Introduction to Argus

http://qosient.com/argus

FloCon 2010
New Orleans, La
Jan 11, 2010

Carter Bullard
QoSient, LLC

carter@qosient.com

Carter Bullard carter@qosient.com

- QoSient - Research and Development Company
  - Naval Research Laboratory (NRL), GIG-EF, JCTD-LD, DISA, DoD Network Performance and Security Research
- Inventor/Developer Argus http://qosient.com/argus
- FBI/CALEA Data Wire-Tapping Working Group
- QoS/Security Network Management - Nortel/Bay
- Security Product Manager – FORE Systems
- CMU/SEI CERT
  - Network Intrusion Research and Analysis
  - NAP Site Security Policy Development
  - Network Security Incident Coordinator
- NFSnet Core Administrator (SURAnet)
- Standards Efforts
  - Editor of ATM Forum Security Signaling Standards
  - IETF Working Group(s) Contributor
  - Internet2 Security WG
  - NANOG

Argus

- Argus is a network utilization audit system

  Argus was officially started at the CERT-CC as a tool in incident analysis and intrusion research. It was recognized very early that Internet technology had very poor usage accountability, and Argus was a prototype project to demonstrate Red Book strategies for LAN and CAN network auditing.

- Composed of
  - Real-time Network flow monitor
  - Network flow data collection system
  - Network flow data processing programs
  - Audit data repository tools

Argus History

- Georgia Tech (1986)

  Argus was the first network flow data system. Started at Georgia Tech, Argus was used as a real-time network operations and security management tool. Argus monitored the Morris Worm, and was instrumental in discovery for the "Legion of Doom" hacking investigations.

- CERT/SEI/Carnegie Mellon University (1991)

  Argus was officially supported by the CERT as a tool in incident analysis and intrusion research. Used to catalog and annotate any packet file that was provided to the CERT in support of Incident Analysis and Coordination, it was a focal point for research in intrusion analysis and Internet security.

- Argus Open Source (1995)

  Transitioned into public domain in 1995. Supported by CMU and CERT/SEI at many levels including argus developers mailing list.

  Used now by a large number of educational, commercial and governmental sites for network operations, security and performance management.

Argus Licensing

- GNU GPL 3
- US DoD open source license
  - Gargoyle
    - Available from https://software.forge.mil
    - Growing development community
- Alternate licensing available

Disclaimers

- Argus is a proof-of-concept project.
  - Argus and its clients are examples of what can be done
  - Any concept, any time, as long as it fits
- Argus is NOT Netflow™
  - There are a lot of network flow data methodologies
  - Community needs to realize Netflow is Cisco's approach
  - Let's stop using Netflow as term for Network Flow Data
- Argus is NOT IPFIX
  - Lots and lots and lots of issues with IPFIX
  - Argus does attempt to avoid or resolve basic IPFIX problems

Introduction to Argus

- Argus Design (20m)
- Data Generation (25m)
- Client Programs (45m)
- Data Collection and Archiving (45m)
- Situational Awareness (45m)

Argus Design

Argus Design

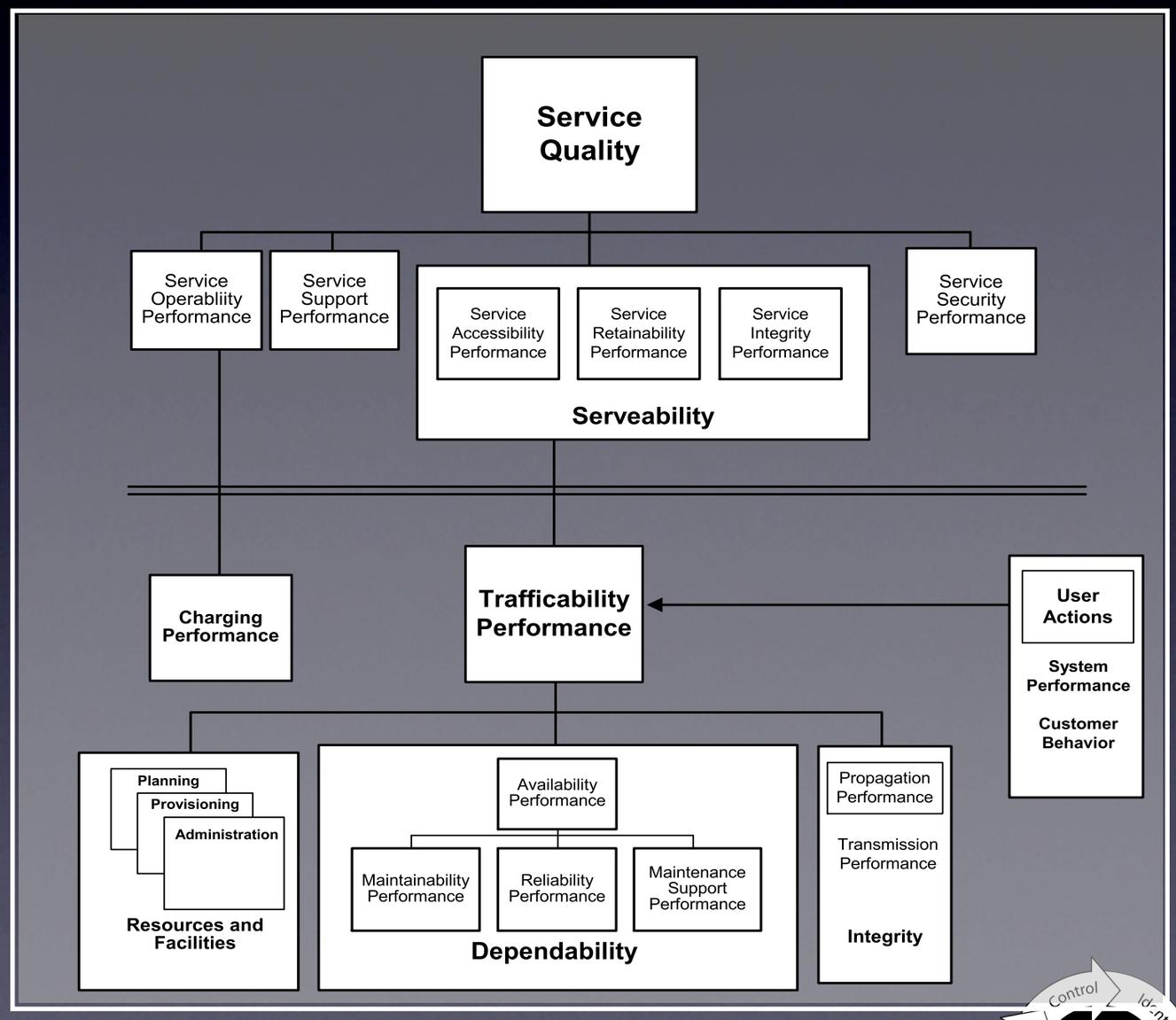
- Comprehensive Network Accountability
- Based on Tried and True Methodology
- High Utility/Applicability
- High Performance
- Deployable / Scalable

Comprehensive Network Accountability

- Ability to account for all/any network use
  - Red Book prescribed method for trusted networking
- At a level of abstraction that is useful
  - Network Service Functional Assurance
    - Was the network service available?
    - Was the service request appropriate?
    - Did the traffic come and go appropriately?
    - Did the traffic get the treatment it was support to?
    - Did the service start and end normally?
  - Network Control Assurance
    - Is the control plane operational?
    - Was the service request appropriate?
    - Did the traffic come and go appropriately?

Tried and True Methodology

- ITU Network Service Quality and Usage strategies
- Support mature network service measurement architectures
- Passive, comprehensive, service oriented, integrated, shareable, extensible, accessible



From ITU-T Recommendation E.800 Quality of Service, Network Management and Traffic Engineering

Network Measurement Architectures

- Examples
  - ITU TMN, X.700, OSI, SPAND, EMA, NIMI, Optivity, NetView, UNMA, PRIMA, MFN, SNMP to name just a very few.
  - Strategies involve either pure passive, or a bundle of active and passive methods.
- Component Design
  - All measurements, whether passive, active, extractive, or injective, involve a passive measurement component
  - Argus is designed to be that passive component
- Systems Design
  - Data Generation, Collection, Disposition, Access
  - May need to provide it all.

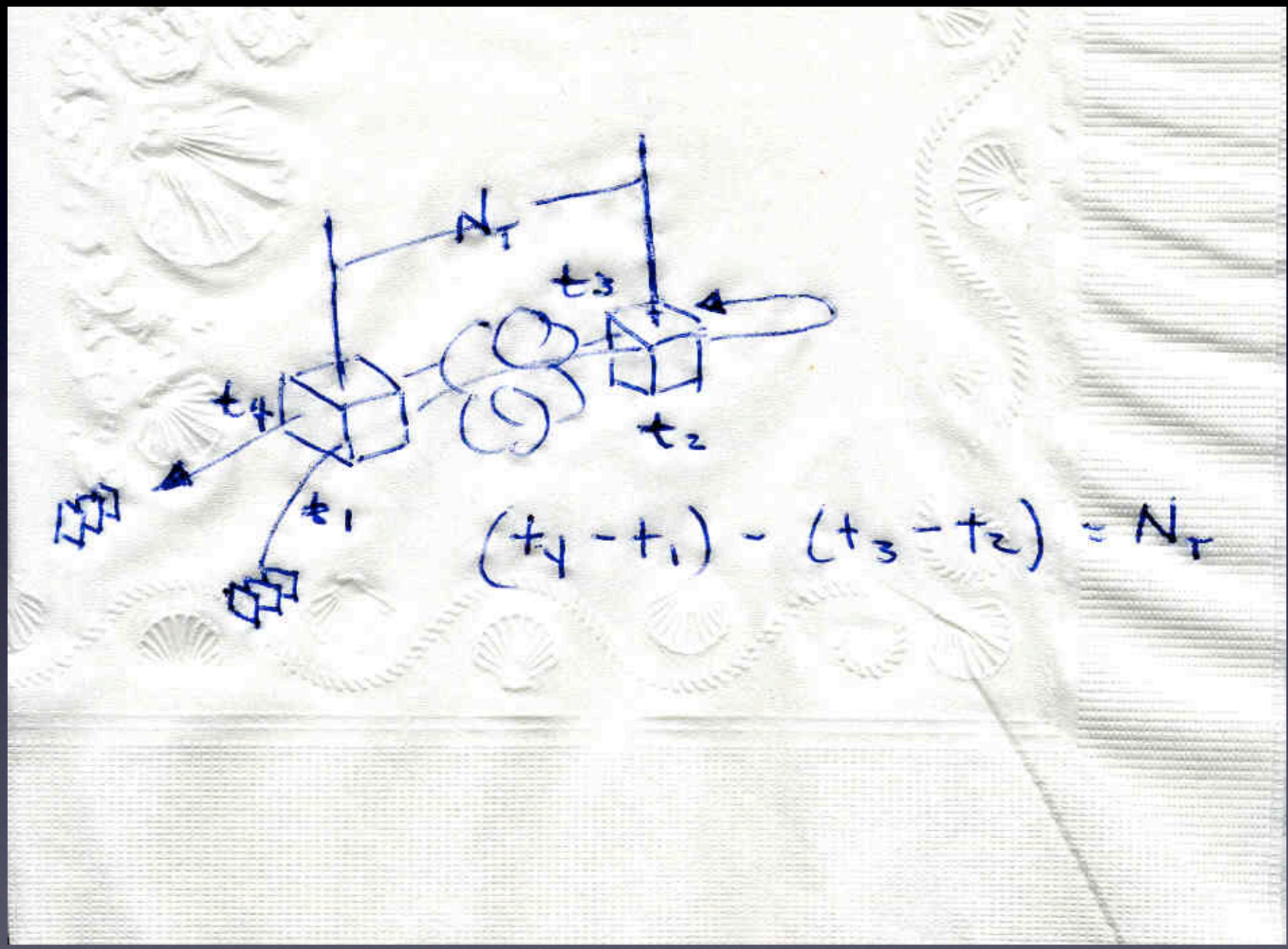
Utility/Applicability

Feedback Directed Network Optimization

Function	Description	
Identify	Discover and Identify comprehensive network behavior	Collect and process network behavioral data
Analyze	Collect and transform data into optimization metrics, establish baselines occurrence probabilities and prioritize events.	
Plan	Establish optimization criteria, both present and future and implement actions, if needed	Provide information and feedback internal and external to the project on the optimization outcomes as events.
Track	Monitor network behavioral indicators to realize an effect.	
Control	Correct for deviations from criteria.	

