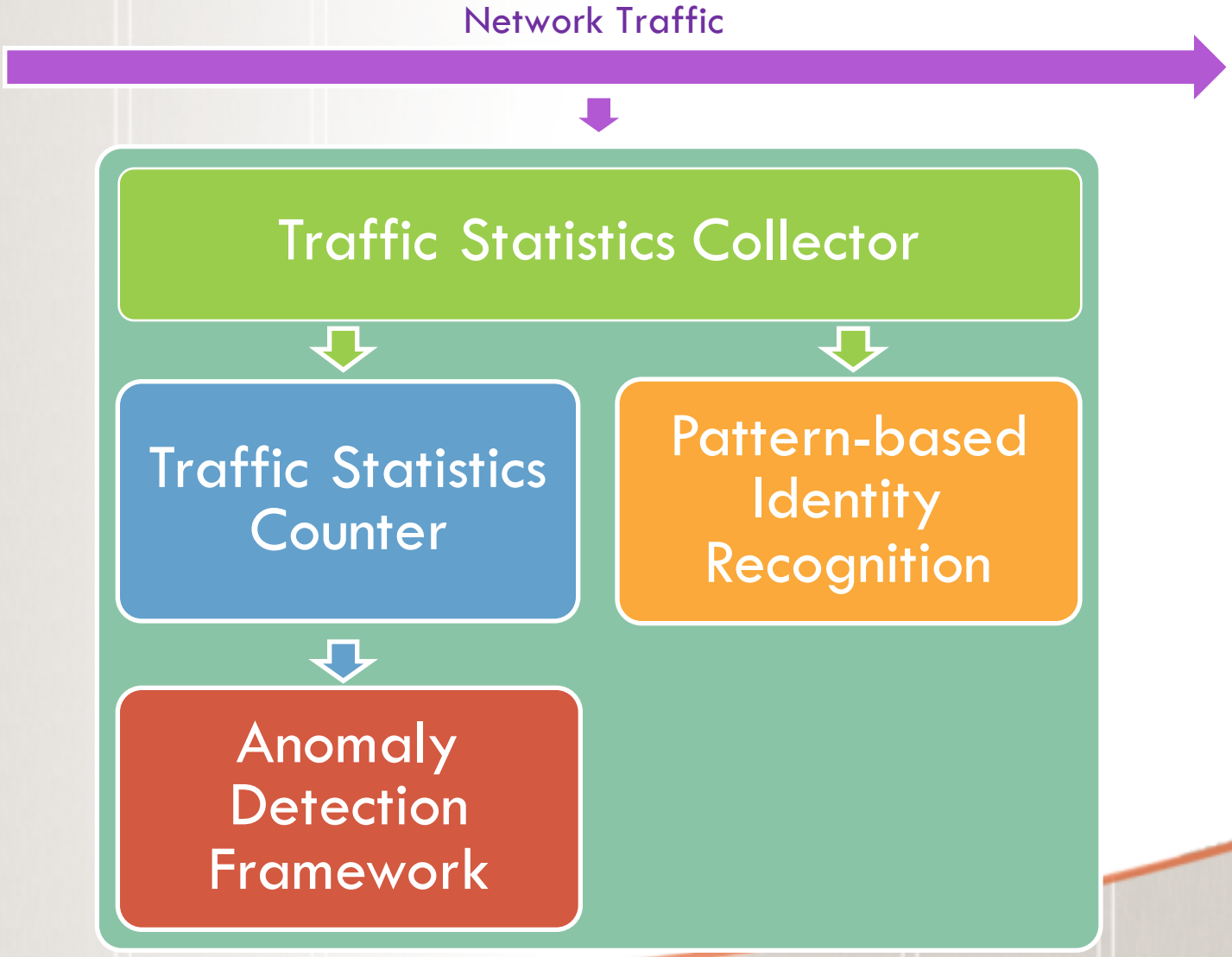
Work by Wenyu Ren



Introduction

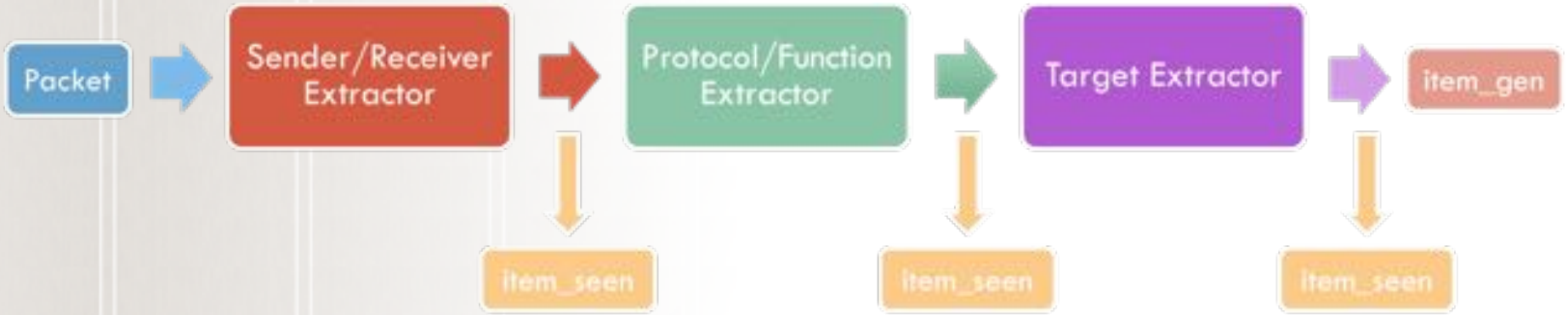
- What is it?
  - *An analyzer that provides dynamic and intelligent analytics for SCADA protocols, increasing visibility into the system behavior*
- What is it using?
  - *Bro's scripting engine*
- What protocols does it support at the moment?
  - ✓ *DNP3*
  - ✓ *Modbus*
  - ✓ *Extensible to any other protocol*

Structure

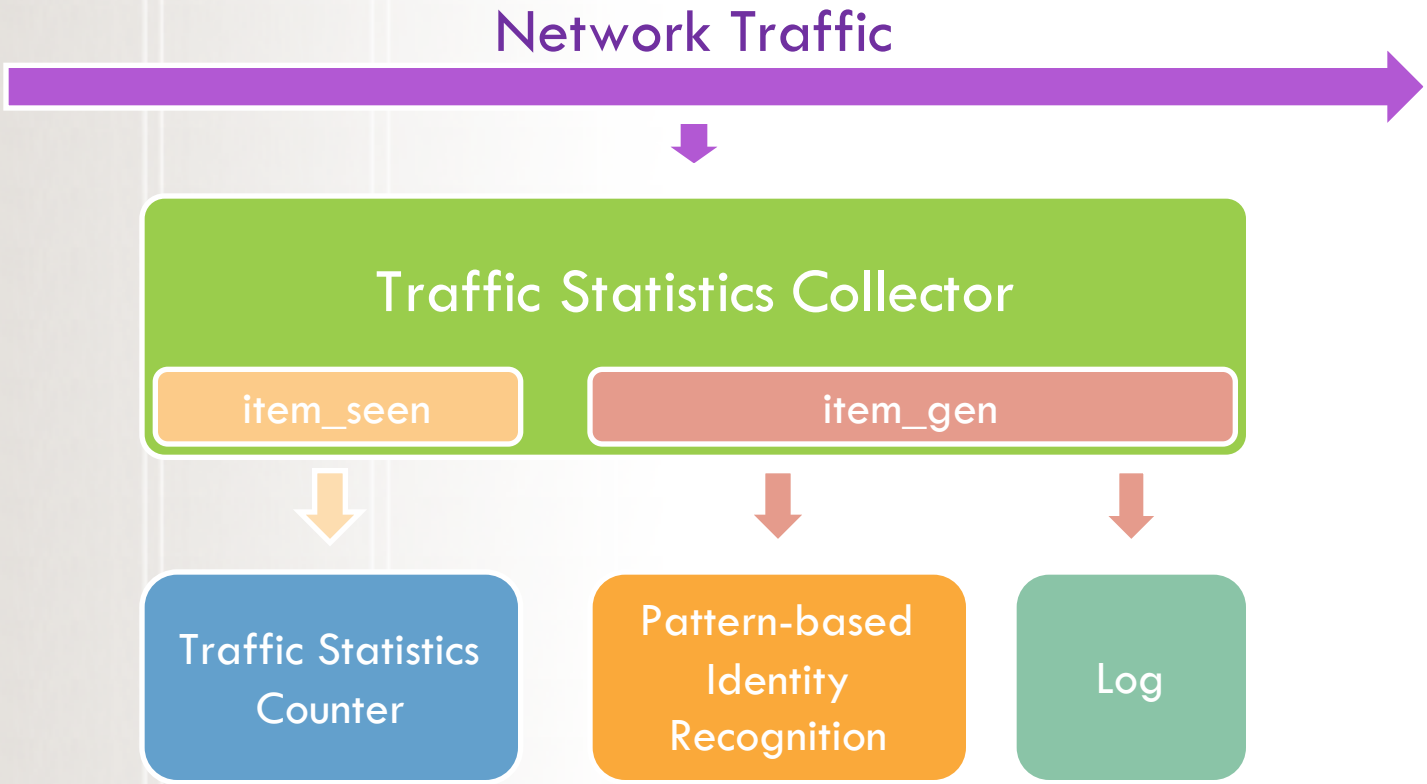

Network Traffic
Traffic Statistics Collector
Traffic Statistics Counter
Pattern-based Identity Recognition
Anomaly Detection Framework

Traffic Statistics Collector

- Input: network traffic
- Output: two kinds of events
  - *item_seen: instantaneous, item contains incomplete information of the packet*
  - *item_gen: delayed, item contains complete information of the packet*



