GRID PROTECTION ALLIANCE

ARMORE

Applied Resiliency for More Trustworthy Grid Operation

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

GRID PROTECTION ALLIANCE

Pacific Northwest NATIONAL LABORATORY

DEPARTMENT OF ENERGY · UNITED STATES OF AMERICA ·

GPA User's Forum 2015

Atlanta, Georgia

Motivation for ARMORE

- 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

ARMORE Design Objectives

- Security appliance 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

Major Software Components

- Operating System – Linux
  - Debian Wheezy 7.8 x64 - Modified 3.12.0 Linux Kernel
- Bro - INSPECTOR – Dynamic Traffic Analyzer
- NetMap
  - Kernel Module for High Speed Packet I/O
- MANAGER - Administrator's Interface

Core Components

- Bro – ENFORCER - Policy Enforcement
- PROXY (new GPA OSS)
  - Transparent data stream encapsulator
  - ZeroMQ with Curve security
  - Copious logging

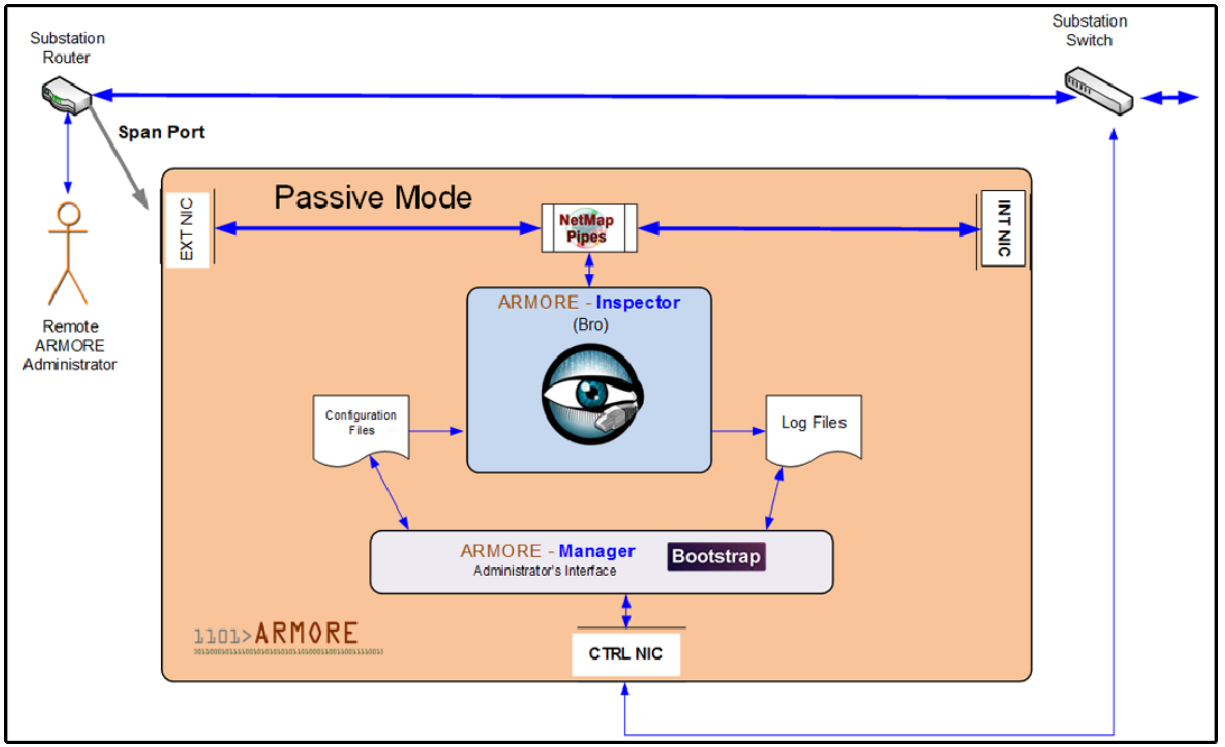
For Active Modes Only

Implementation Options

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

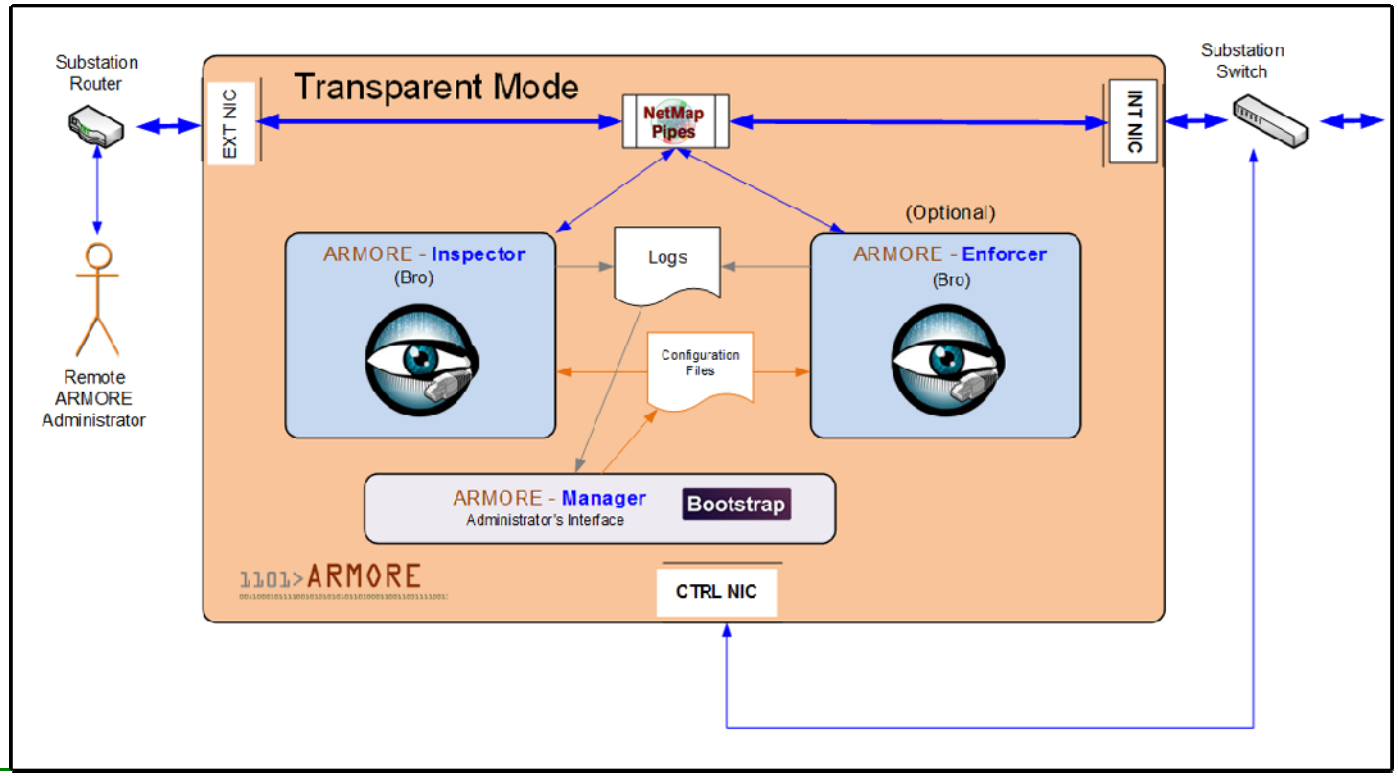
Passive Implementation

- Payload inspection
- Network visibility and intelligence
- Network analytics enablement