Utility/Applicability

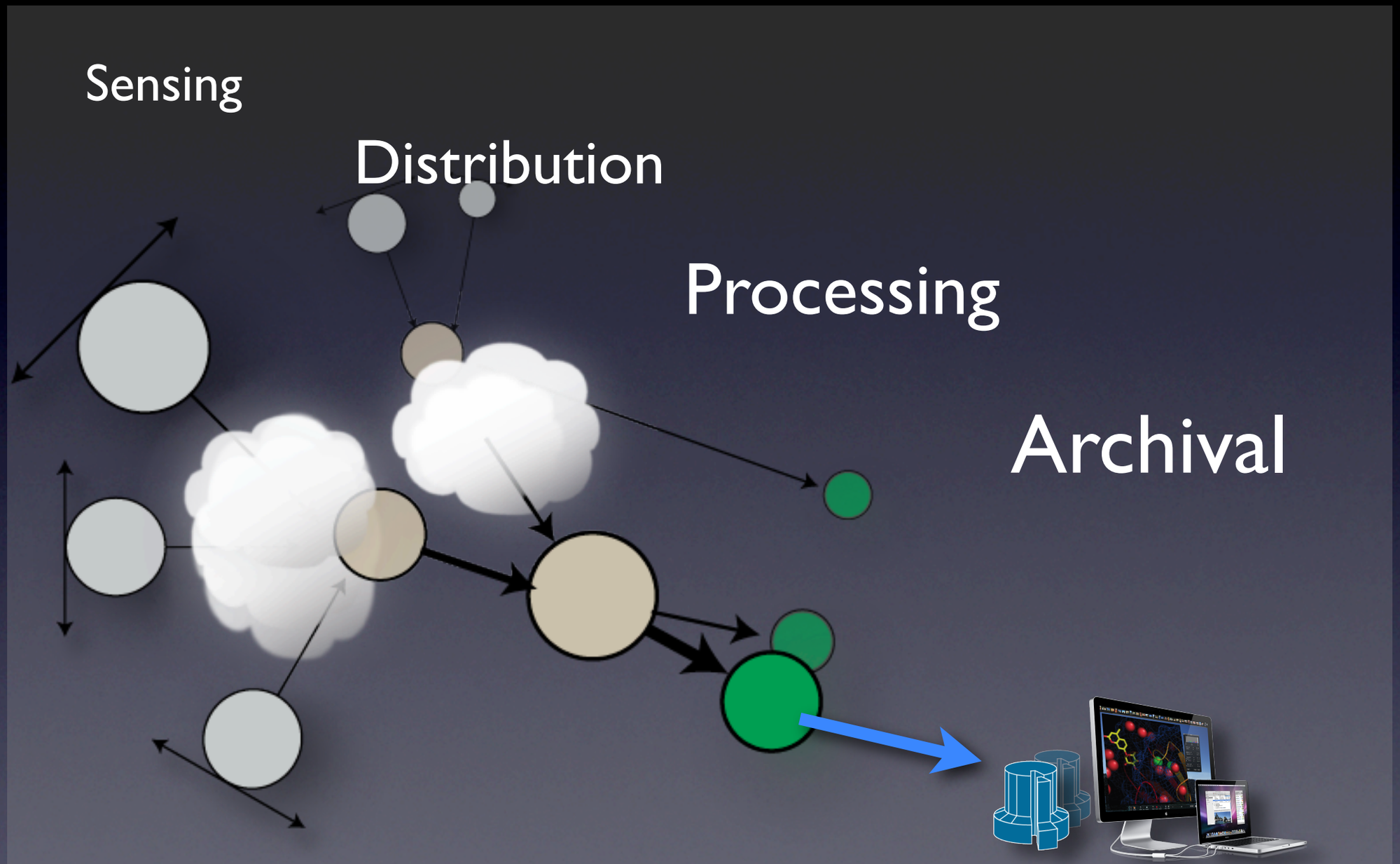
- Extensive Identifiability Methods
  - Multi-Layer Accountability
    - End Point Object Identifiers
      - Arbitrary encapsulation parsing and reporting
      - Need to support many, many formats
    - Service Oriented Object and State Identifiers
- Standards Based Traffic Metrics
  - Availability, Reachability, Connectivity (RFC 2678)
    - Bi-Directional
  - Service Oriented Usage/Performance Measurements
  - Advanced Performance Measurements
    - Loss, Jitter, Delay, Power, Distance, Performance
- Relational Algebraic Constraints


N_T
t_3
t_4
t_2
t_1
$(t_4 - t_1) - (t_3 - t_2) = N_T$

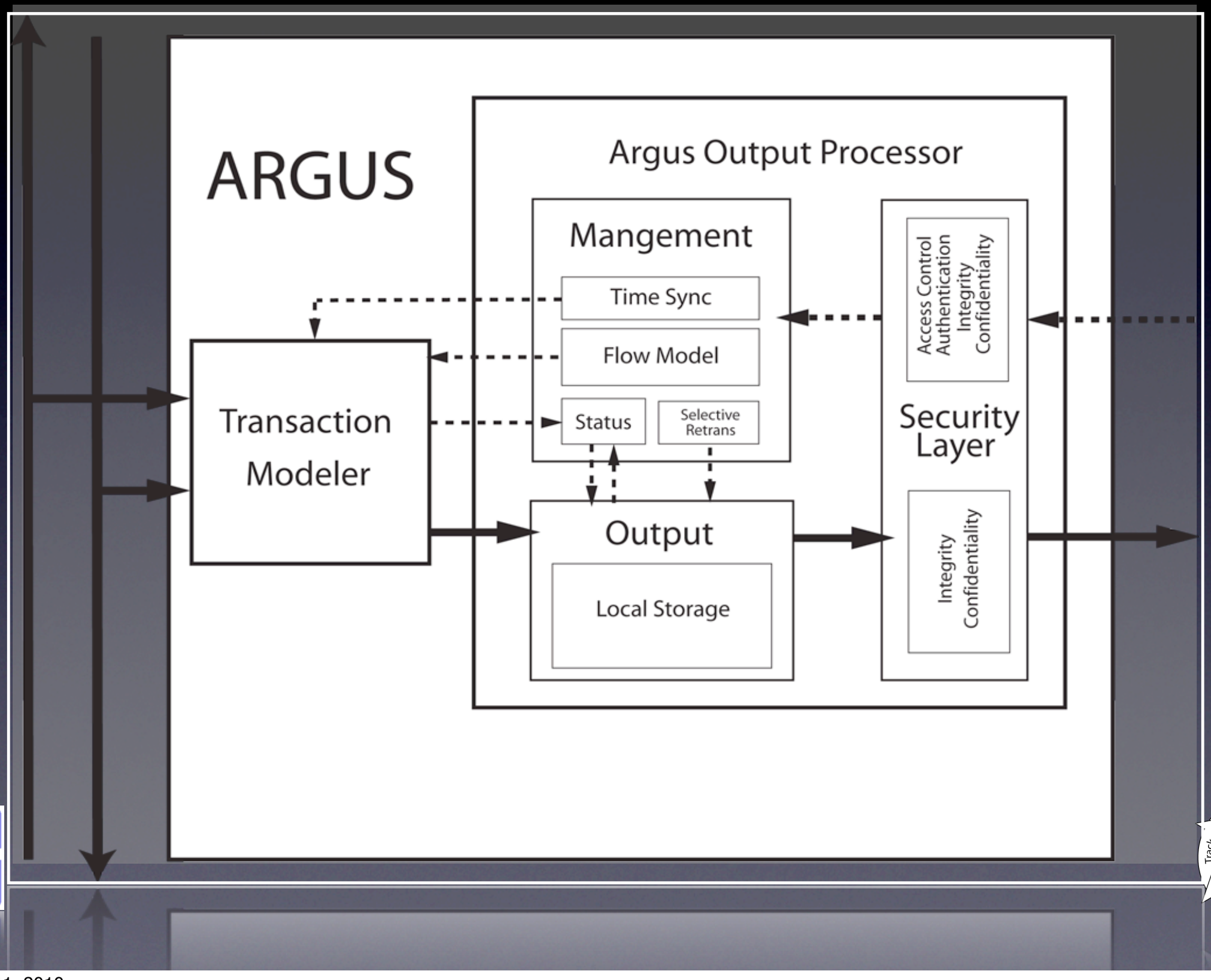
So what does all that mean?

- We need an Internet transaction concept
  - Network flow data record (Net CDR)
  - Network service oriented
    - Initiation, status, termination state indications for 5-tuple flows
    - All services - ARP, DHCP, DNS, OSPF, TCP....
- Data generation must be timely/deterministic/non-statistical/relevant/comprehensive
- Approach needs to perform and scale
  - Formal generation/consumption architectures
  - Collect/transport/process/correlate/join/select/search/store data
- Need to convey as much about the network traffic as possible
  - Support Layer 2/3/4/5+ semantics (including non-IP traffic)
- Needs to really solve some problems