Traffic Statistics Collector



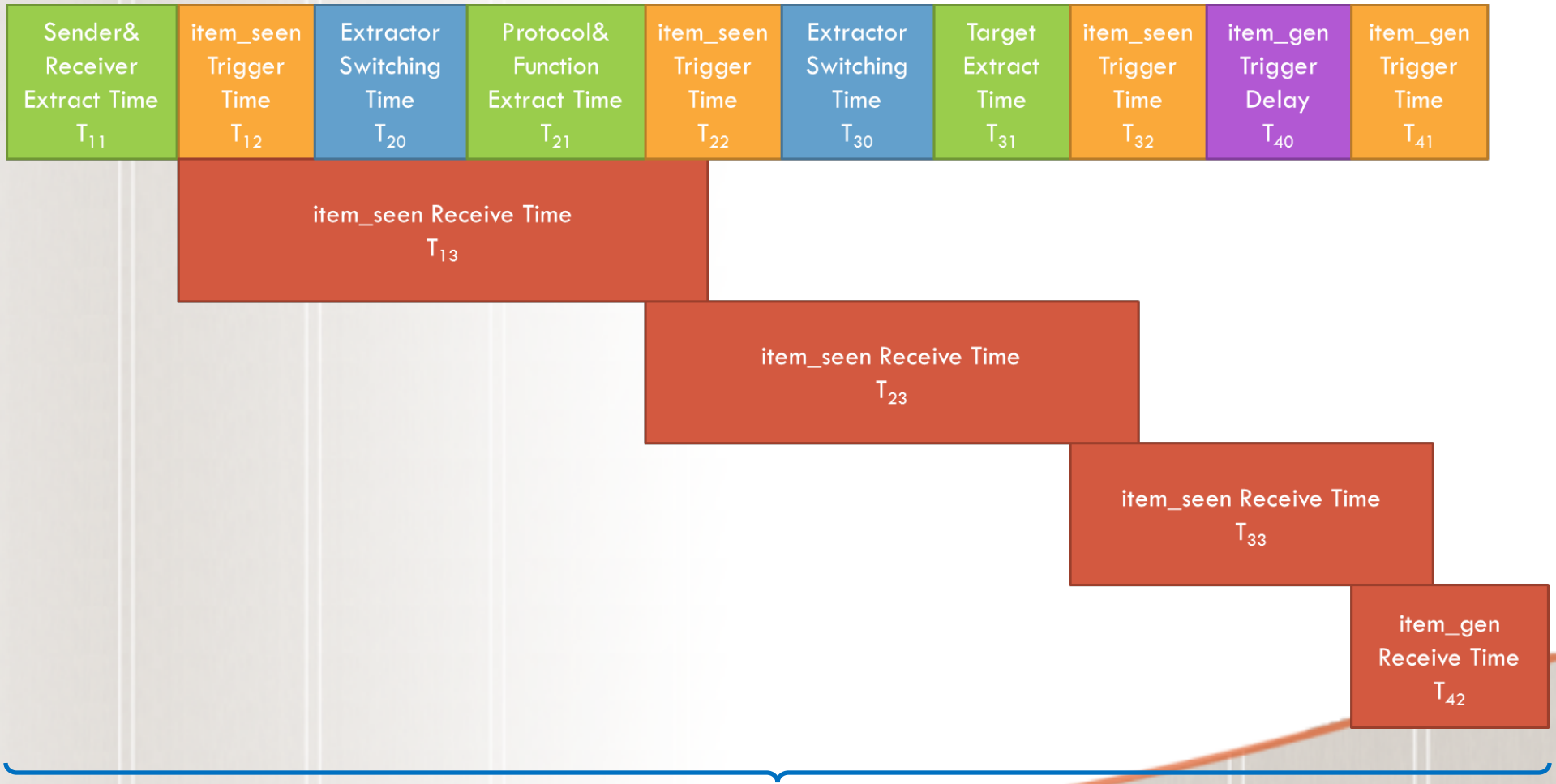
Traffic Statistics Collector

- Trace: synthetic Modbus traces

Subject	Value
Average Packet Interval	6ms 876us
Average Burst Interval	1s 824ms
Average Burst Length	32
Total Valid Time	1h 2min
Total Packet Number	60227