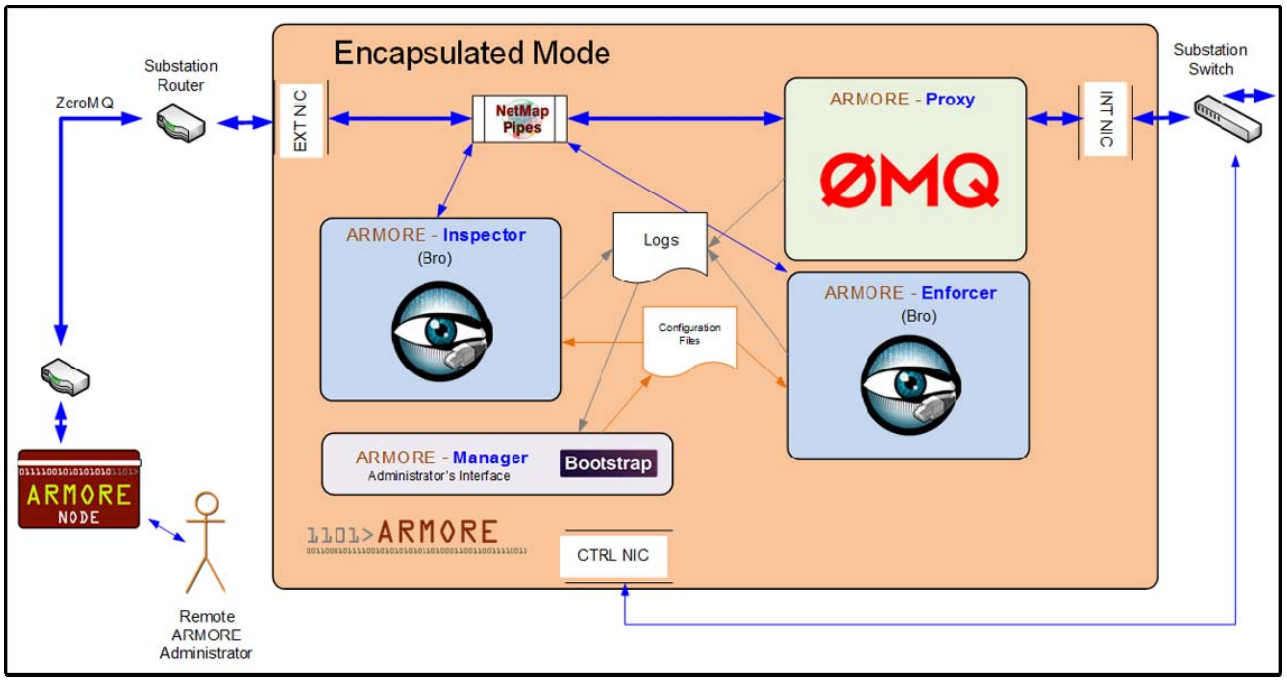
Transparent Implementation

- Passive plus…
- Communication endpoints operate without any changes
- Optional policy enforce-ment