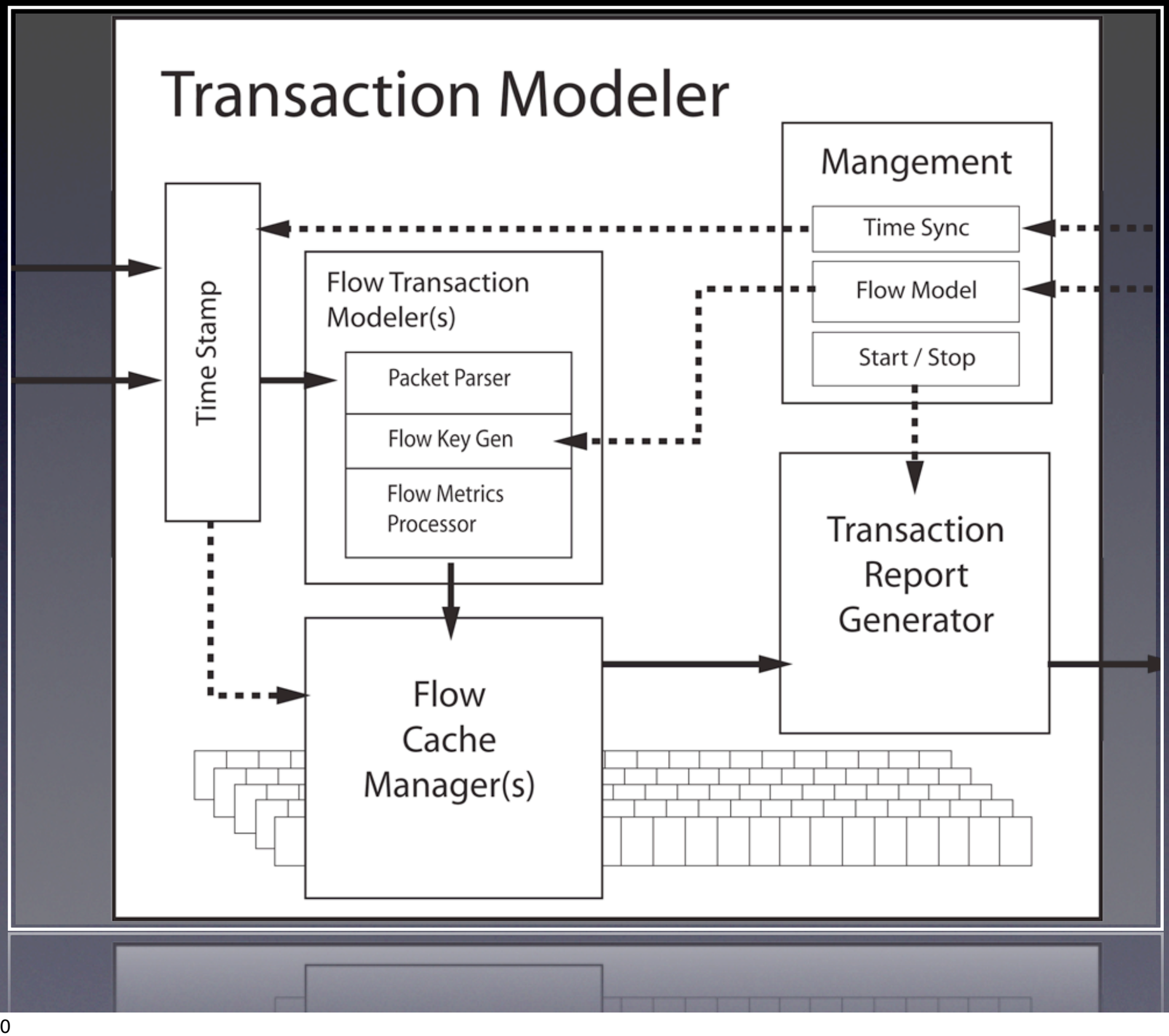
Argus System Design



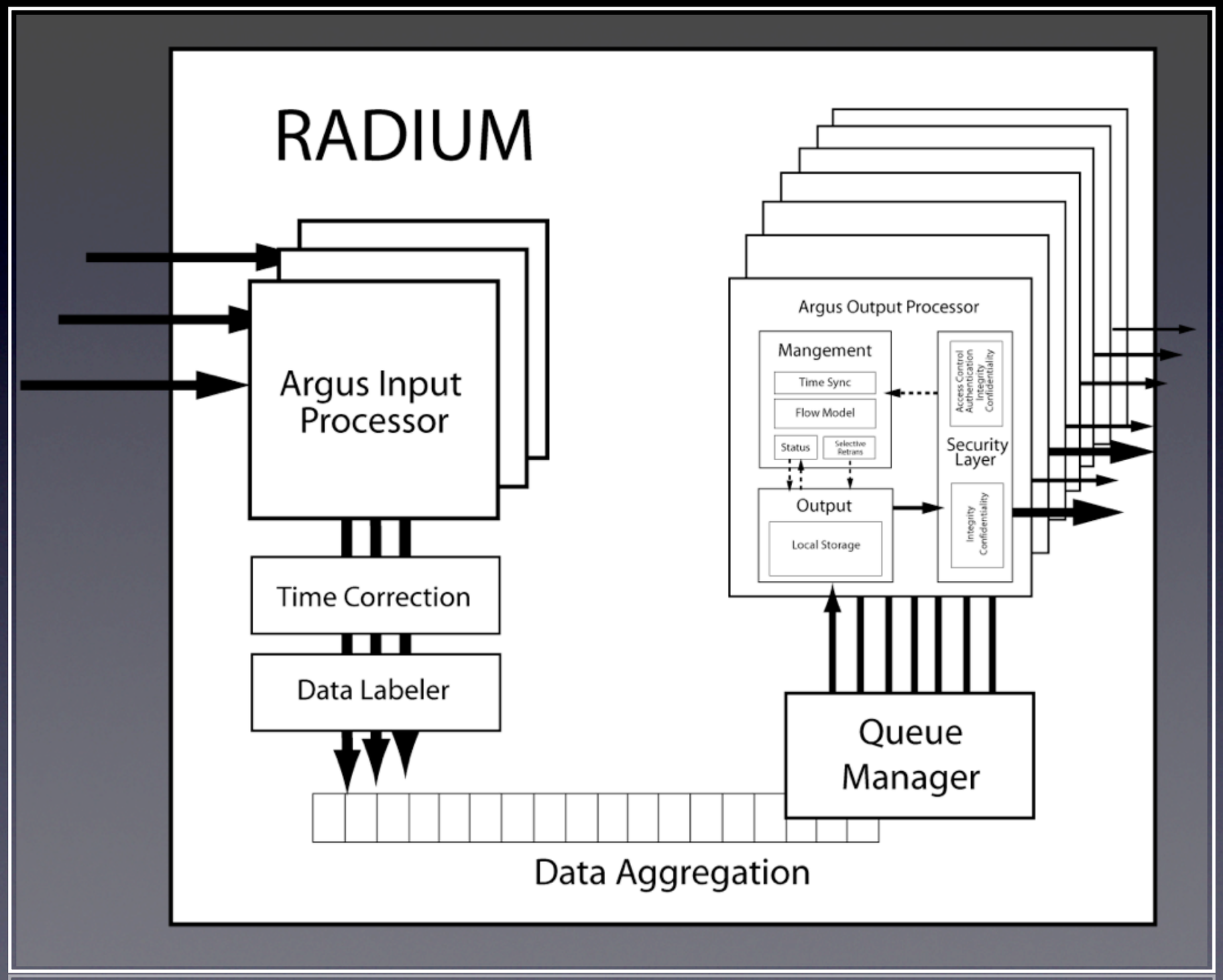
Argus Sensor Design



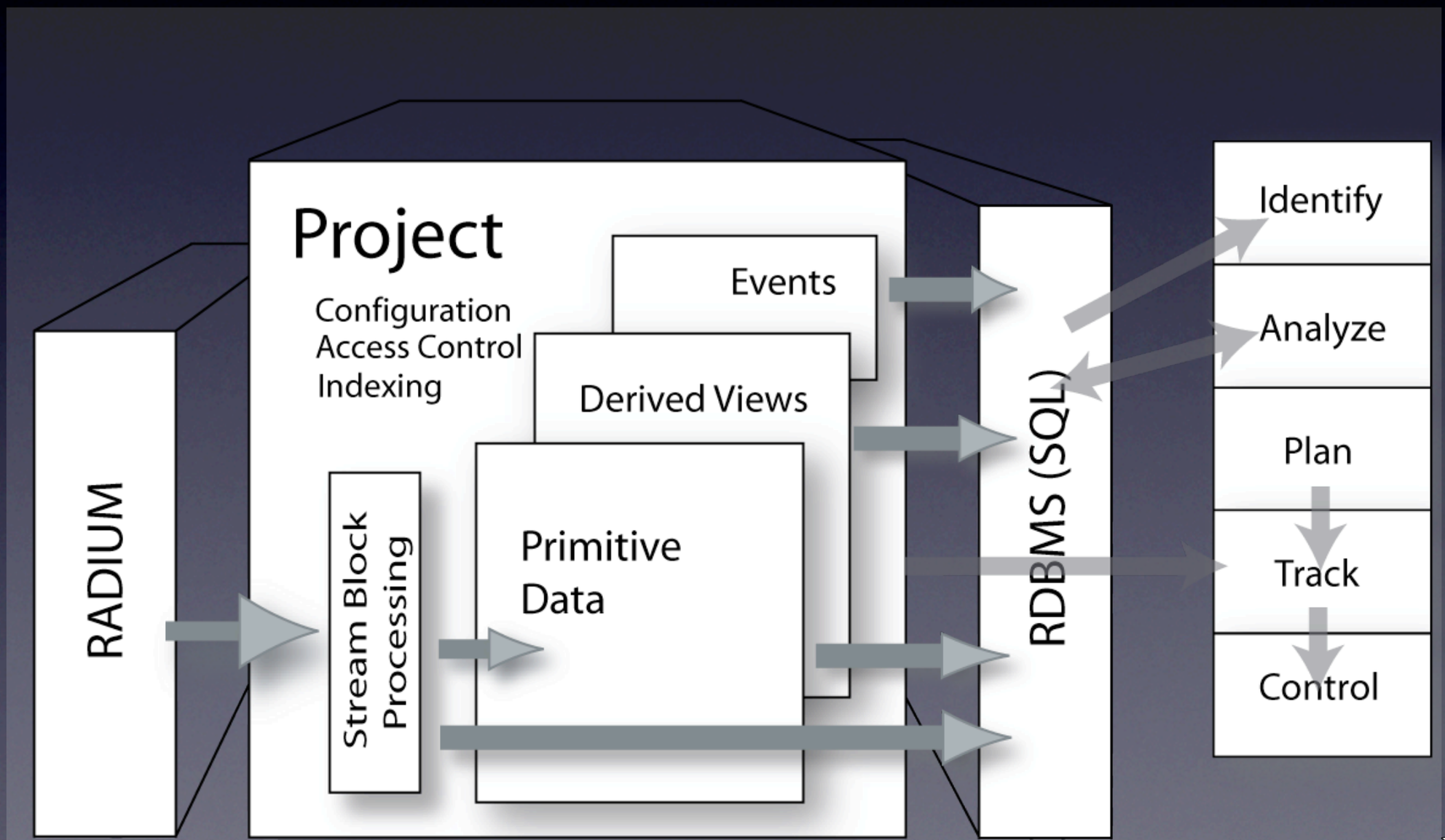
Argus Sensor Design



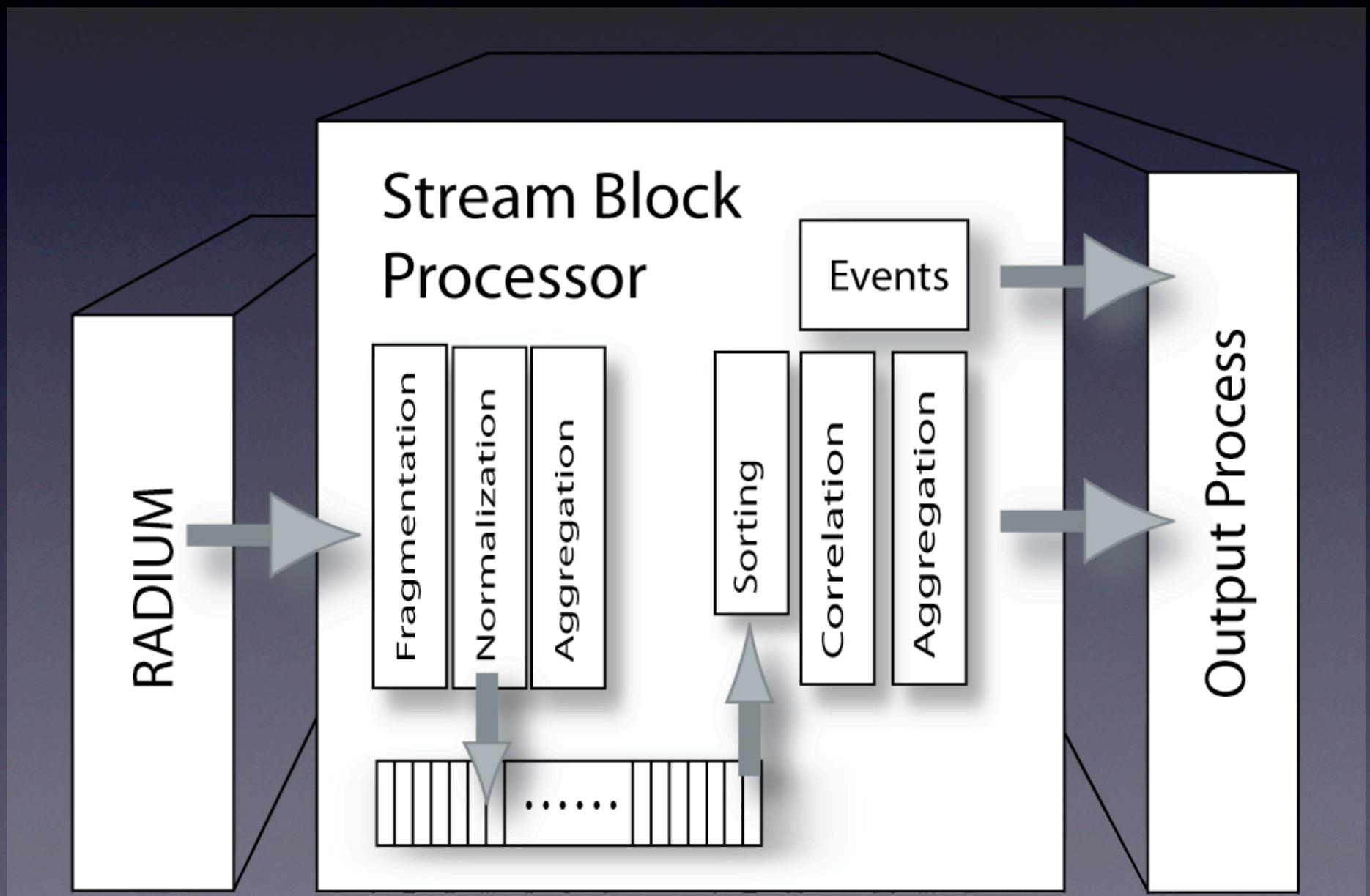
Radium Distribution



Argus Processing Design



Stream Block Processor Design

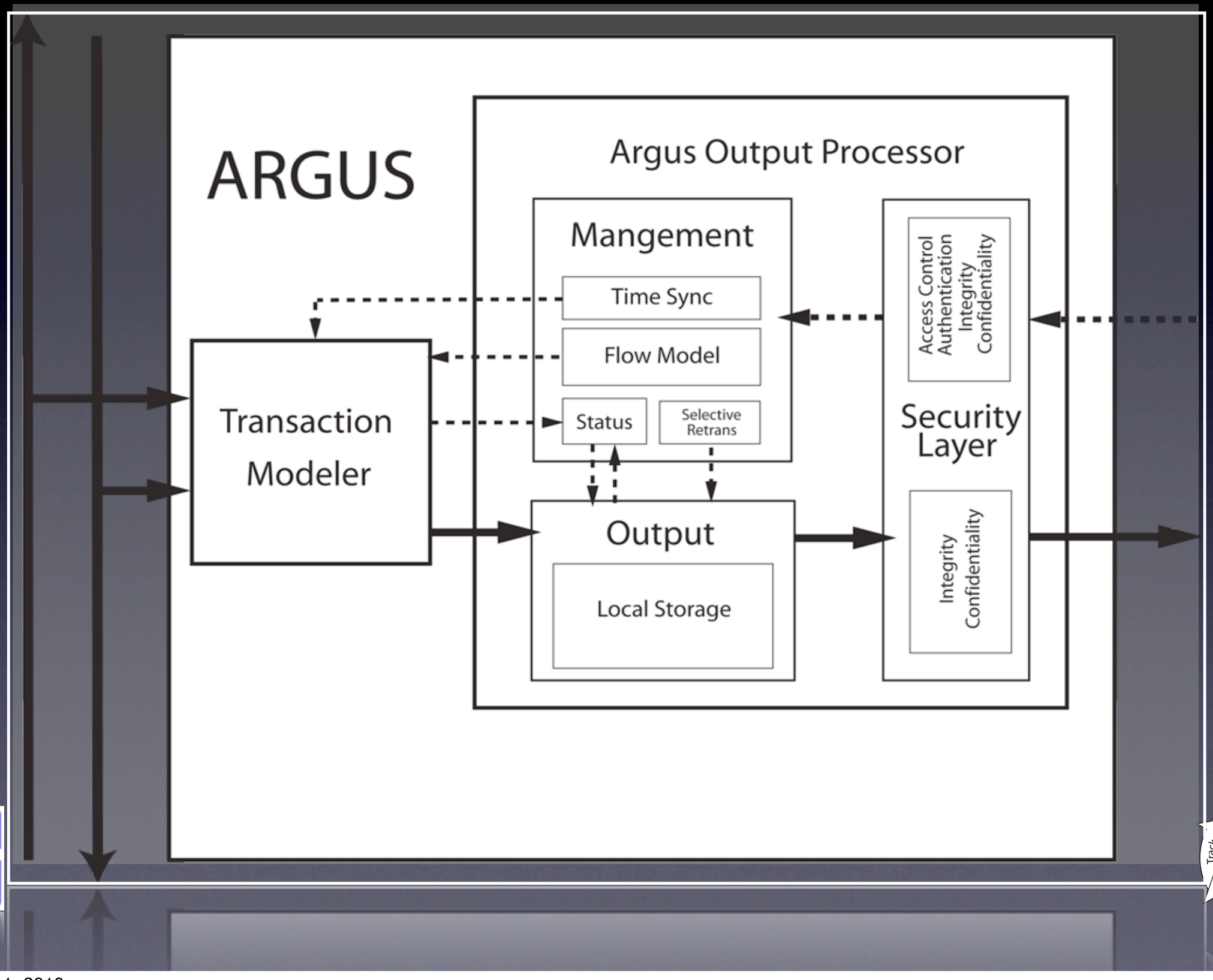


Data Generation

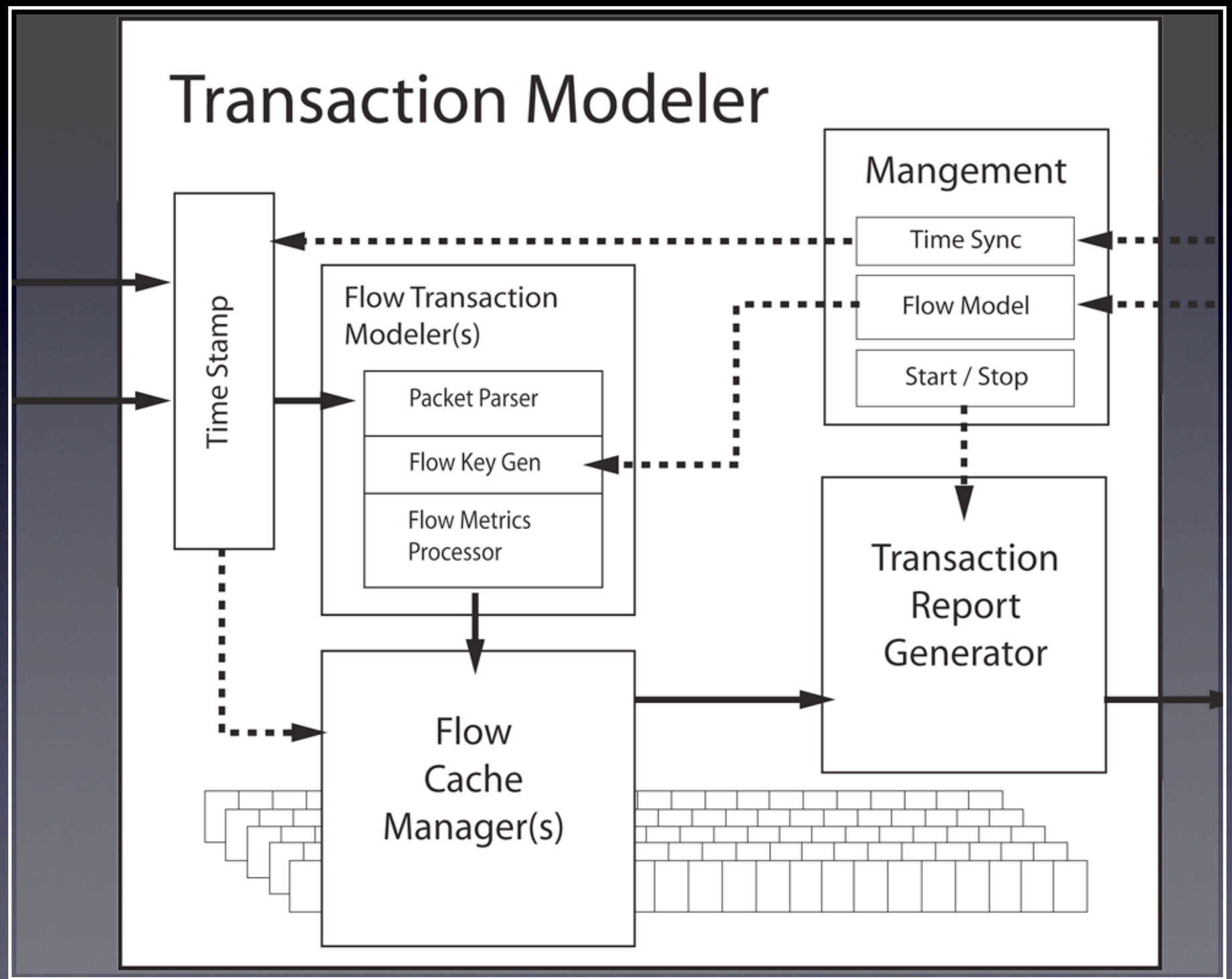
Data Generation

- Packets to Flows
- Getting Started with Argus
- Argus Deployment
- Configuration
- Running Argus

Packets to Flows



Argus Sensor Design



Packets to Flows

- Packet Timestamping
  - Methodology, Time Synchronization and resolution
- Packet Header Parser
  - Multiple flow tracking strategies determines parser
  - Supports OSI, IEEE, IP and Infiniband packet formats
  - Innermost Layer 3 target header (service layer)
    - Complex encapsulation stacking
      - L2 -> L3 -> L2 -> L3 -> L4 -> L3 -> L3
      - Support protocol discovery
  - Limited by packet snap size
    - Argus supports complex packet capture support
    - Privacy issues
    - Control plane vs data plane parsing

Packets to Flows

- Flow Key Generation
  - All packets are classified into a flow of some kind
  - Argus supports 14 fundamental flow types
    - Not protocols, flow types (P, P1-P2, Multicast/Unicast, etc....)
    - Bi-directional support for all flow types (when they exist)
    - Bi-direction flow keys for all supported encapsulations
- Flow Key is "key" to all flow tracking
- One packet one flow rule
  - Simplify flow machine call structure
  - Control plane is the exception
    - ICMP packet accounted for in ICMP flow
    - ICMP state mapped to flow identified in contents

Packets to Flows