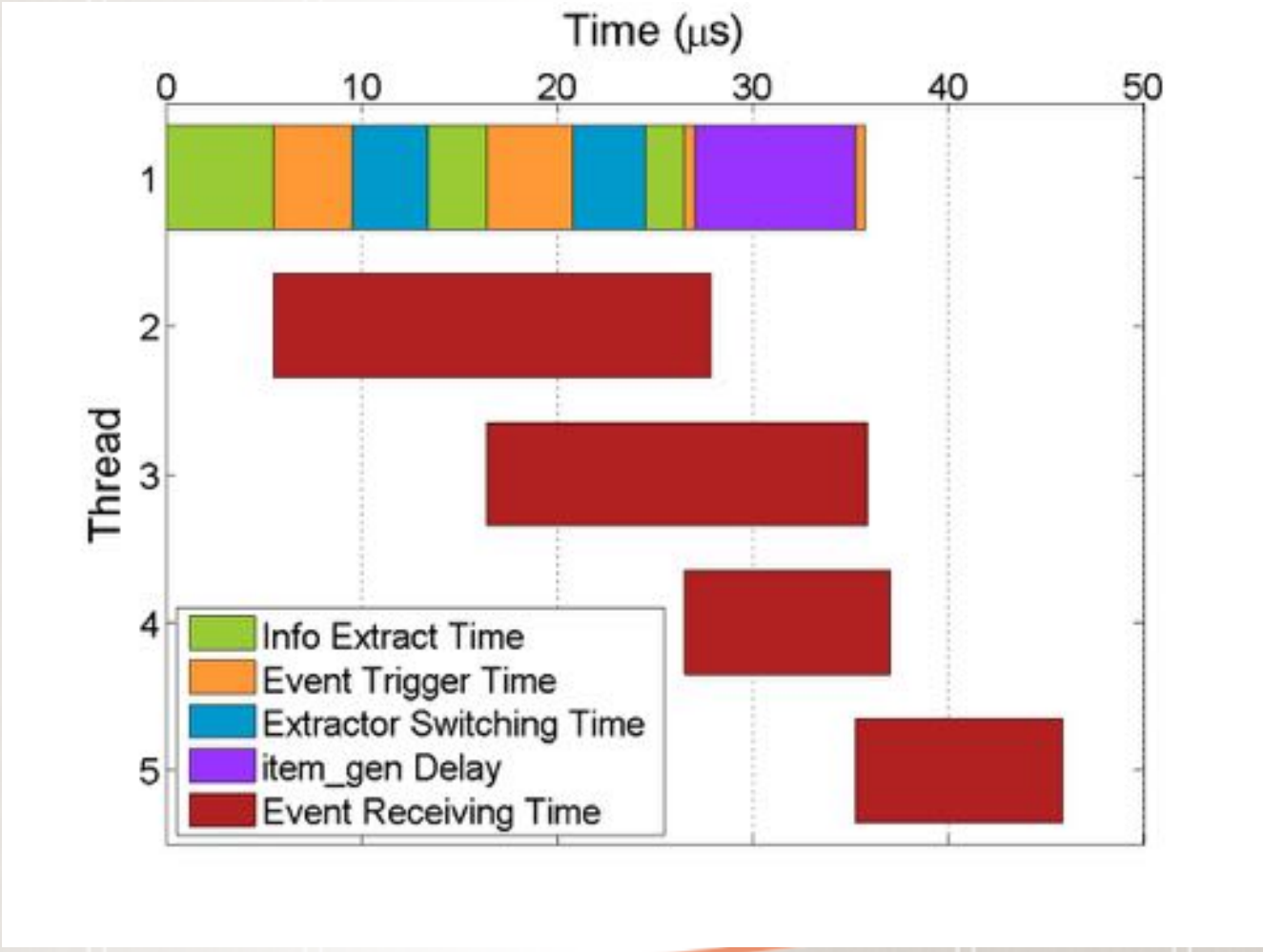
Traffic Statistics Collector

- 5 level



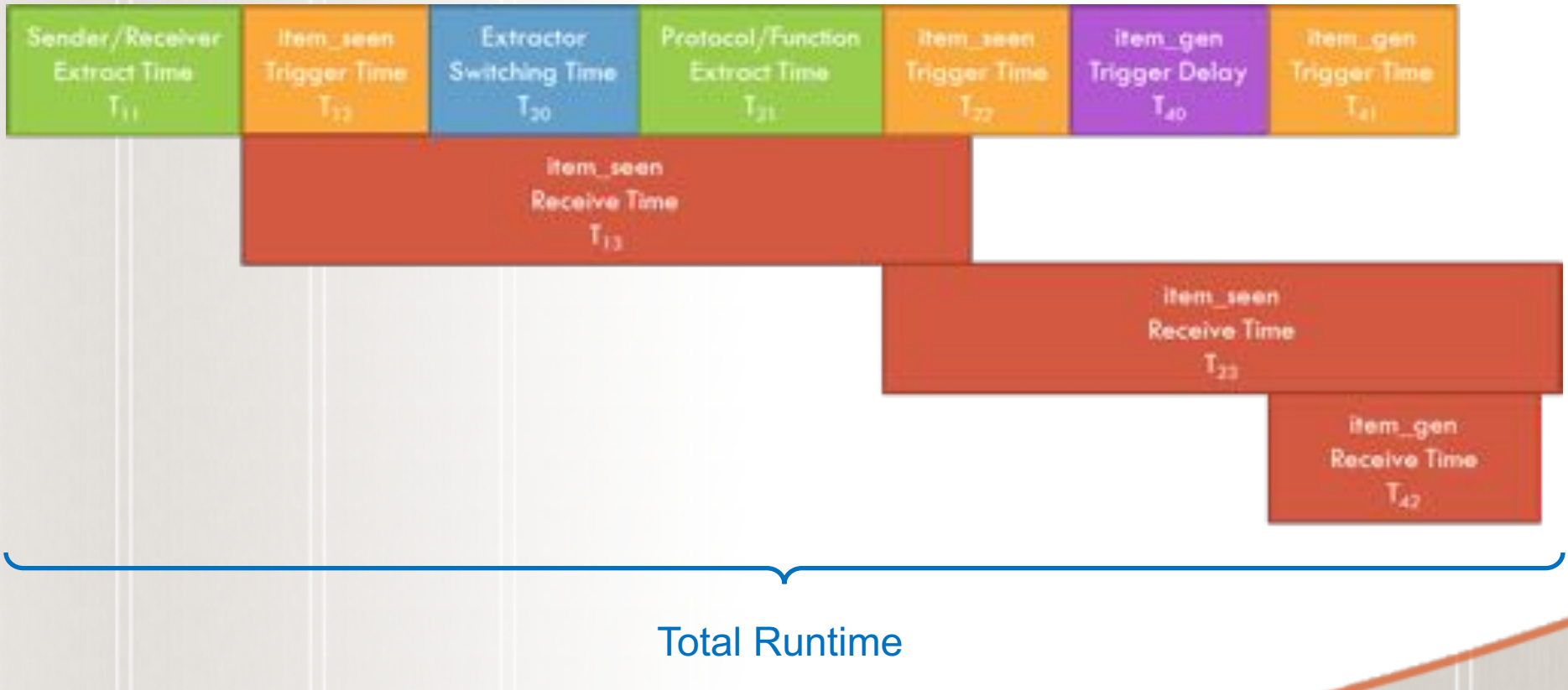
Traffic Statistics Collector

- 5 level



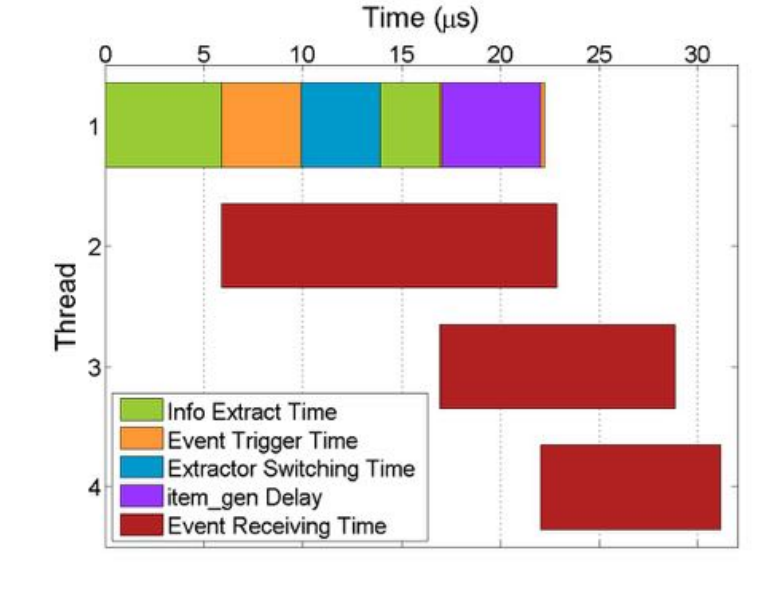
Traffic Statistics Collector

- 3-4 level

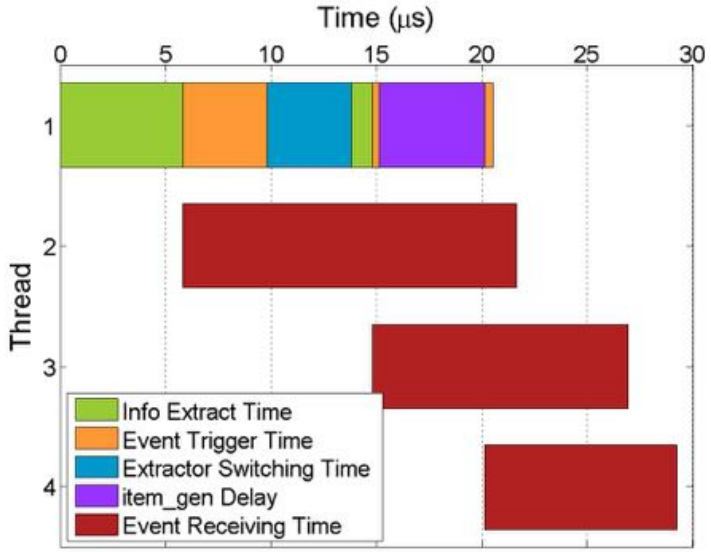


Traffic Statistics Collector

- 3-4 level



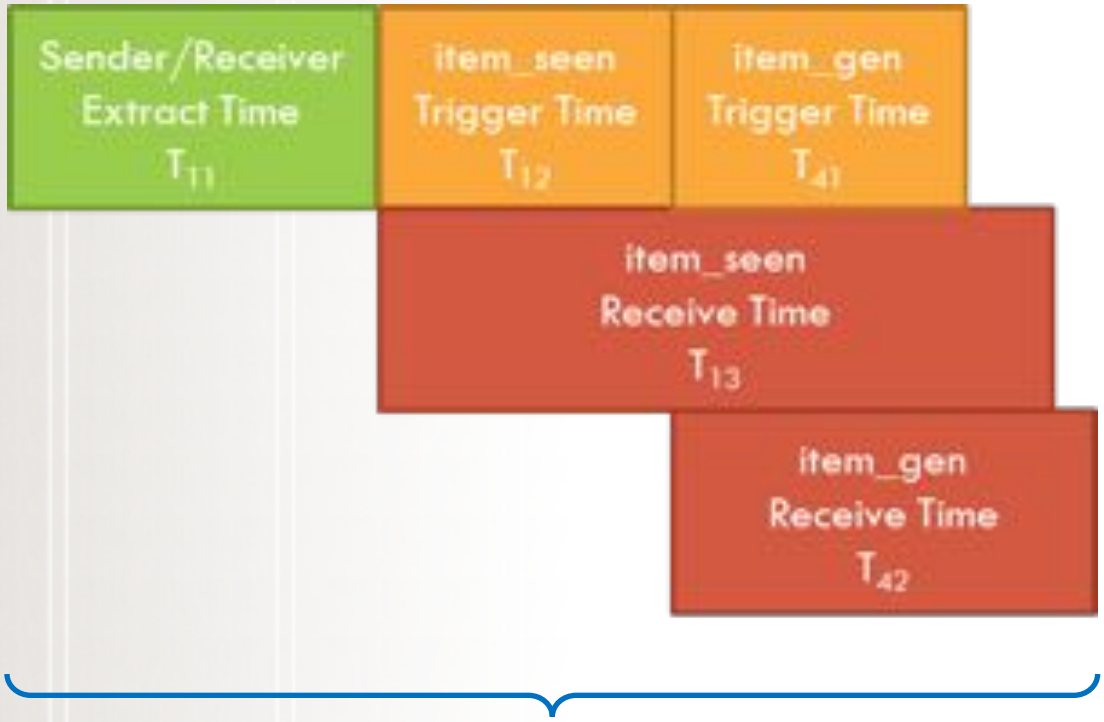
4 level



3 level

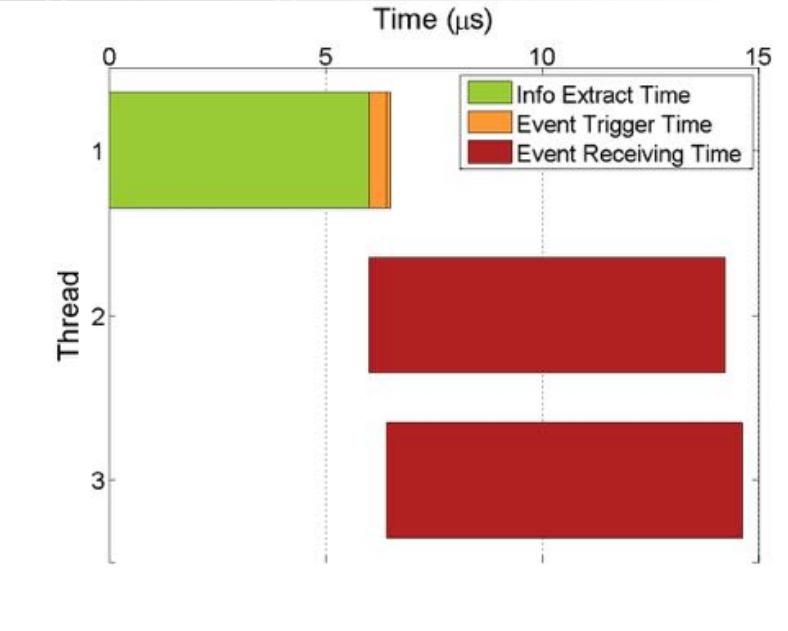