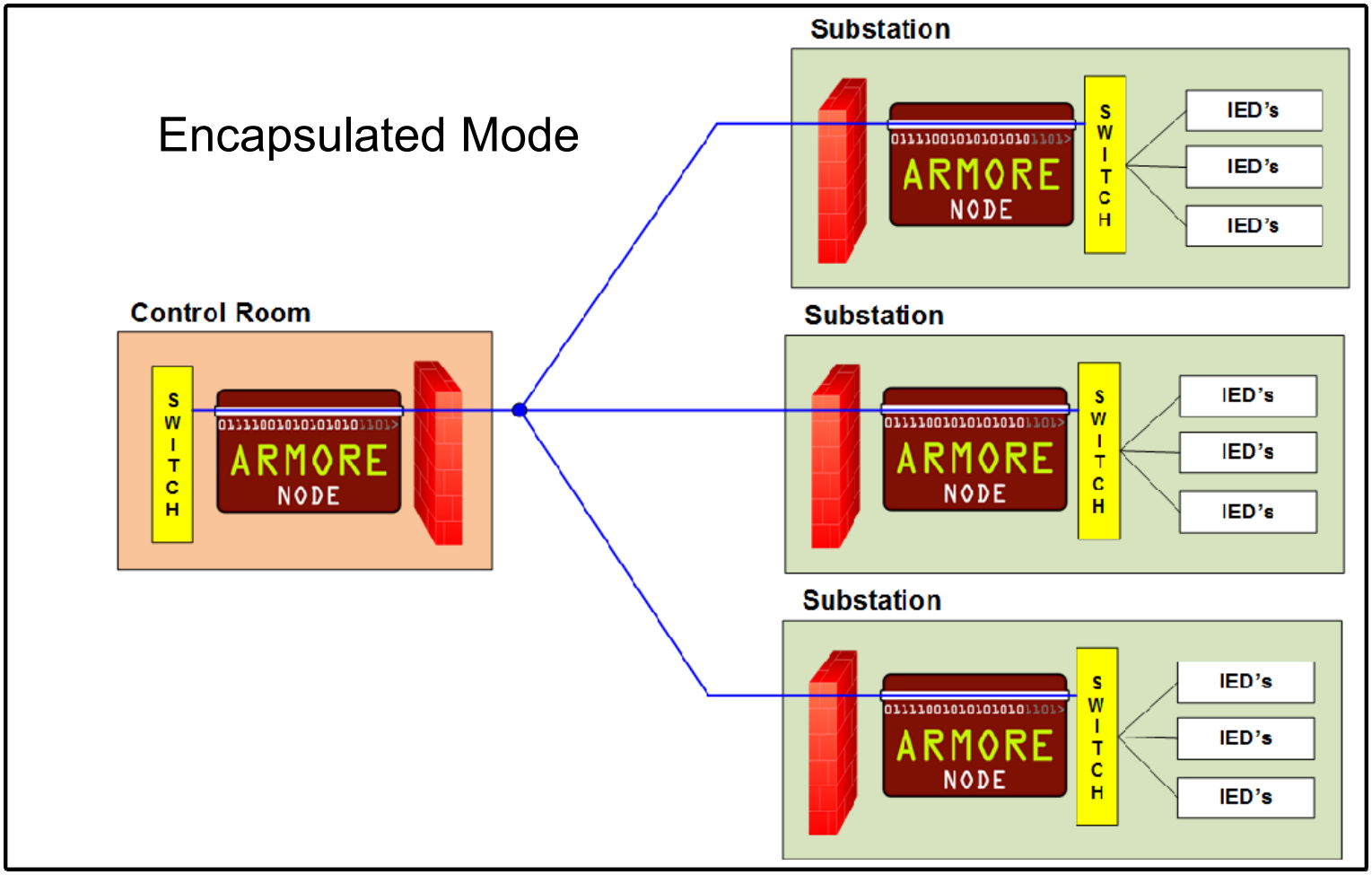


Encapsulated Implementation

- Transparent plus…
- Encapsulation and encryption
- Security augmentation (access filtering)
- Optional policy enforce-ment
- Fault tolerance



Full Implementation



ARMORE Proxy

- Abstract class for middleware library inclusion
  - ZeroMQ implemented with Curve security
  - DDS stubbed but not implemented
- Abstract packet capture interface
  - PCAP
  - Netmap
- Many options for logging
- MAC address translation mode

DDS vs. ZeroMQ

DDS

- Commercial options
- No open source security
- Extensive functionality built in
- Steep learning curve
- Slightly more resource heavy
- 4 languages
- Restricted to pub/sub