- Flow Metrics Processor
  - Metric and attribute generation
    - Some metrics can be derived from packet itself
      - Packet size, application demand, reachability
    - Others require state
      - connectivity, availability, RTT, rate, loss, jitter
  - Flow attribute (re)assignments
    - Flow state machine tracking
    - Dynamic attribute tracking
- Flow Cache Manager
  - Controls reporting of flow status
  - Controls dynamic flow redefinitions/reassignments

Getting Started

- http://qosient.com/argus
- ‘Using Argus’ and ‘Getting Argus’ Links
- Argus documentation
  - Man pages provided in distribution
  - HOW-TO and FAQ on the web site.
  - Argus developers mailing list
    - argus-info@lists.andrew.cmu.edu.
    - Most questions are answered here
    - Email carter@qosient.com

Getting Argus

- http://qosient.com/argus/downloads.htm
- Many linux distributions have a port
- Current stable version is argus-3.0.2
- Provided as tarball source package

- Depends on:
  - libpcap - http://tcpdump.org/release
  - bison - http://www.gnu.org/software/bison/bison.html
  - Client ragraph() requires perl & rrd_tool

Making Argus

- Simple installation
  - ./configure; make
- Complex environments
  - Read ./README and ./INSTALL
  - Cygwin/OpenWRT
- Support standard autoconf options
  - ./configure --help
  - Common variations
    - prefix=/your/destination/directory
    - SASL Support
    - Native compiler options

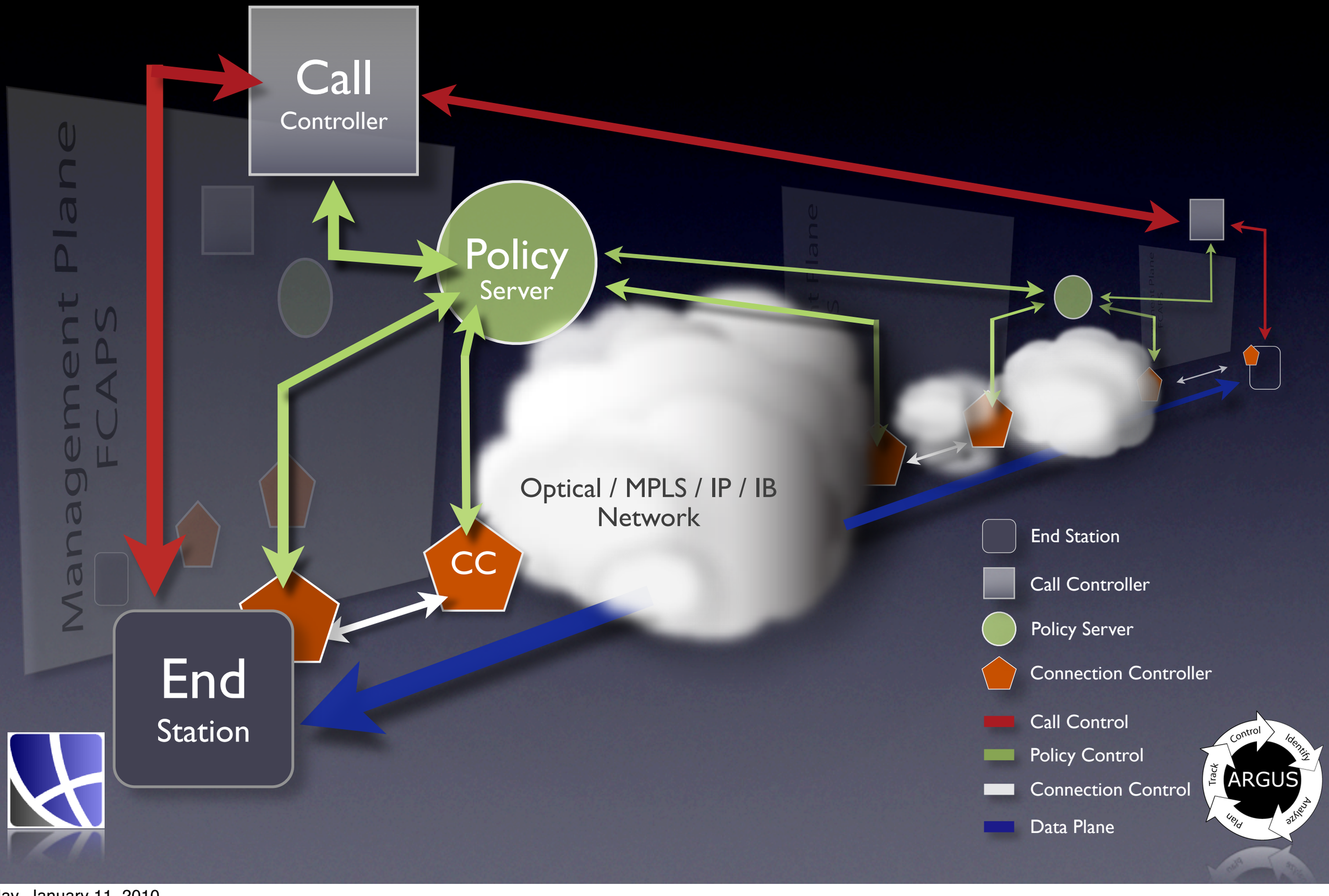
Installing Argus

- Simple installation
  - make install
- ./INSTALL describes some complex examples
- /etc/argus.conf
- System startup configuration
  - Linux chkconfig.1 support
  - MacOS X /Library/StartupItems support
- RPM support - ./lib/argus.spec