Traffic Statistics Collector

- 1-2 level

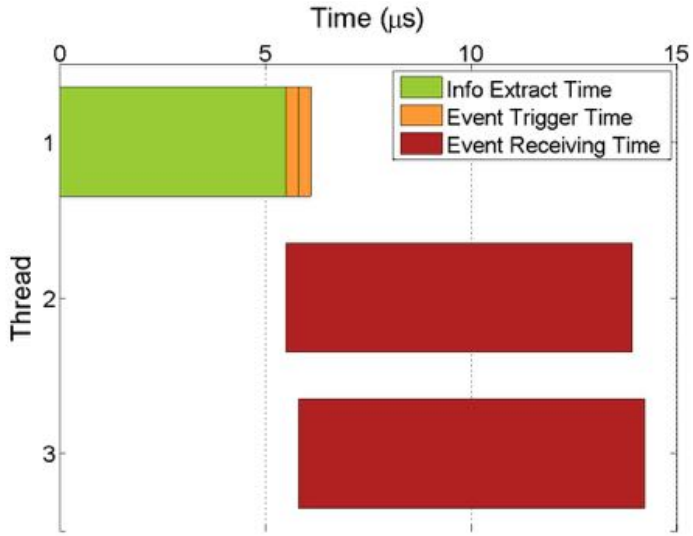


Traffic Statistics Collector

- 1-2 level



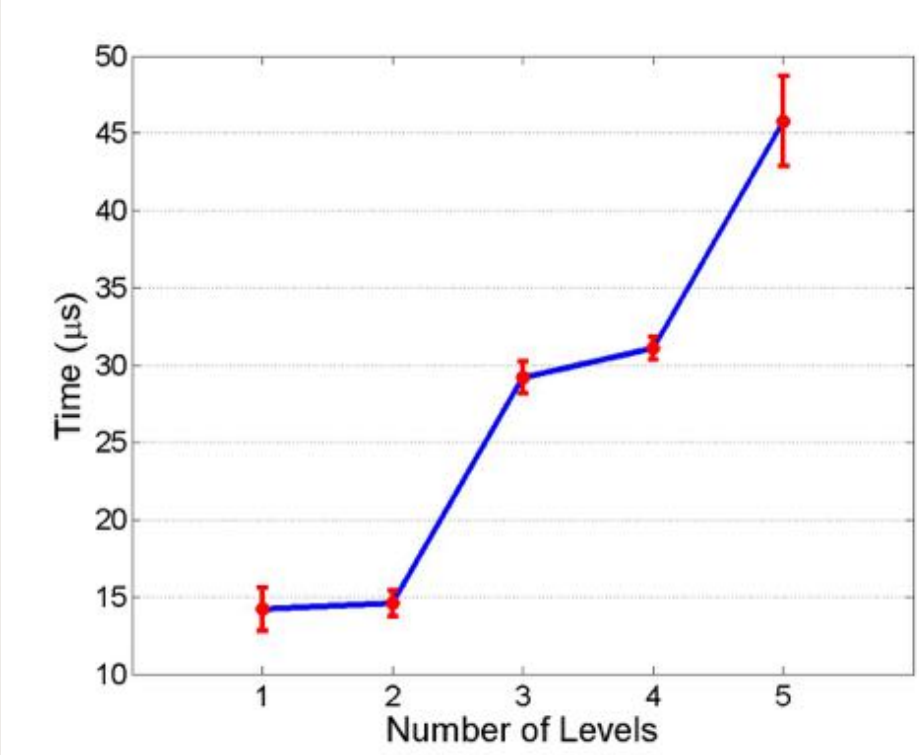
2 level



1 level

Traffic Statistics Collector

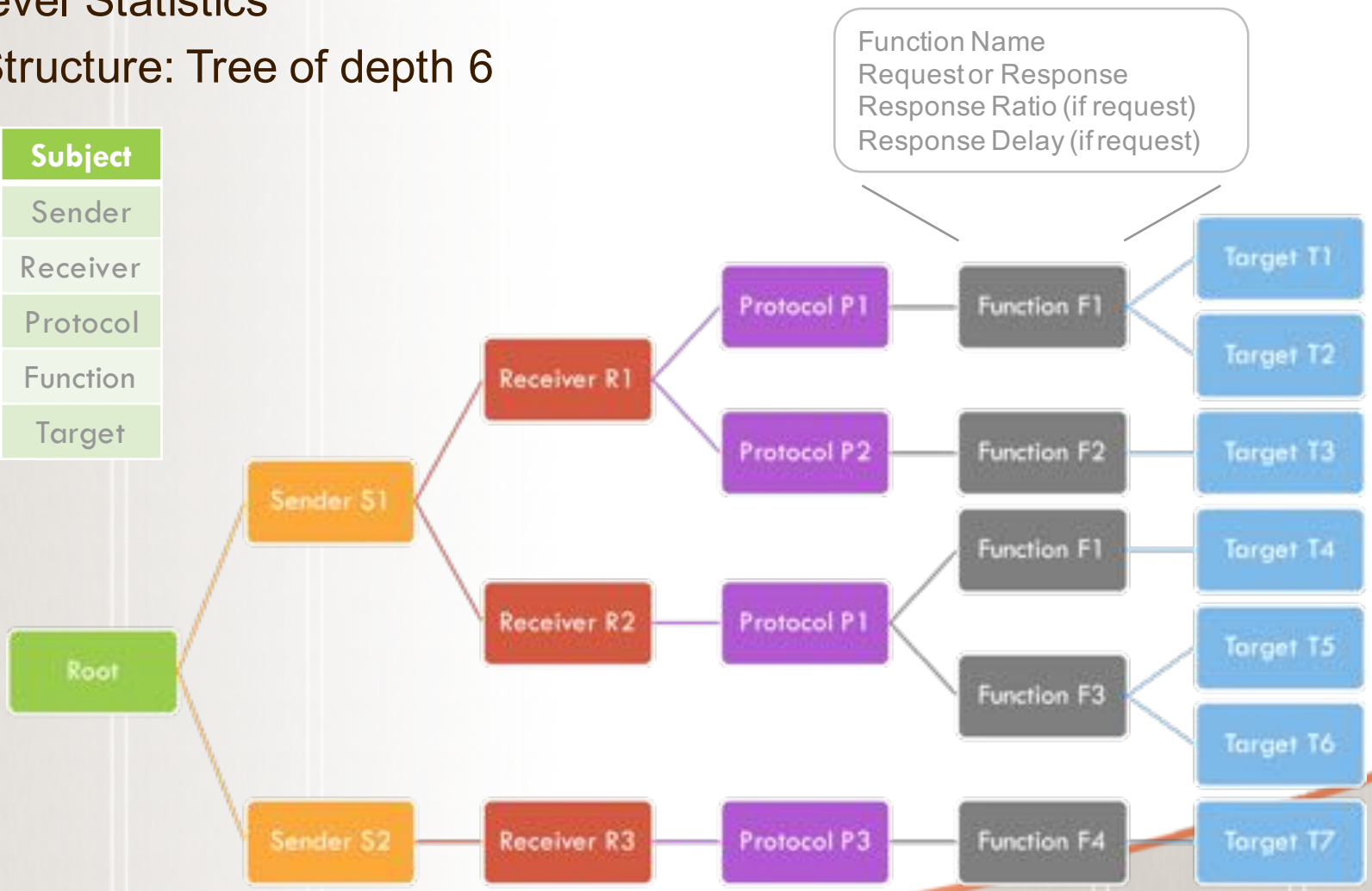
- Total Runtime



Traffic Statistics Counter

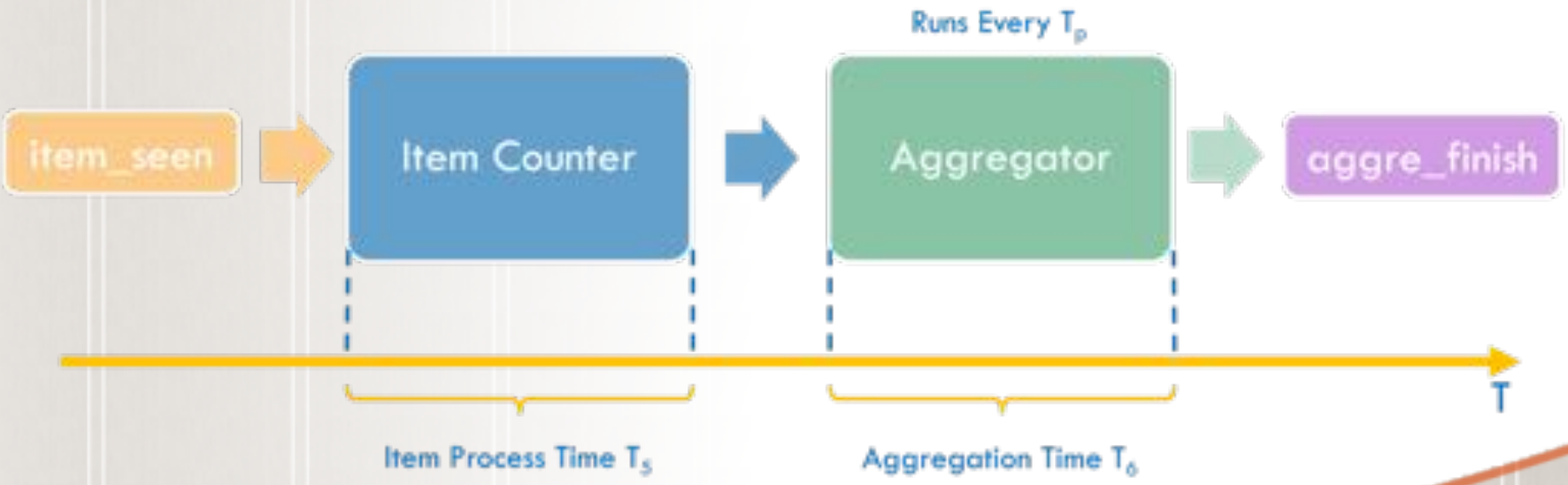
- Multi-level Statistics
- Data Structure: Tree of depth 6