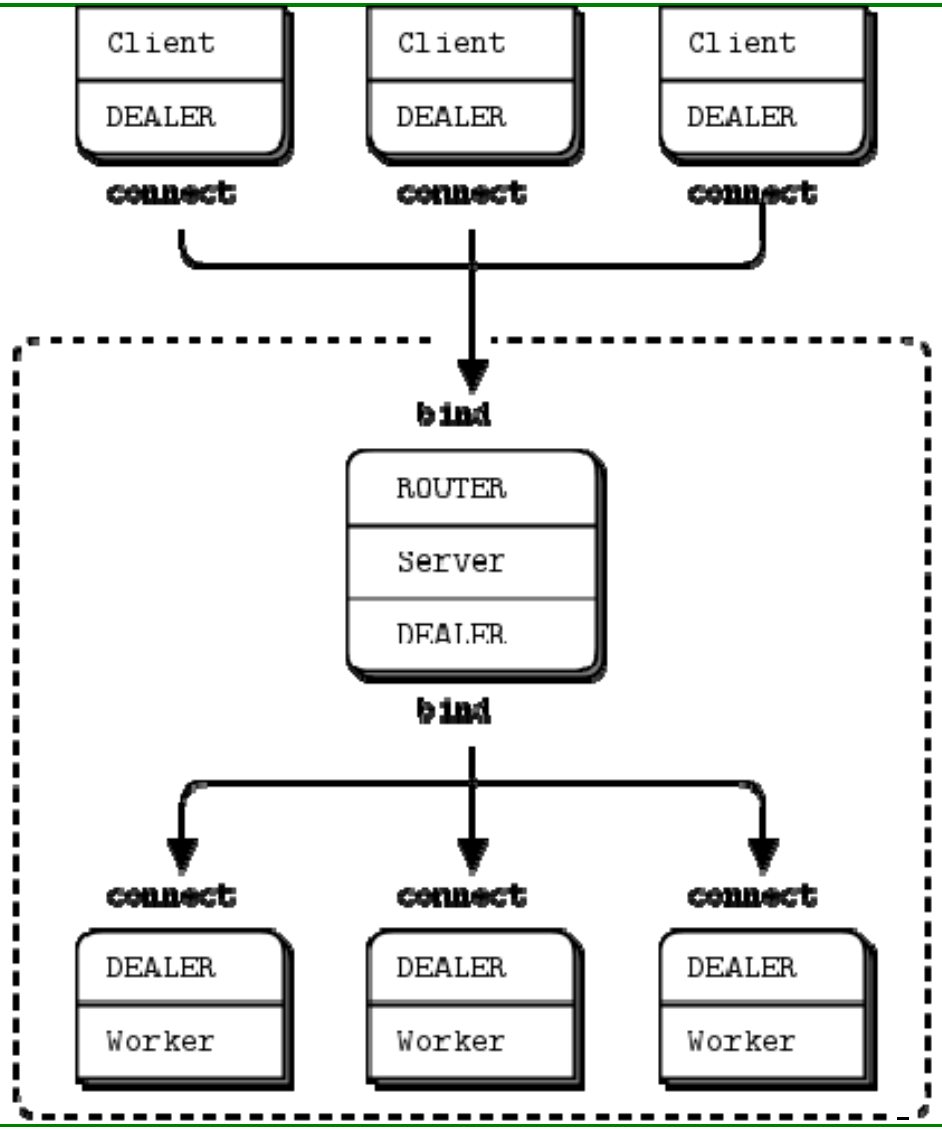
ZeroMQ

- Exclusively open source
- Sufficient features for ARMORE
- Easy to learn
- Lightweight
- 30+ languages
- Multiple pattern flexibility