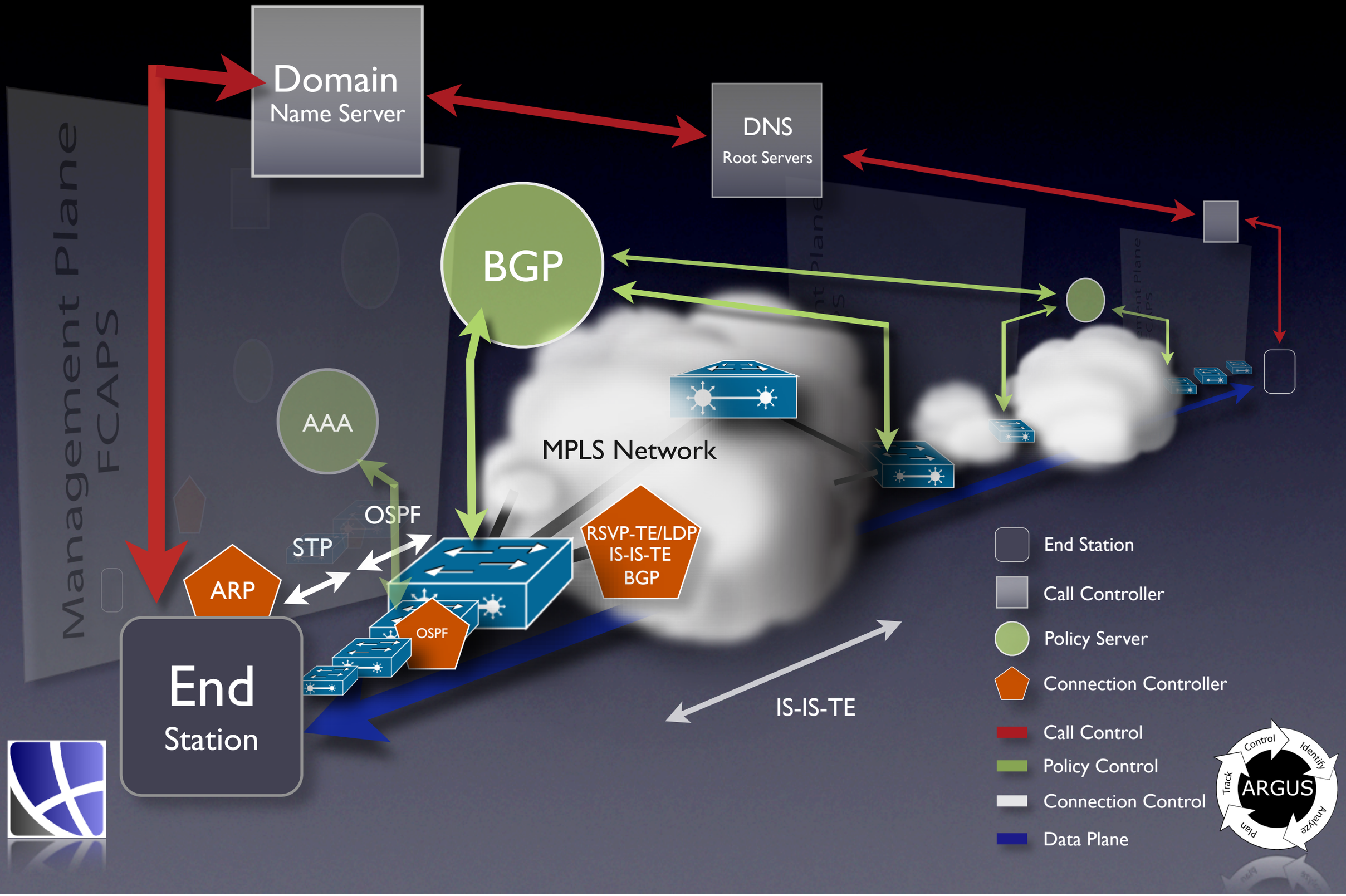
Deployment

- Monitoring Strategies
  - Enterprise Border Monitoring
  - Subnet Monitoring
  - End System Monitoring
  - Complex/Comprehensive Monitoring

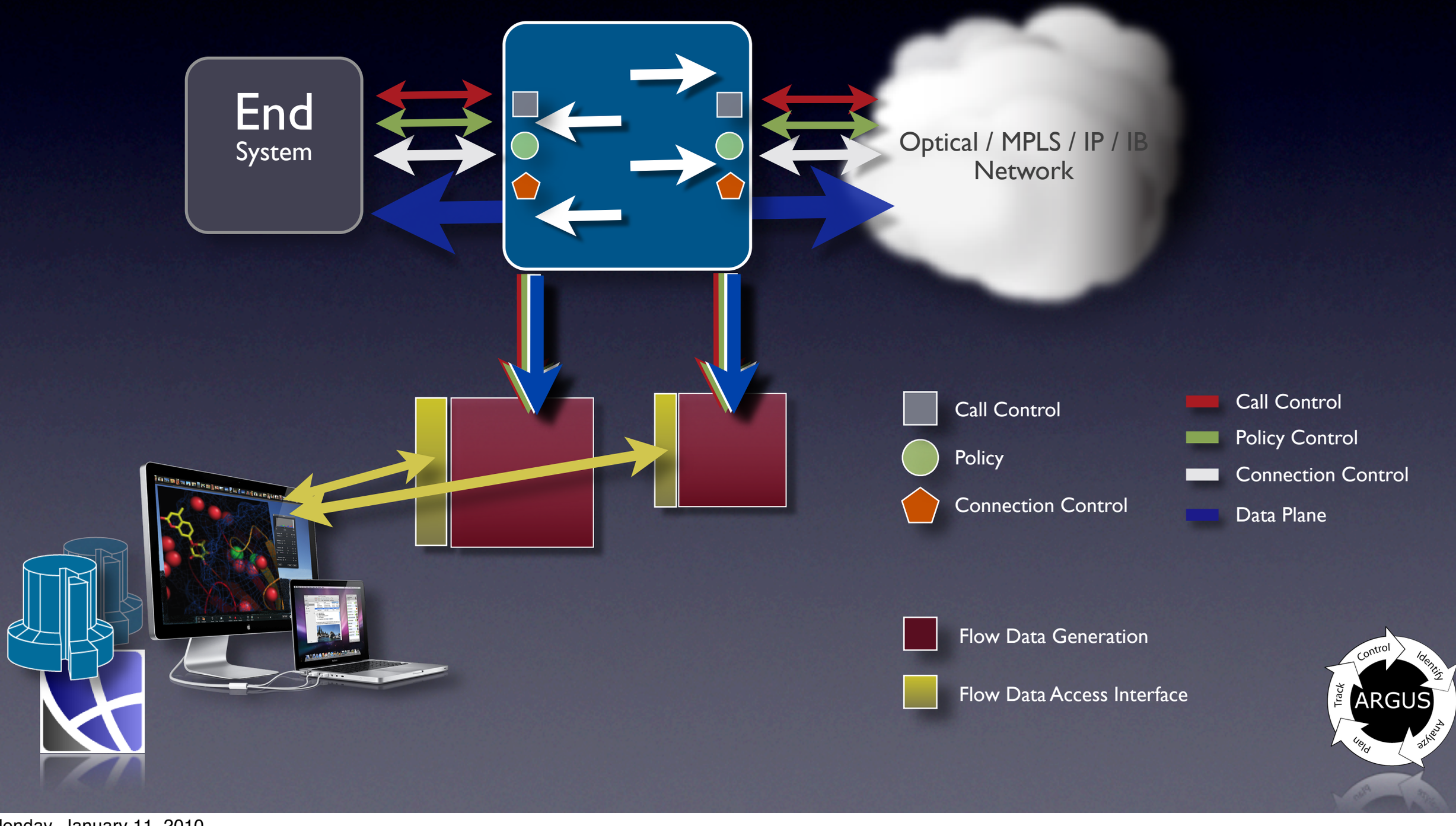
Network Reference Model



Internet Networking Model



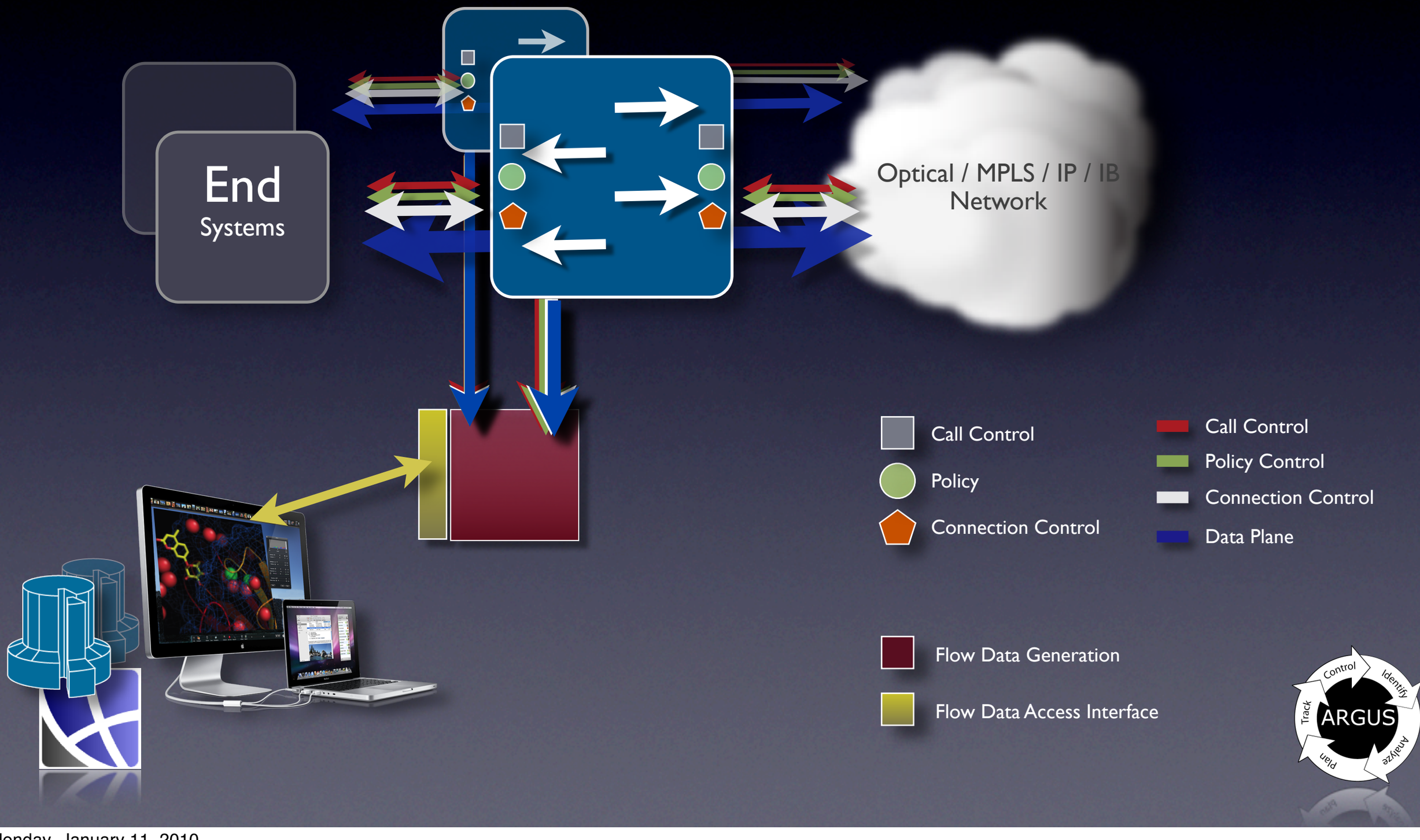
Enterprise Border Monitoring
Internal/External Strategies