Level	Subject
1	Sender
2	Receiver
3	Protocol
4	Function
5	Target



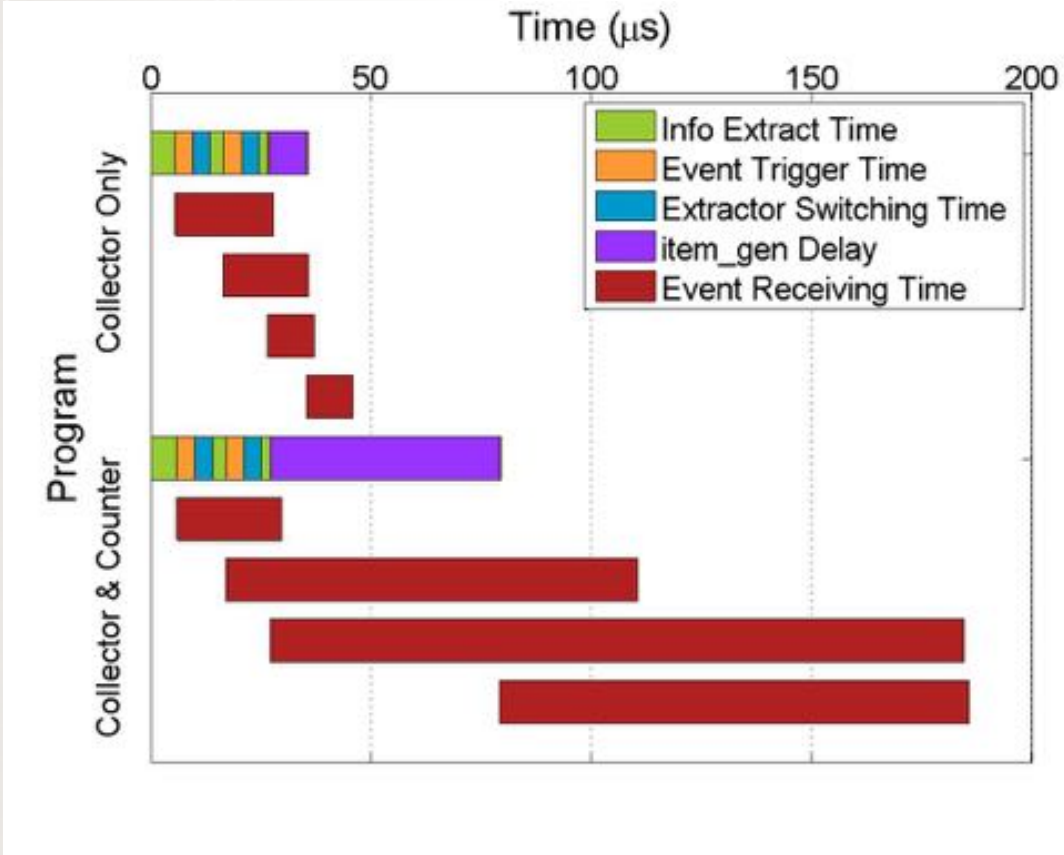
Traffic Statistics Counter

- Item process time T_5 is calculated per item_seen event. We further add all the item process time according to the same packet to calculate a total item process time per packet T_5'



Traffic Statistics Counter

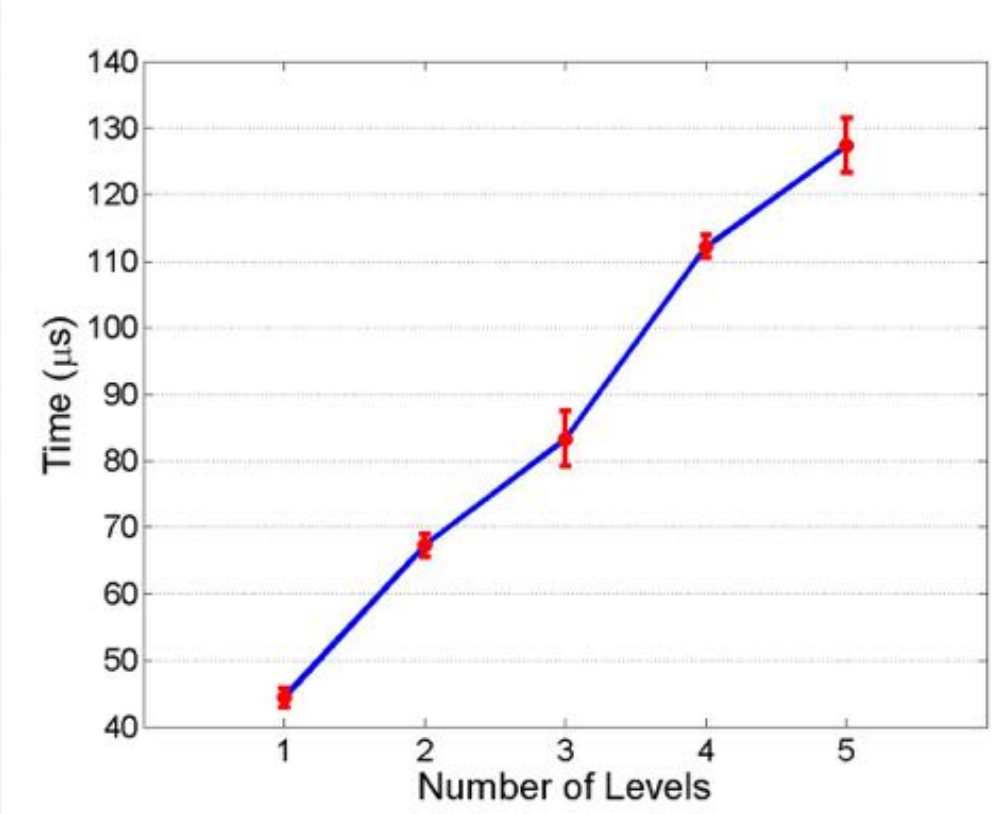
- Time flow comparison of the collector when running different programs



Traffic Statistics Counter

- Total item process time per packet with different number of levels

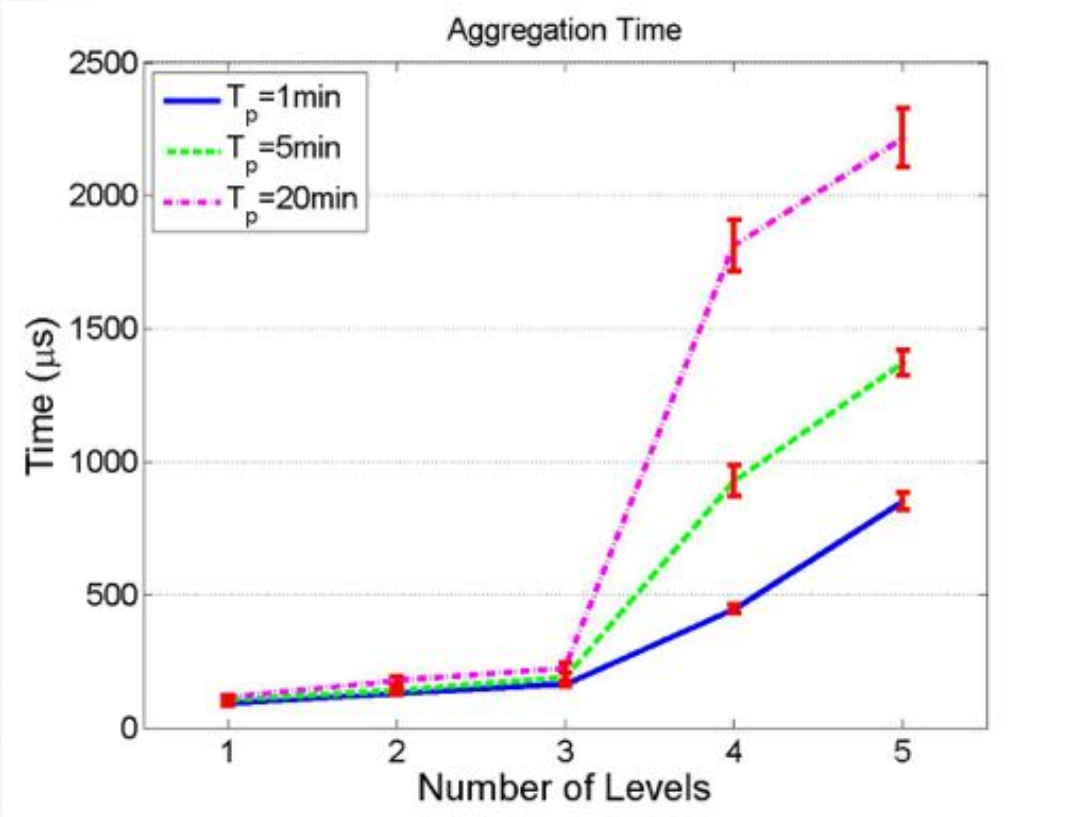
Subject	Number
Sender	8
Receiver	8
Protocol	1
Function	262
Target	37