Zero MQ

- 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

ZeroMQ Dealer/Router Pattern



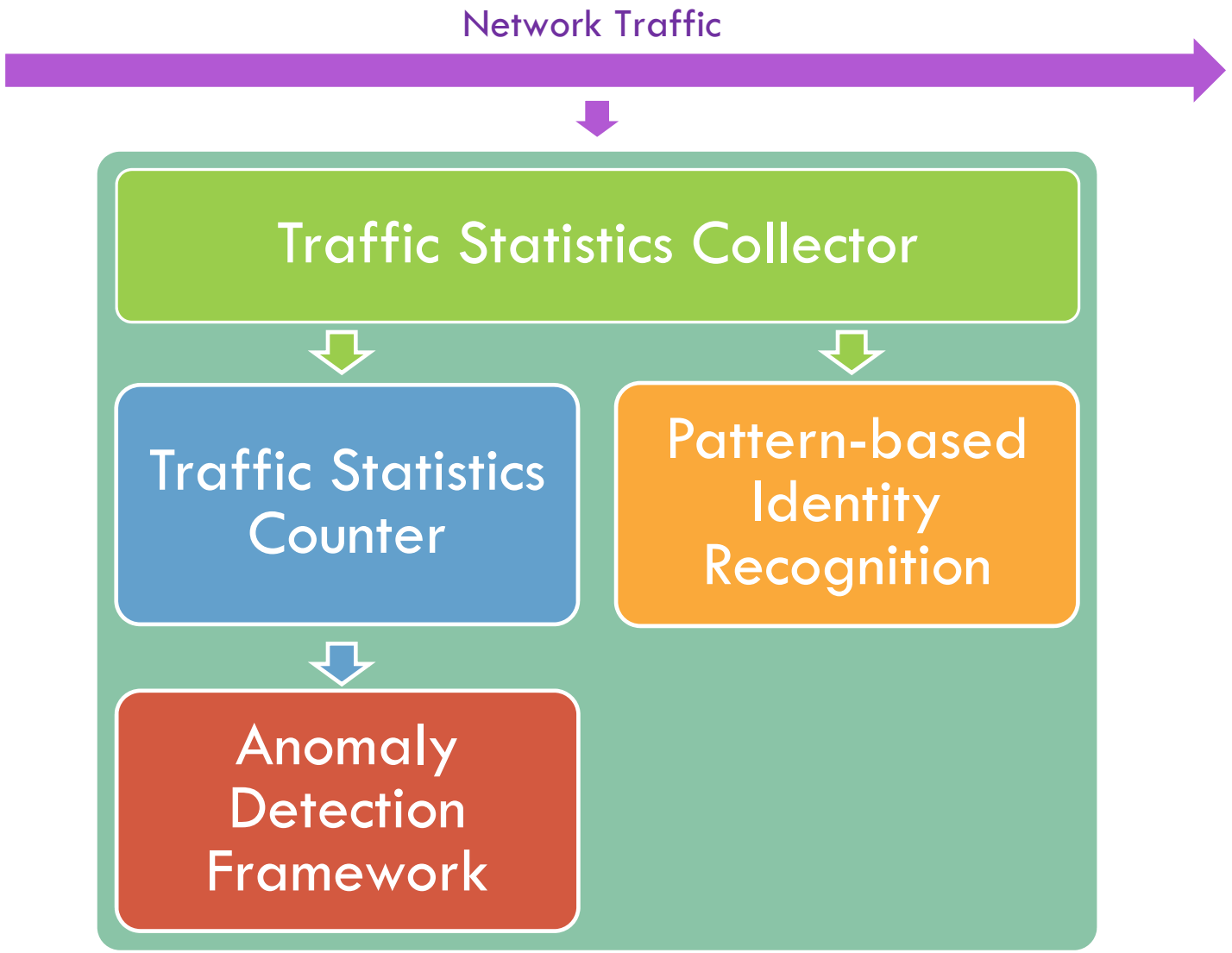
Administrator's Interface

- Front end connects UI with ARMORE node internals. Based on Bootstrap.
  - 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

Dynamic Traffic Analyzer

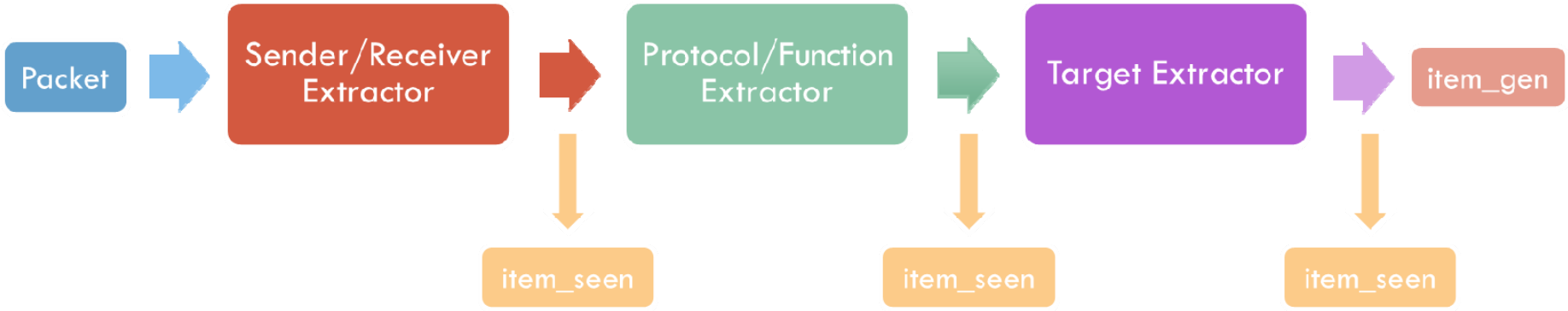
- What is it?
  - An analyzer that provides dynamic and intelligent analytics for SCADA protocols, increasing visibility into the system behavior
- What is it using?
  - Bro's scripting engine
- What protocols does it support at the moment?
  - ✓ DNP3
  - ✓ Modbus
  - ✓ Extensible to any other protocol