Enterprise Border Monitoring

Asymmetric Routing Strategies

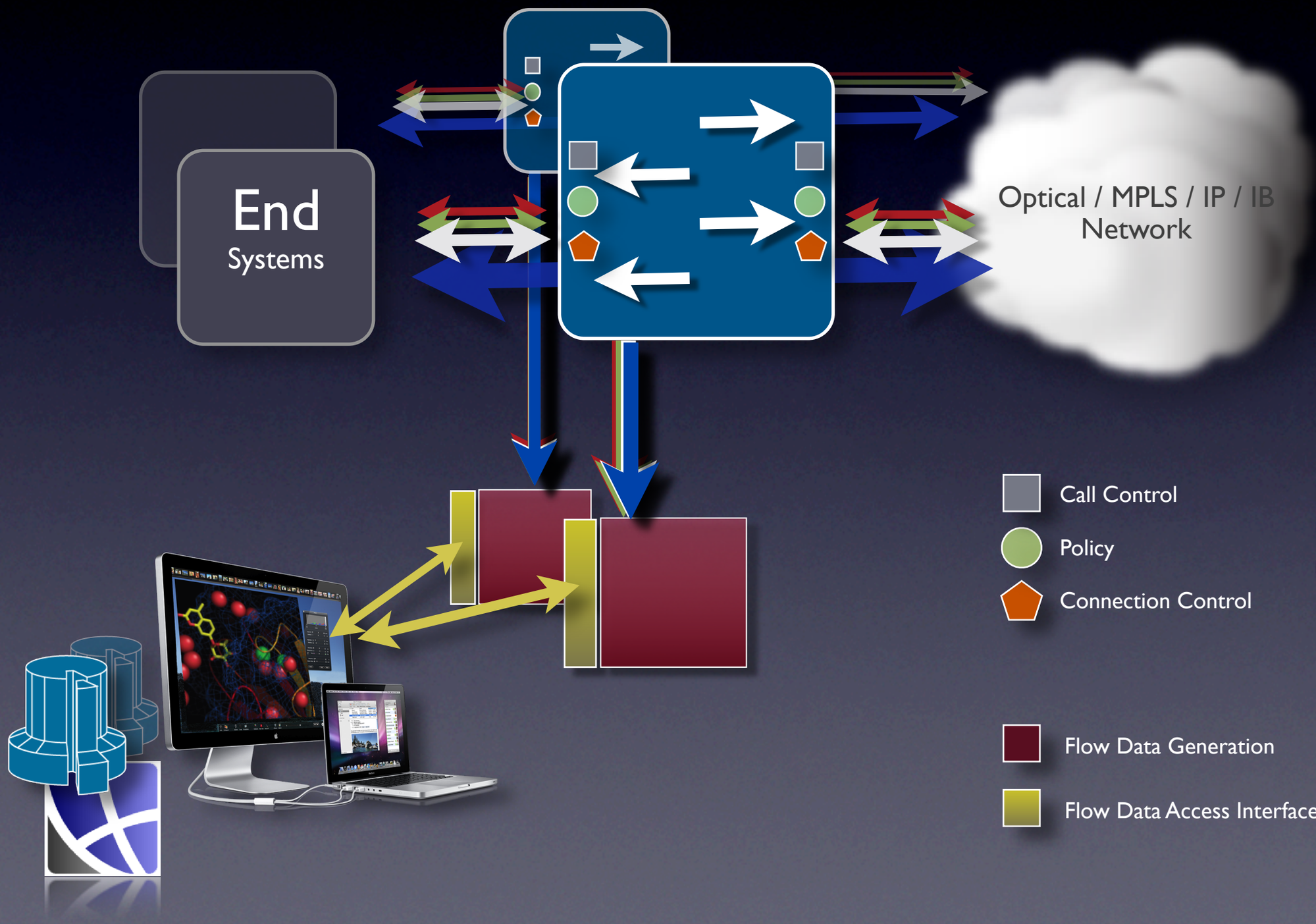
Single Probe



Enterprise Border Monitoring

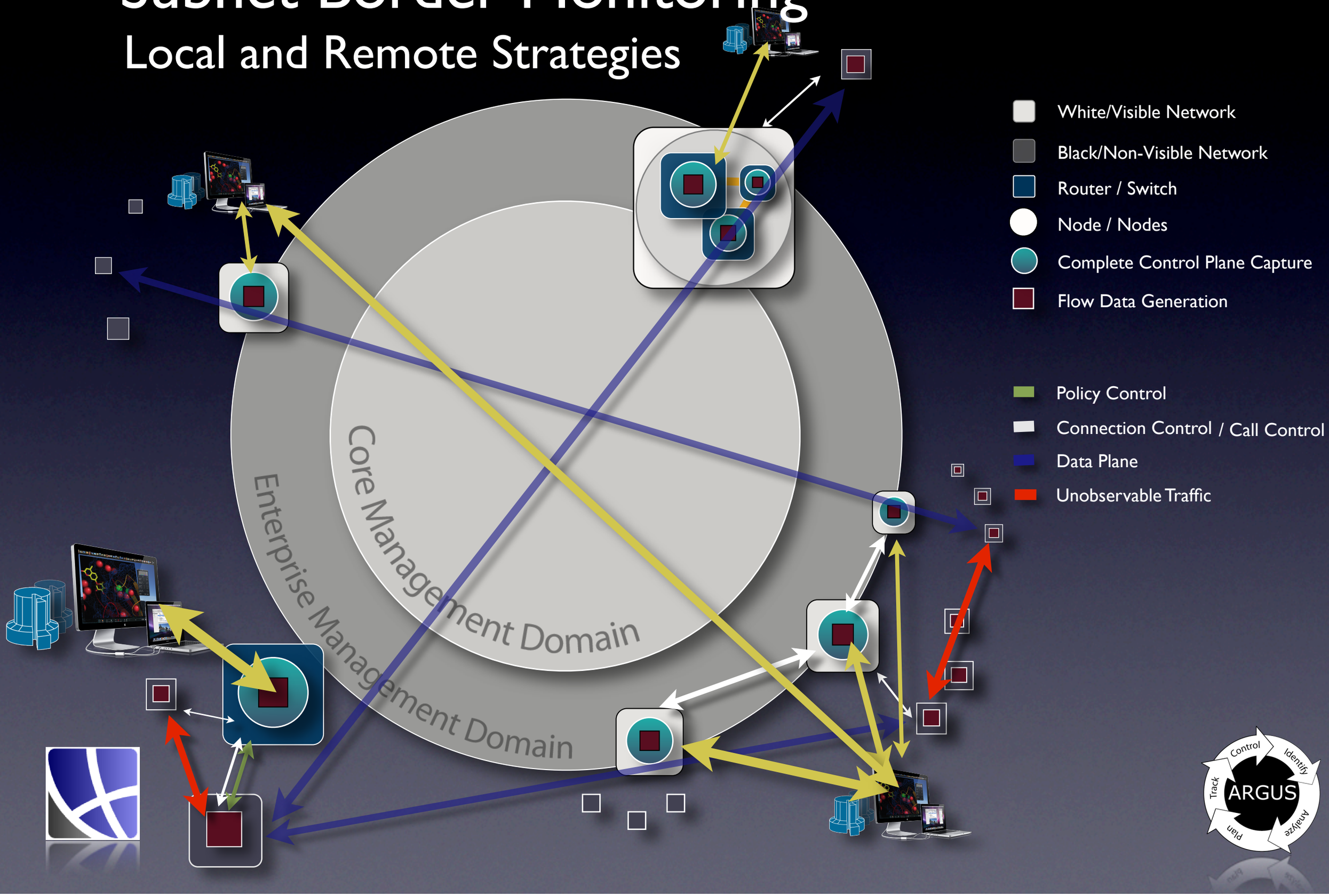
Asymmetric Routing Strategies

Multiple Probes

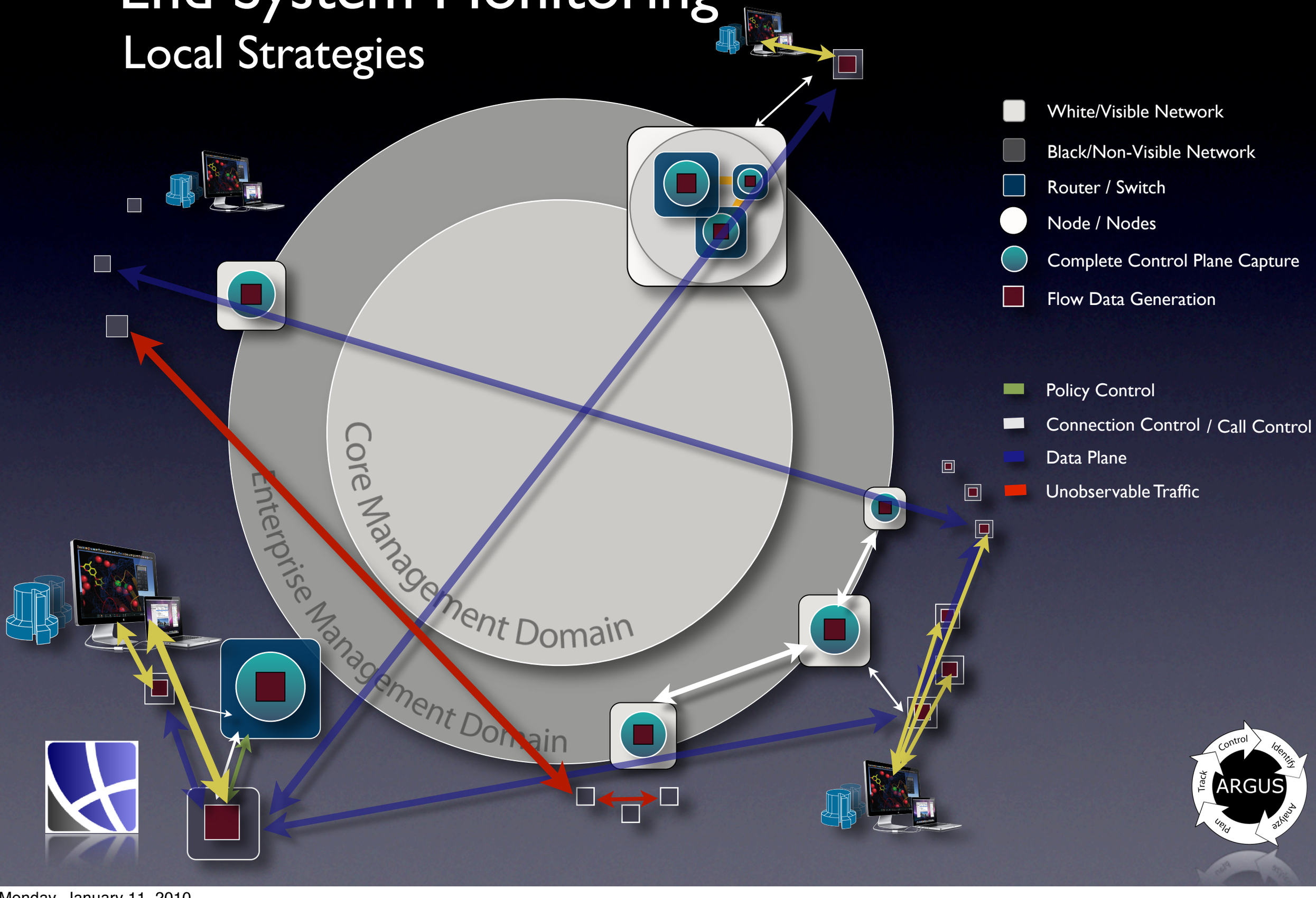


Subnet Border Monitoring

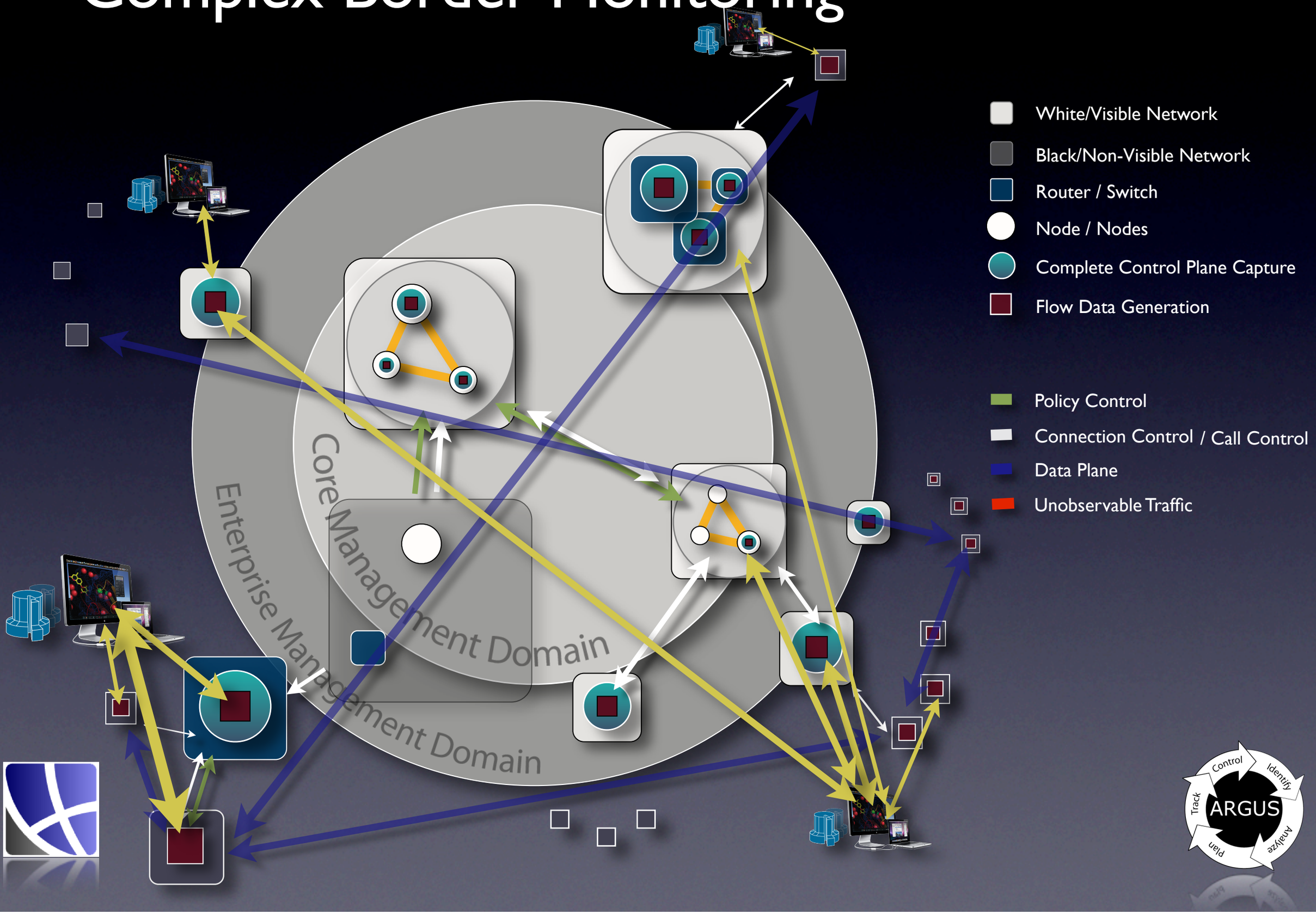
Local and Remote Strategies



End System Monitoring
Local Strategies



Complex Border Monitoring



Complex Monitoring

- Critical elements
  - Time synchronization
  - Comparable flow key models
  - Observation Domain ID Allocations
    - argus sourceID
- Best Common Practices
  - Time synchronization
  - Comparable flow key models
  - Observation Domain ID Allocations
    - argus sourceID