Traffic Statistics Counter

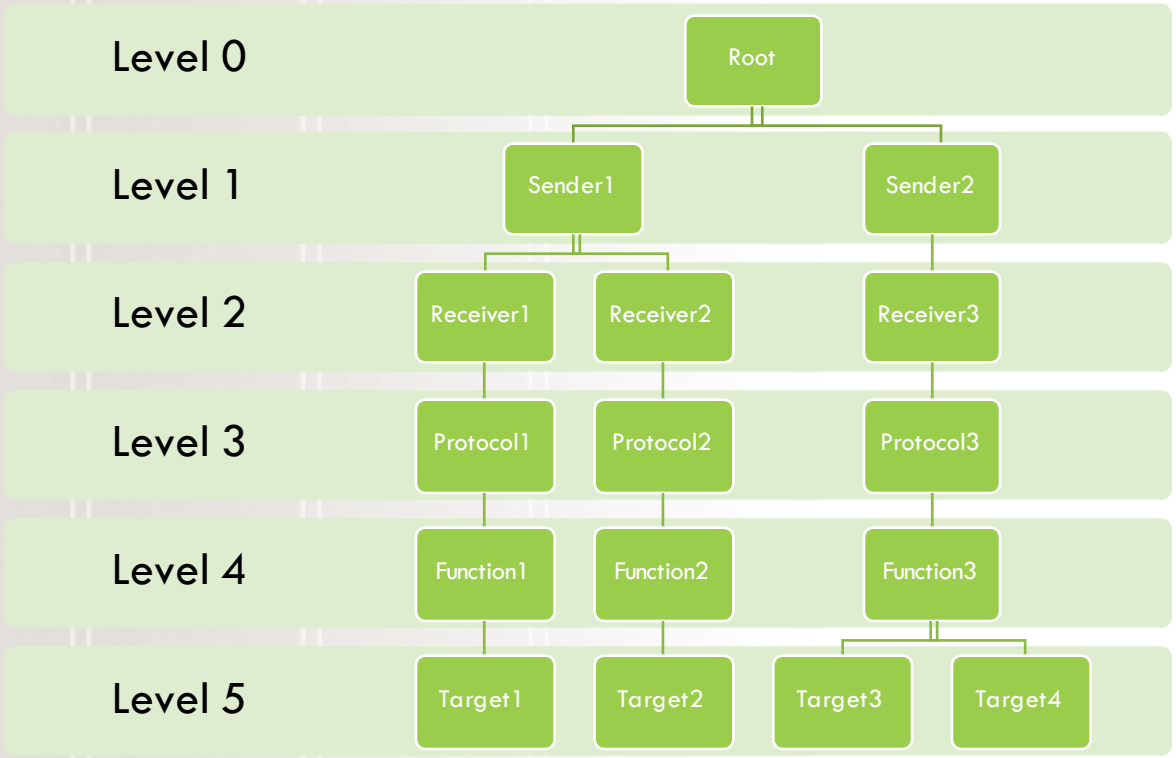
- Aggregation time with different number of levels and different aggregation period T_p

Subject	Number
Sender	8
Receiver	8
Protocol	1
Function	262
Target	37



Anomaly Detection Framework

- Each logging of traffic statistics will generate a data structure like a B-tree.



Each Node:

Count
Children[]

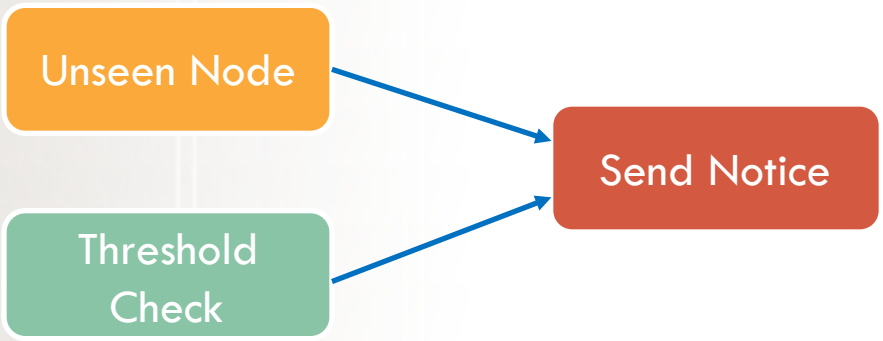
Count: Number of events represented by this node which happened in this logging period
Children[]: Pointers to children of this node

Anomaly Detection Framework

- SCADA traffic is periodic
- May vary in short time, but has a pattern over time.
- Construct "normal" tree and use it as a criterion

$$Tree_i \underbrace{Tree_{i+1}\ Tree_{i+2} \ldots Tree_{i+k}}_{Tree_{normal}}\ Tree_{i+k+1}$$

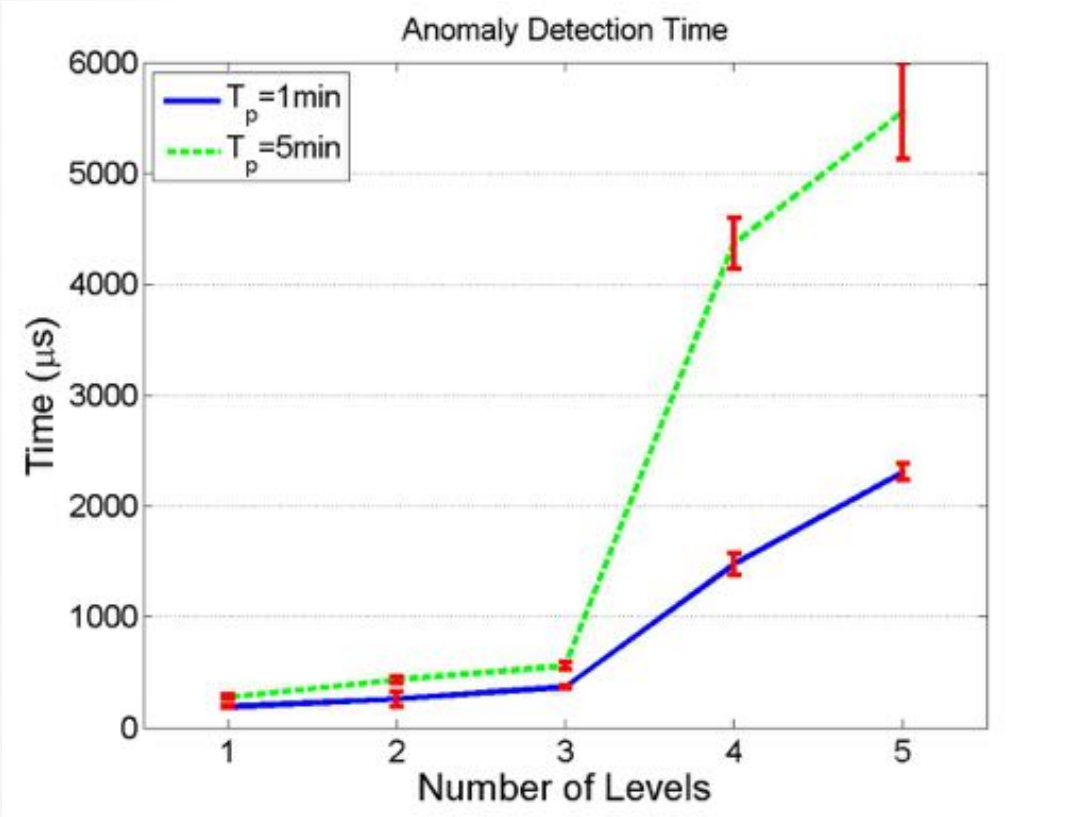
- When to send notice



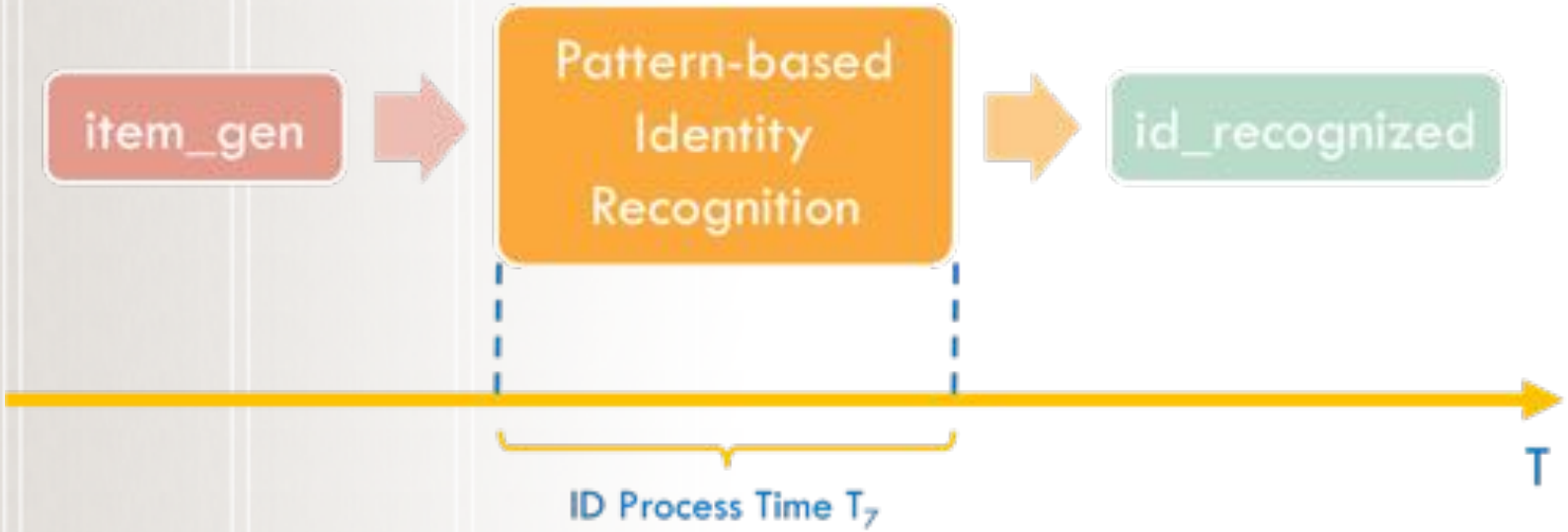
Anomaly Detection Framework

- Anomaly detection time with different number of levels and different aggregation period T_p