Components

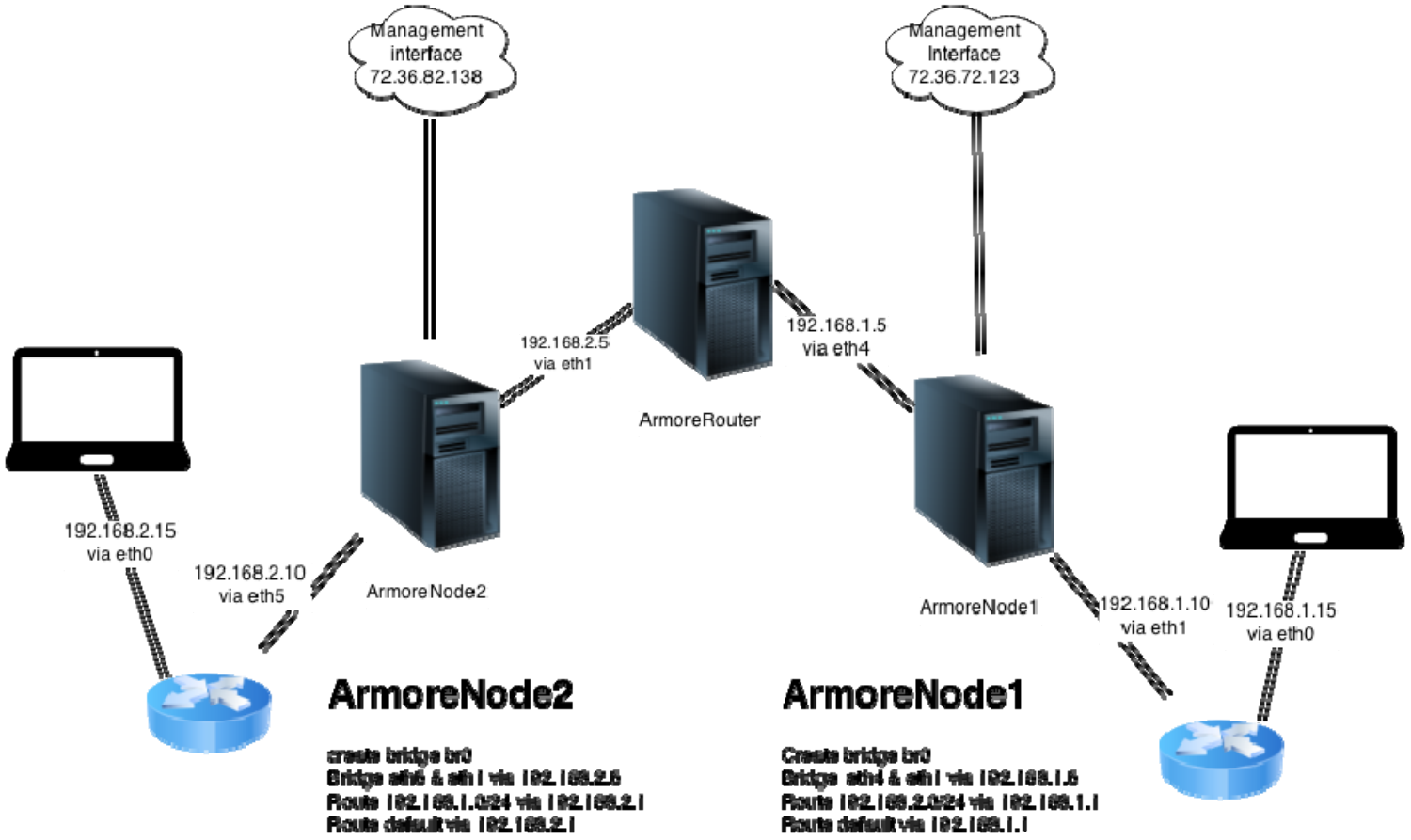

Network Traffic
Traffic Statistics Collector
Traffic Statistics Counter
Pattern-based Identity Recognition
Anomaly Detection Framework