Running Argus

- Packet File Processing
- Multiple Interface Strategies
- Security
- Local Storage

Packet File Processing

- Supported formats
  - Historically supported all of them
  - Now, there are many packet capture conversion tools
  - Currently supports libpcap and ERF
- argus -r packet.file -w packet.file.argus
  - Default options are a good start
  - There are hundreds of options
  - Should utilize argus.conf when possible
- Best Common Practices
  - Don’t throw away the original packet capture file
  - Keep your data together; packets and flow data
  - Structure your data around time

Packet Streams

- Boils down to Access to Packets!!
- Host/End System Deployment
  - Integrated NIC Interfaces
  - Special purpose NIC/FPGA capture
- Network Deployment
  - Integrated Deployment
    - Linux based routers / OpenWRT
    - Experimental hardware
  - Packet Copy Techniques
    - Multiport Repeaters
    - Span Ports
    - UTP Network Taps (including Regen Taps)
    - Optical Network Taps
    - Packet Demultiplexors

Configuration

- argus.conf
  - Running Environment
  - Monitor Characteristics
  - Flow Data Metrics
  - Security Mechanisms

Running Environment Configuration

- Argus Daemon
- Argus Access Port
- Argus Bind IP Address
- Argus Interface
- Argus Go Promiscuous
- Argus Collector
- Argus Chroot Directory
- Argus Set User ID / Group ID
- Argus Output File
- Argus Set PID & PID Path
- Argus Packet Capture File
- Argus Environment Variables

Monitor Characteristics Configuration

- Argus Monitor ID
- Argus Flow Status Interval
- Argus MAR Status Interval
- Argus Debug Level
- Argus Filter Optimizer

Flow Data Metrics Configuration

- Argus Flow Type
- Argus Generate Response Time
- Argus Generate Packet Size
- Argus Generate Jitter Data
- Argus Generate MAC Data
- Argus Generate Application Byte Metrics
- Argus Generate TCP Performance Metrics
- Argus Generate Bi-Directional Time Stamps
- Argus Capture Data Length

Security Mechanisms Configuration

- Argus support the use of SASL to provide strong authentication and confidentiality protection.
- The policy that argus uses is controlled through the use of a minimum and maximum allowable protection strength. Very SASL specific.
  - RA_MIN_SSF
    - This is the minimum security strength factor for the connection. An SSF of 0 allows for no protection. An SSF of 1 will supply integrity protection without privacy.
  - RA_MAX_SSF
    - The MAX_SSF is normally used to specify the strength of encryption. 56, as an example, specifies 56-bit DES. This value should not be less than the MIN_SSF.

Common Problems

- Can't start argus
  - Permissions (interface/filesystem)
    - run as root
- Can't connect to running Argus
  - Tcp_wrappers getting in the way
    - check syslog()
- Argus closes connection after a while
  - Client doesn't read data fast enough
    - improve resources between argus and client

Client Programs

Disclaimers

- Argus clients do indeed have bugs
  - Argus and its clients are prototypes
  - It really is just me doing this work
- If it doesn't do what its suppose to do
  - More than likely its a bug.
  - Sometimes its because we haven't finished it.
  - Sometimes the documentation is wrong.
- Please report any problems!!!
  - Send email to the list or to me
  - Sometimes, it gets fixed that day.

Argus Client Programs

- Basic Operations
- Aggregation
- Data Splitting
- Graphing/Visualization
- User Data Processing
- Semantic Enhancement
- Anonymization

Basic Operations

- ra - provides data file creation, printing, filtering, stripping
- ratop - curses() based argus data GUI
- rasort - provides in memory sorting
- racount - simple record counting
- rahosts, raports, routers - Perl script examples of simple data processing.

Argus Client Program Configuration

- Extensive command line options
- .rarc configuration
- Runtime Environment
- Data Access
- Printing Support
- Security

Argus Client Program Configuration

- Runtime Environment
  - RA_SET_PID
  - RA_PID_PATH
  - RA_OUTPUT_FILE
  - RA_TIME_RANGE
  - RA_TZ
  - RA_DEBUG_LEVEL
  - RA_TIMEOUT_INTERVAL
  - RA_UPDATE_INTERVAL
  - RA_DELEGATED_IP
  - RA_RELIABLE_CONNECT

Argus Client Program Configuration

- Data Access
  - RA_ARGUS_SERVER
  - RA_SOURCE_PORT
  - RA_CISCONETFLOW_PORT
  - RA_TIME_RANGE
  - RA_RUN_TIME
  - RA_FILTER

Argus Client Program Configuration

- Printing Support
  - RA_PRINT_MAN_RECORDS
  - RA_PRINT_LABELS
  - RA_FIELD_SPECIFIER
  - RA_FIELD_DELIMITER
  - RA_FIELD_WIDTH
  - RA_PRINT_NAMES
  - RA_PRINT_RESPONSE_DATA
  - RA_PRINT_UNIX_TIME
  - RA_TIME_FORMAT
  - RA_USEC_PRECISION
  - RA_USERDATA_ENCODE

Argus Client Program Configuration

- Security
  - RA_USER_AUTH
    - This is the user name and, depending on the MECH, a group name, for the SASL account to be used for authentication.
  - RA_AUTH_PASS
    - This is the password for the SASL account to be used for authentication. This plain-text entry does pose some issues, so be sure and protect your .rarc file if this method is used.
  - RA_MIN_SSF
    - This is the minimum security strength factor for the connection. An SSF of 0 allows for no protection. An SSF of 1 will supply integrity protection without privacy.
  - RA_MAX_SSF
    - The MAX_SSF is normally used to specify the strength of encryption. 56, as an example, specifies 56-bit DES. This value should not be less than the MIN_SSF.

Aggregation

Network data aggregation is a MASSIVE topic. It drives most of the data analysis and report generation and is the heart of all the interesting programs.

- racluster, rabins, ratop
  - Each field has its own aggregation methods
  - Semantic preservation

ra()

ra is the basis of all ra* programs, in that it is the simplest of the client programs written against the client library. It is simply, "read the data source and print each record, one record at a time".

- All ra* programs do what ra() does
- Use ra to inspect individual records
- ra.1 is your primary documenation
- If you want to develop argus clients
  - ra.c should be your first example

ratop()

ratop is the top() equivalent for argus data. It is becoming the argus data editor as time goes on.

ratop is also an argus data aggregator.

- ratop is the 1° argus client
- it incorporates all basic functions into a single program.
- it is the best program example for near-realtime situational awareness

Aggregation

- Process of merging records
  - Isoflow Aggregation
    - Time extension / data reduction schemes
      - Consolidate status reports to single record
  - Anisoflow aggregation
    - Reduced flow key
      - Select/Join style data consolidation
    - Expanded flow key
      - Addition of flow attributes to key (i.e. DSBytes)
      - Can be used to generate new metrics
- Each field has its own aggregation rules
  - Data accumulation, reduction, replacement, f(x)
  - Must be statistically or relationally sound

Aggregation

- racluster
  - Default options create isoflow aggregation
  - simple command-line options
    - “-m field [field ...]” option defines anisoflow
      - aggregation scope is duration of data stream
      - storage problems as each flow is cached
- racluster.conf
  - compound aggregation rule structures
  - control of cache holding times and idle times
  - option for semantic preservation

Aggregation

- Configuration

```
RACLUSTER_MODEL_NAME=Test Configuration
RACLUSTER_PRESERVE_FIELDS=yes
RACLUSTER_REPORT_AGGREGATION=yes
RACLUSTER_AUTO_CORRECTION=yes

filter="tcp or udp"  model="saddr daddr proto dport"       status=120 idle=3600 cont
label ="Class-Video" model="srcid saddr daddr proto dport" status=5   idle=10
filter="tcp or udp"  model="saddr daddr proto dport"       status=30  idle=120
filter="icmp"        model="saddr daddr proto dport sport" status=60  idle=30
filter="arp"         model="saddr daddr proto dport sport" status=120 idle=60
filter=""            model="saddr daddr proto"             status=300 idle=3600
```

Data Splitting

Argus can generate a lot of data. Tools that help in data disposition are very, very helpful. Here we are providing basic file processing tools, like the unix command split().

- rasplit
  - First designed to support sorting
  - Split based on time, size, count, and contents
  - Time based splitting basis for 1° archive methods
- rastream
  - rasplit + file processing on close

Graphing/Visualization

Network data graphing is a powerful communication tool for report generation, etc..., but it is also the best way to verify and validate the correctness of data processing.

- ragraph
- Perl script processing rabins() output
  - Generating rrd and running rrd_graph
    - Single object / multi metric graphing
    - Strict time-series representations
    - 1 second minimum resolution
  - Variations use GNU Plot, Mathmatica, MatLab
- CSV file generation
  - Data support for many 3rd party systems
    - Numbers, Excel, AfterGlow, PicViz, etc...
    - Issues with date formats - ./support/Config/excel.rc

User Data Processing

- All ra* programs can grep() user data.
- radump
  - Port of tcpdump decoding logic to argus user data buffers.
  - Use this as a prototype for content sensitive analytics.
- rauserdata/raservices
  - Upper protocol discovery, classification, identification, verification
  - Will provide guess for unknown protocols.
  - Experimental, but works very well.

Semantic Enhancement

- ralabel
  - Geospatial Information Merging
    - Country Codes, City, Area Code, Postal Codes, Physical Address, Lat/Lon
  - Netspatial Information Fusion
    - Origin AS Number
    - Domain Name
    - Path Information
  - Flow Classification
  - Tagging
  - ralabel.conf

Anonymization

Network data anonymity is a big topic when considering sharing for research and collaboration.

The strategies used by argus are intended to preserve the information needed to convey the value of the data, and change or throw everything else away.

- ranonymize
  - User Data Capture
  - Time
  - Network Object Identifiers
    - Network Addresses
    - Service Access Point Identifiers (ports)
  - Sequence Numbers

Data Collection

Data Collection

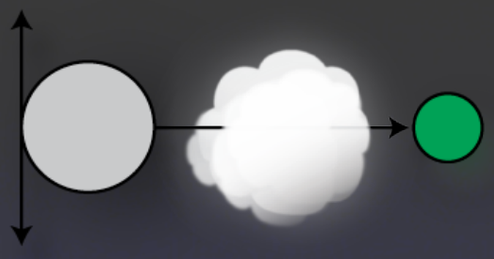
All ra* programs can read data from any Argus data source, files, stream, encrypted, and/or compressed, and can write current file structure.

Making an argus data repository needs just a little bit more.

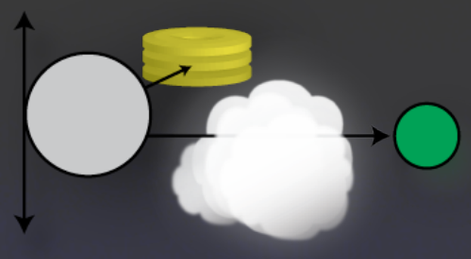
- File Distribution
- Radium Distribution
- Argus Repository Establishment
  - cron
  - rasplit/rastream
  - rasqlinsert/rasql

Data Collection

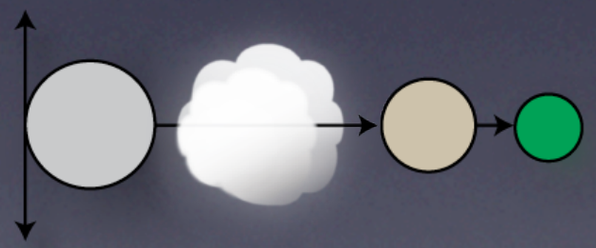
Argus reading from packet files or network and writing directly to disk



Argus reading from the network and writing directly to network based client



Argus reading from the network and writing directly to disk and network based client



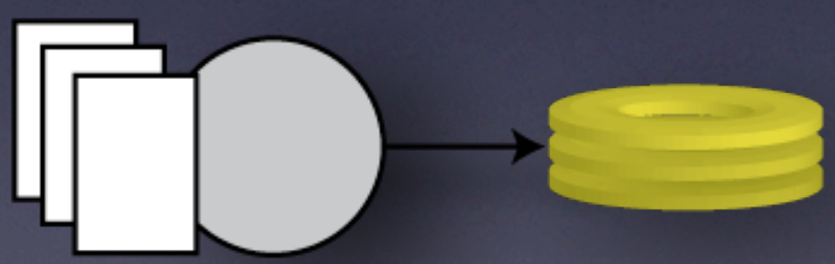
Argus reading from the network and writing directly to a network Radium, writing to a client

Argus writing to local Radium which is writing directly to disk and to network based clients

Many Argi writing directly to a Radium based distribution network, which is providing data to a set of clients.