Subject	Number
Sender	8
Receiver	8
Protocol	1
Function	262
Target	37

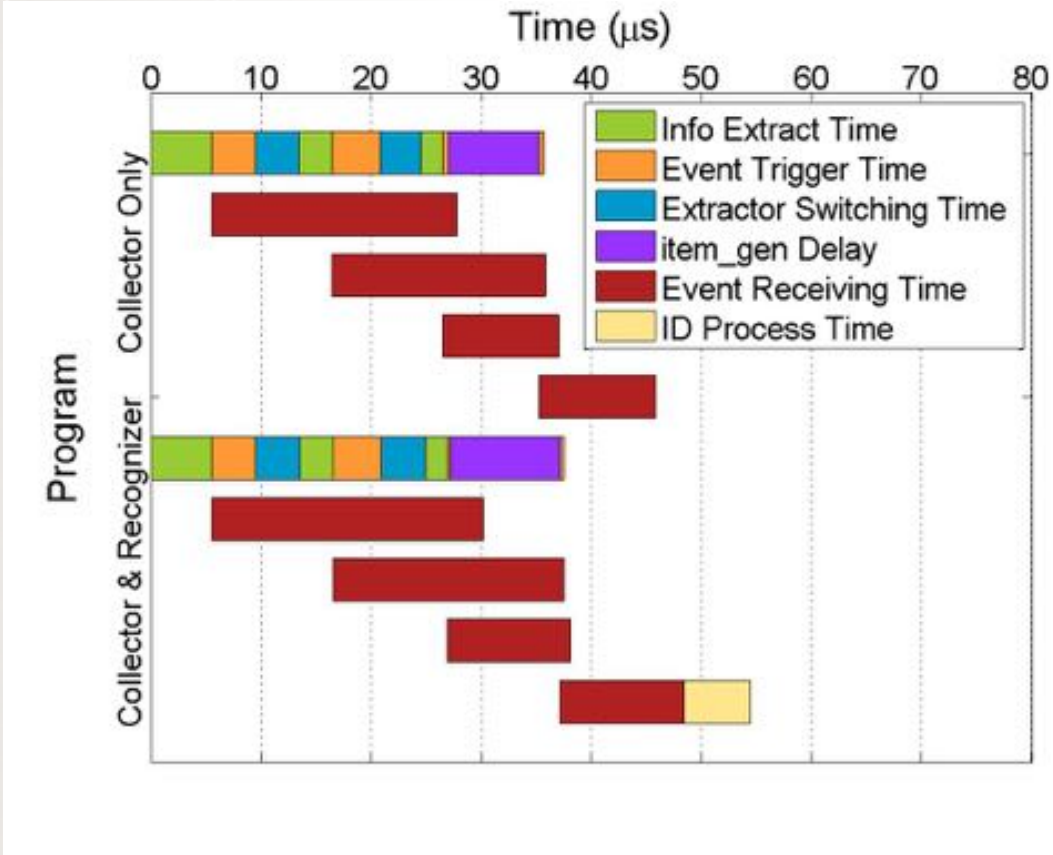


Pattern-Based Identity Recognition



Pattern-Based Identity Recognition

- Time flow comparison of the collector when running different programs



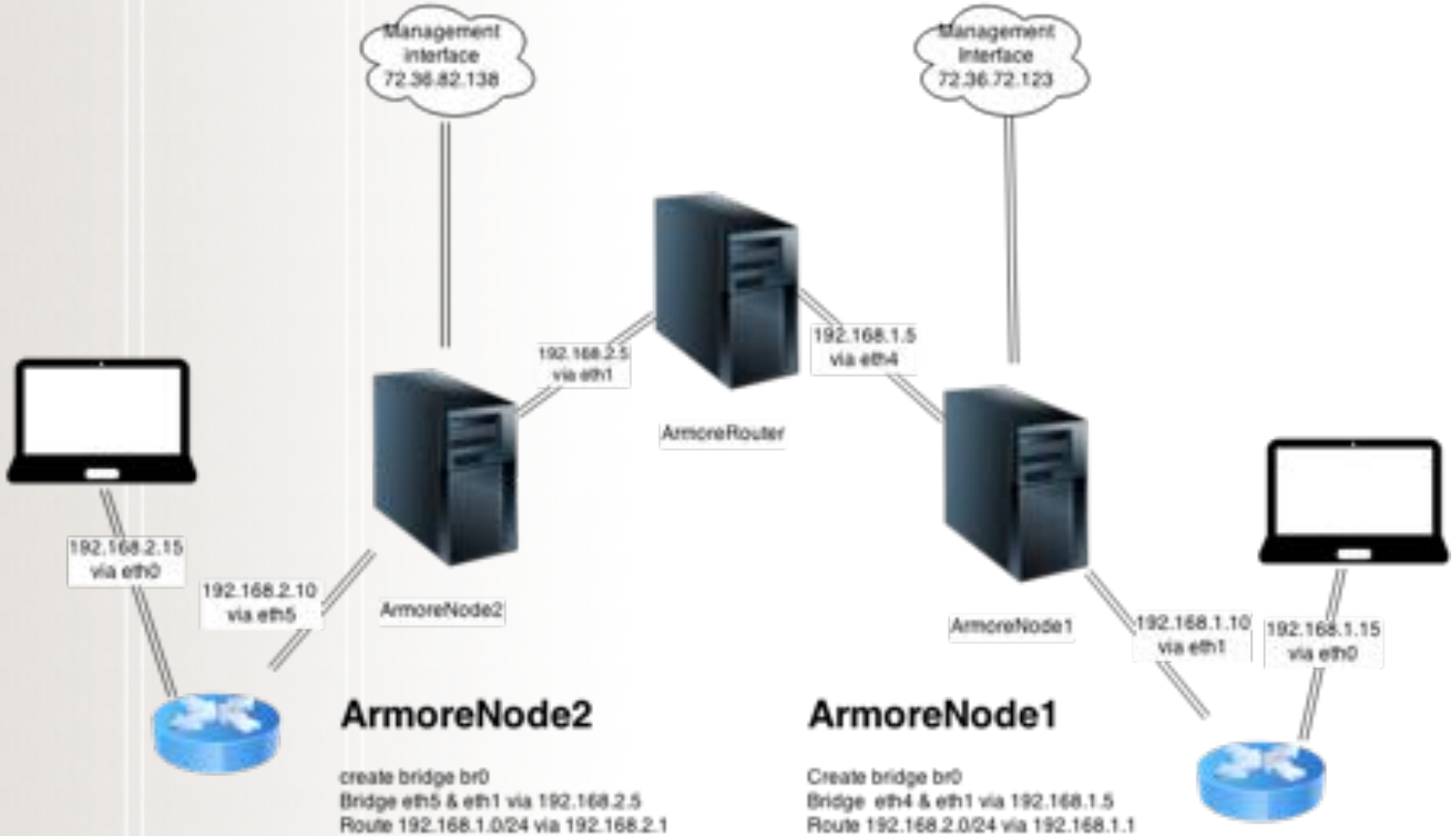
Example Uses of Analytics

- If one can inspect the communications, one can observe patterns and behaviors
  - *E.g., DNP3 SBO message, with affirmative response…*
    - *Probably a relay*
- With inspection, one can then enforce
- What's going on in your network?
- Future planning
- Encryption
- Fault-tolerance