Data Flow

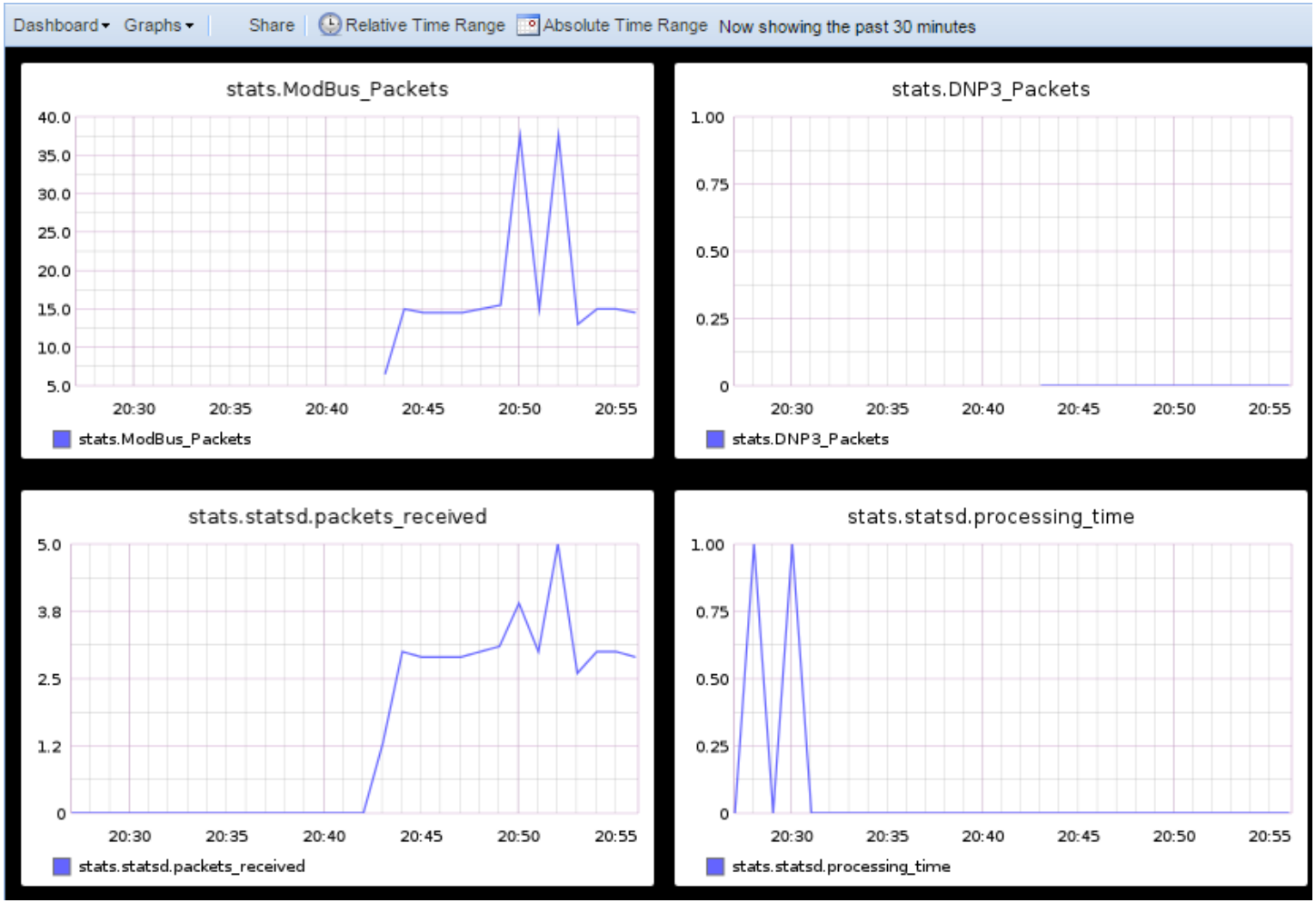
- Input: network traffic
- Output: two kinds of events
  - item_seen: instantaneous, item contains incomplete information of the packet
  - item_gen: delayed, item contains complete information of the packet



UIUC INSPECTOR (Bro) Test



ModBus Traffic Visualization



GPA PROXY Test