System Testing

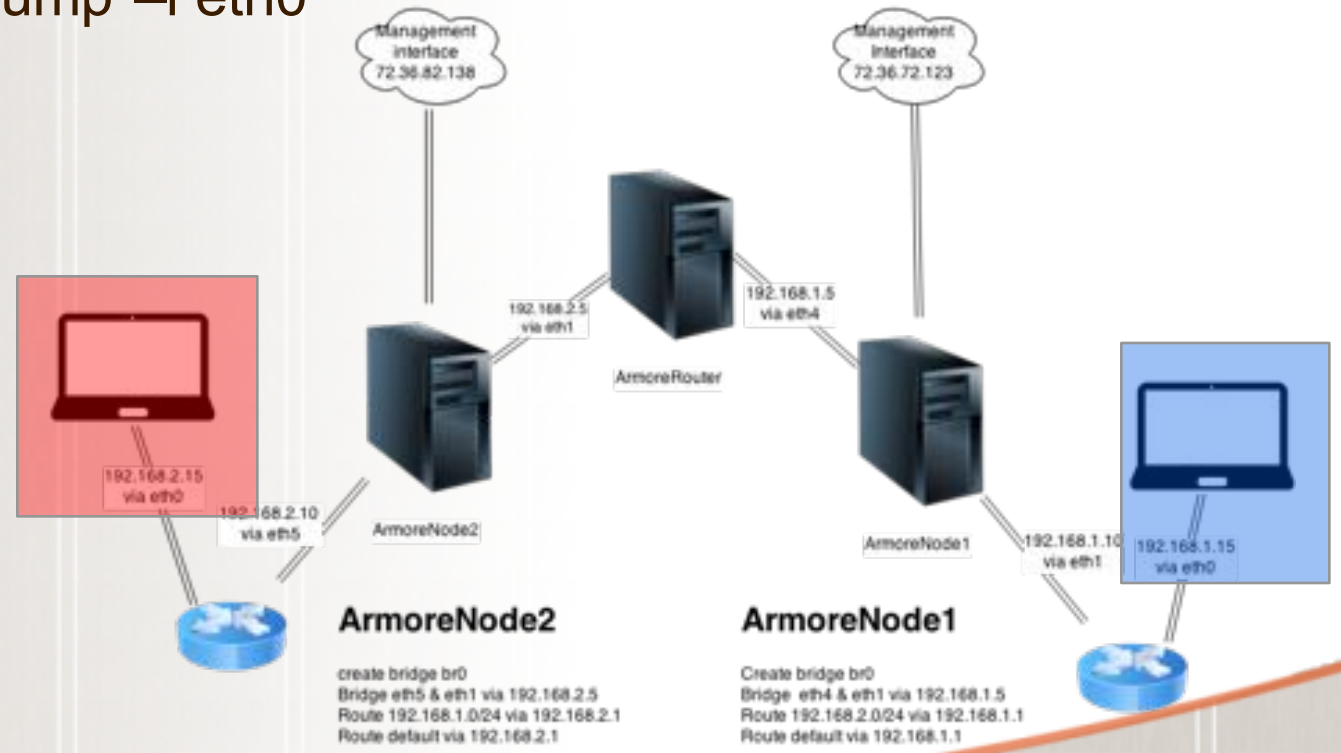
Work by Chris Drew and Steve Granda

Physical Test bed Overview



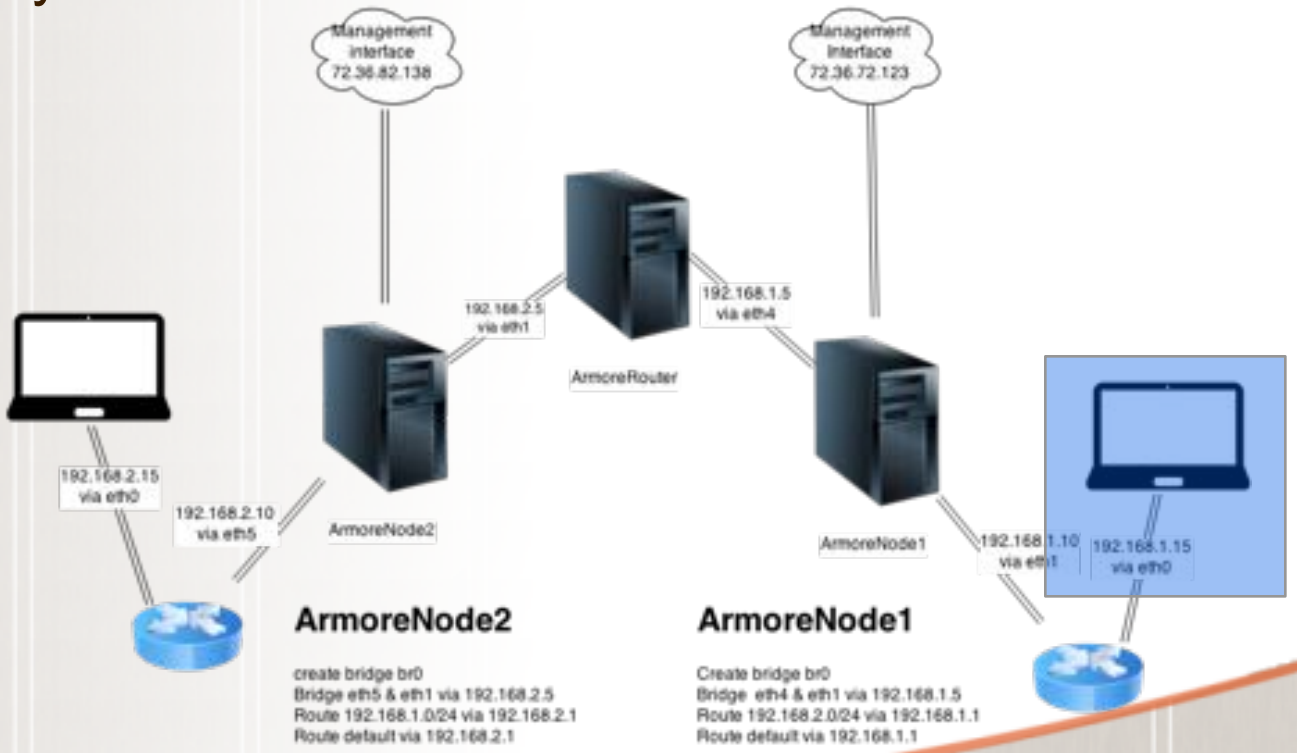
Proxy Testing Procedures

- With the armoreconfig service running on armorenodes we generate traffic with iperf on the blue node to the red node.
  Iperf –c 192.168.2.15 –i 1
- In the above example we'll listen for generated traffic with tcpdump on red node:
  tcpdump –i eth0



Testing Procedures

- Once the ARMORE Nodes are configured we use the pkt-gen script to send data on a netmap pipe:
  ./pkt-gen –i netmap:eth0}0 –f rx
- Anything listening on netmap pipe **eth0}0** should be able to transparently receive data.



Server Room


SEL
137:14:42

TCIPG
SMART METER RESEARCH PLATFORM

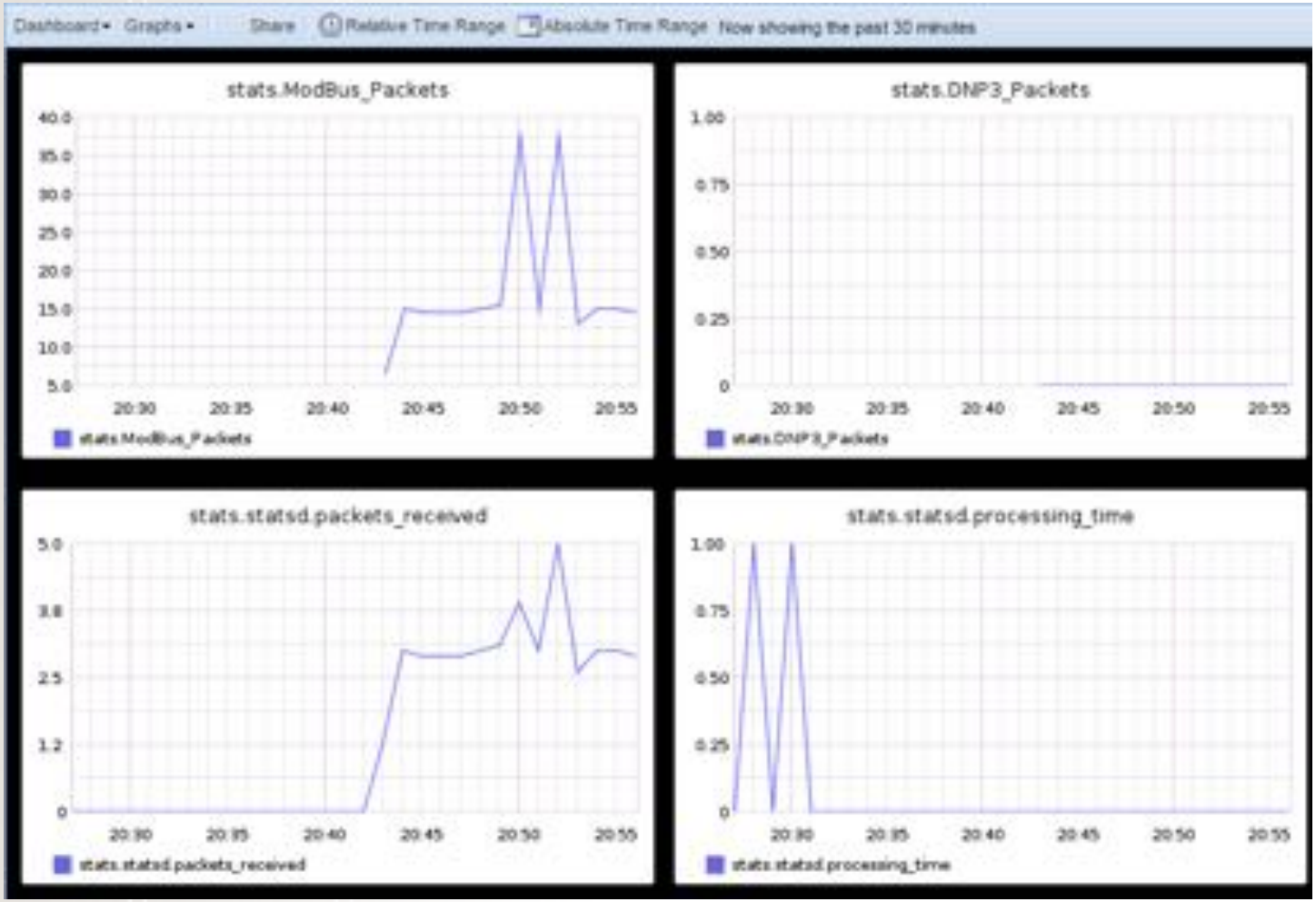
DANGER PELIGRO

PASS
FAIL
ALARM
TEMP
BAT LOW
POWER MODULE
MODEL 8310
120VAC/DC

SCADA/ICS Testing

- DNP3 and Modbus Protocol Test Harnesses
  - *Will generate typical traffic and verify back and forth connectivity*
  - *Leveraging open stacks for implementation*
  - *Might also be able to leverage compliance testing suites*

ModBus Traffic Visualization



Future Work

- Example policy creation
  - *And "policy builder"*
- Enforcement actions
  - *iptables hooks*
- More advanced analytics processing
  - *Smarter anomaly detection*
  - *Passive device profiling and determination*
  - *Network mapping*
- Integration with Debian 8 Jessie x64
  - *More testing needs to be done with systemd and 4.0 Linux Kernel before pushing to our repository.*
- Bro/Broker
  - *Broccoli is being phasing out and will be replaced by Broker.*
- Visualization
  - *Integration of web base monitoring with bro-statsd to aid in monitoring traffic of an ARMORE node.*

Interested?

yardley@illinois.edu