Data Collection

- Local Generation and Storage
  - Basis for argus-2.0 argusarchive.sh
  - Direct argus support for renaming files
  - Normally cron mediated
  - Issues with time and record spans
- System designer has most control !!!



Argus reading from packet files or network and writing directly to disk

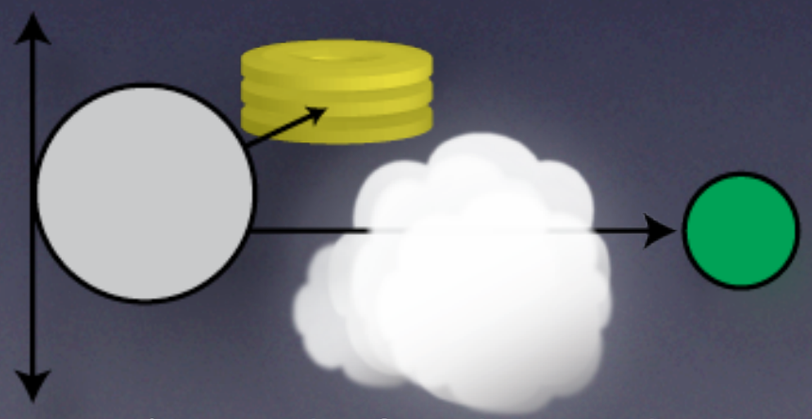
Data Collection

- Local Generation Remote Collection
  - Most high performance systems use this strategy
    - Provides explicit scalability and performance capabilities
    - Relieves argus from physical device blocking
    - Network interfaces generally faster than local storage devices
- Introduces network transport issues
  - Reliability, connection vs. connection-less, unicast vs multicast, congestion avoidance, access control and confidentiality

Argus reading from the network and writing directly to network based client

Data Collection

- Local Storage and Remote Collection
  - Used when data reliability is most critical
    - Local storage provides explicit data recovery
    - File collection provides additional distribution flexibility
    - Scheduled transport
- Reduces ultimate sensore performance
  - Argus itself is doing a lot of work
  - Packet processing is really the ultimate limit



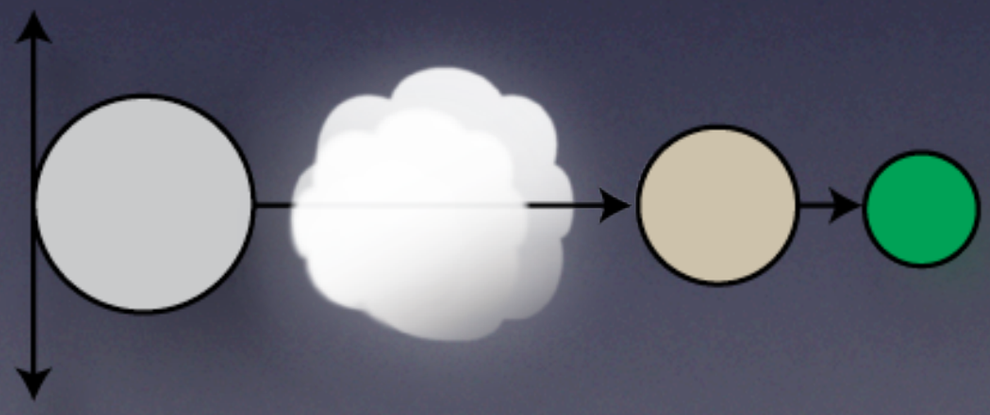
Argus reading from the network and writing directly to disk and network based client

Data Collection

- Radium
  - Primary argus data distribution technology
  - Radium is a ra* program with an argus output processor.
    - Read from many sources
    - Write to many clients
    - Serve up argus data files
    - Process/transform data
    - Configuration is combo of argus() and ra()
- Supports very complex data flow machine architectures.

Data Collection

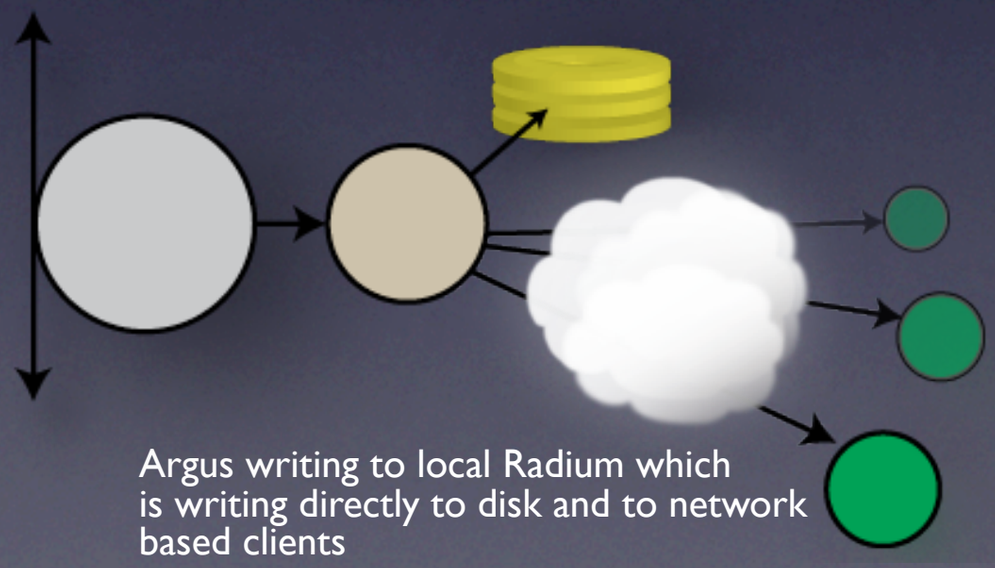
- Local Generation Remote Distribution
  - Most prevalent strategy used in argus-3.0
    - Provides explicit scalability and performance capabilities
    - Provides most stable collection architecture from client perspective
    - Single point of attachment for complete enterprise
  - Least reliable of 'advanced' strategies
    - Radium failure interrupts continuous stream collection, with no opportunity for recovery



Argus reading from the network and writing directly to a network Radium, writing to a client

Data Collection

- Local Distribution and Storage
  - Best methodology
    - Provides explicit scalability and performance capabilities
    - Provides most reliable collection architecture
    - Multiple points of attachment, multiple points of control
- Most expensive strategy at data generation
  - Radium deals with device and remote client requests for data which does come with a processor and memory cost



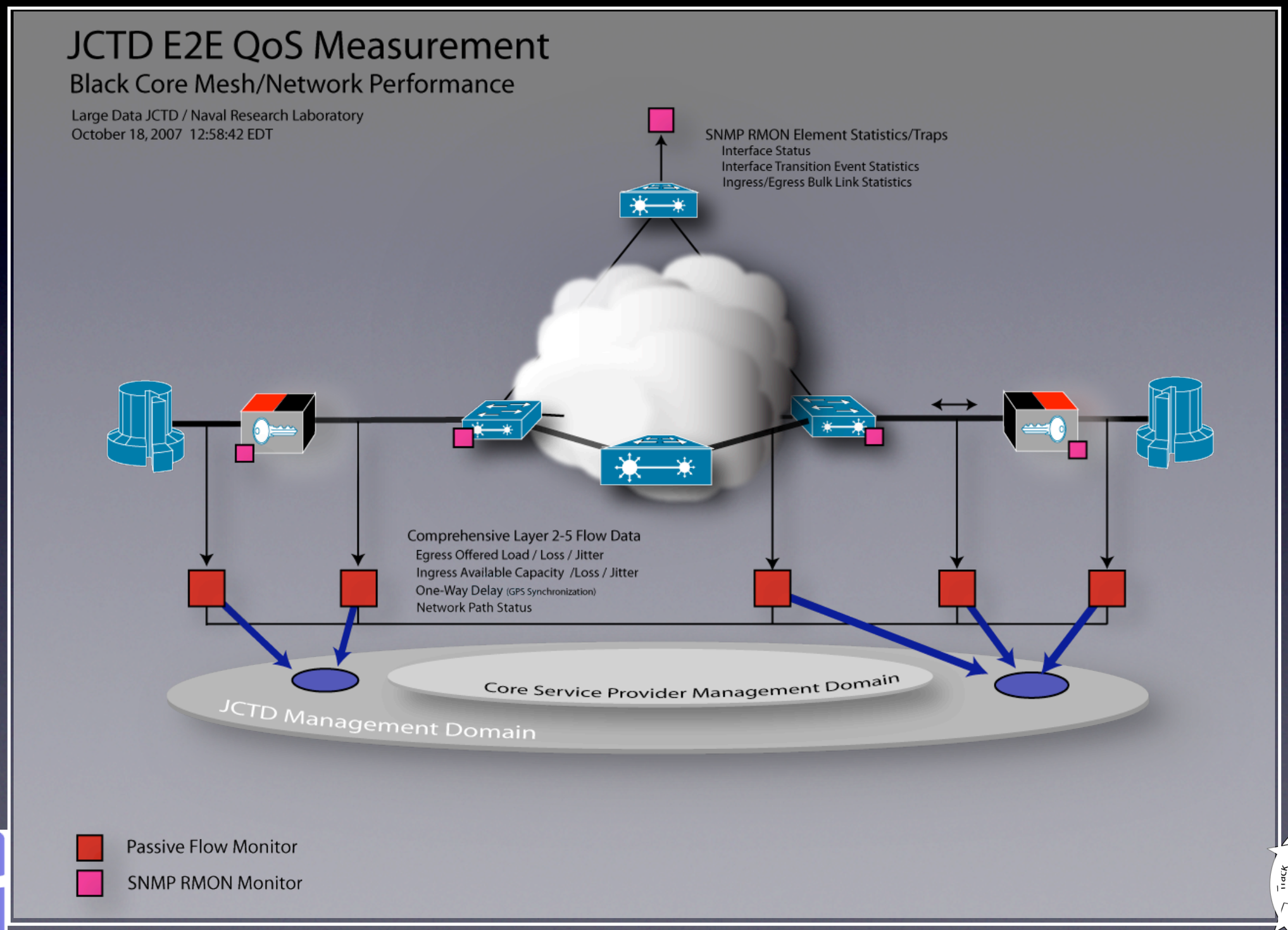
Argus writing to local Radium which is writing directly to disk and to network based clients

Data Collection

- Complex data flow machine architectures
  - Architecture of choice for scalability
    - Provides explicit scalability and performance capabilities
    - Provides most parallelism
    - Multiple points of attachment, multiple points of control
- Can get a little too complex
  - Merging of multiple flows, multiple times, introduces complex data duplication issues, and allows for complex, incompatible data schemas to co-exist

Many Argi writing directly to a Radium based distribution network, which is providing data to a set of clients.

Multi-Point Monitoring



Argus Repositories

- Argus Repository Establishment
  - Formal Ingest/Disposition
- Repository Function
  - Primitive Data Repository
    - General Archive
    - Access Control
    - Retention Policies
    - Modification Policy (Compression)
  - Derived Data Repositories

Argus Repositories

- Native File System
  - Simplicity
  - Performance
  - Compatibility
- Relational Database System (RDBMS)
  - Extensive Data Handling Capabilities
  - Complex Management Strategies
  - Performance Issues

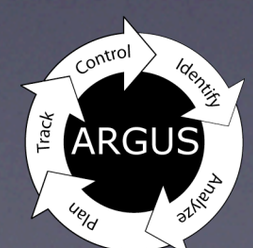
Data Ingest

- Simple Record Storage
  - File generation
  - MySQL table insertion
- Stream Block Processing
- Primitive Data
  - rasplit
  - rastream
- Dervied Data
  - rabins - controlled context rasplit with data aggregation

Best Common Practices

- File system archives
  - Primitive and derived data file systems
  - RDBMS managed complex indexing
  - rastream
    - /sourceId/year/month/day file structure
    - 5 minute files
    - 288 entries per day
    - Matches native file system performance for searching
    - Analogous to Google's Big Table filesystem
- RDBMS based archives
  - Short term data held in RDBMS
  - Rolled into file based system after N days.
  - Binary data inserted into database
  - Primitive data schema includes 'autoid'
  - Table names provide explicit partitioning

Situational Awareness



Situational Awareness

Level 1 SA - Perception

- The perception of elements in the environment within a volume of time and space
- Involves timely sensing, data generation, distribution, collection, combination, filtering, enhancement, processing, storage, retention and access.

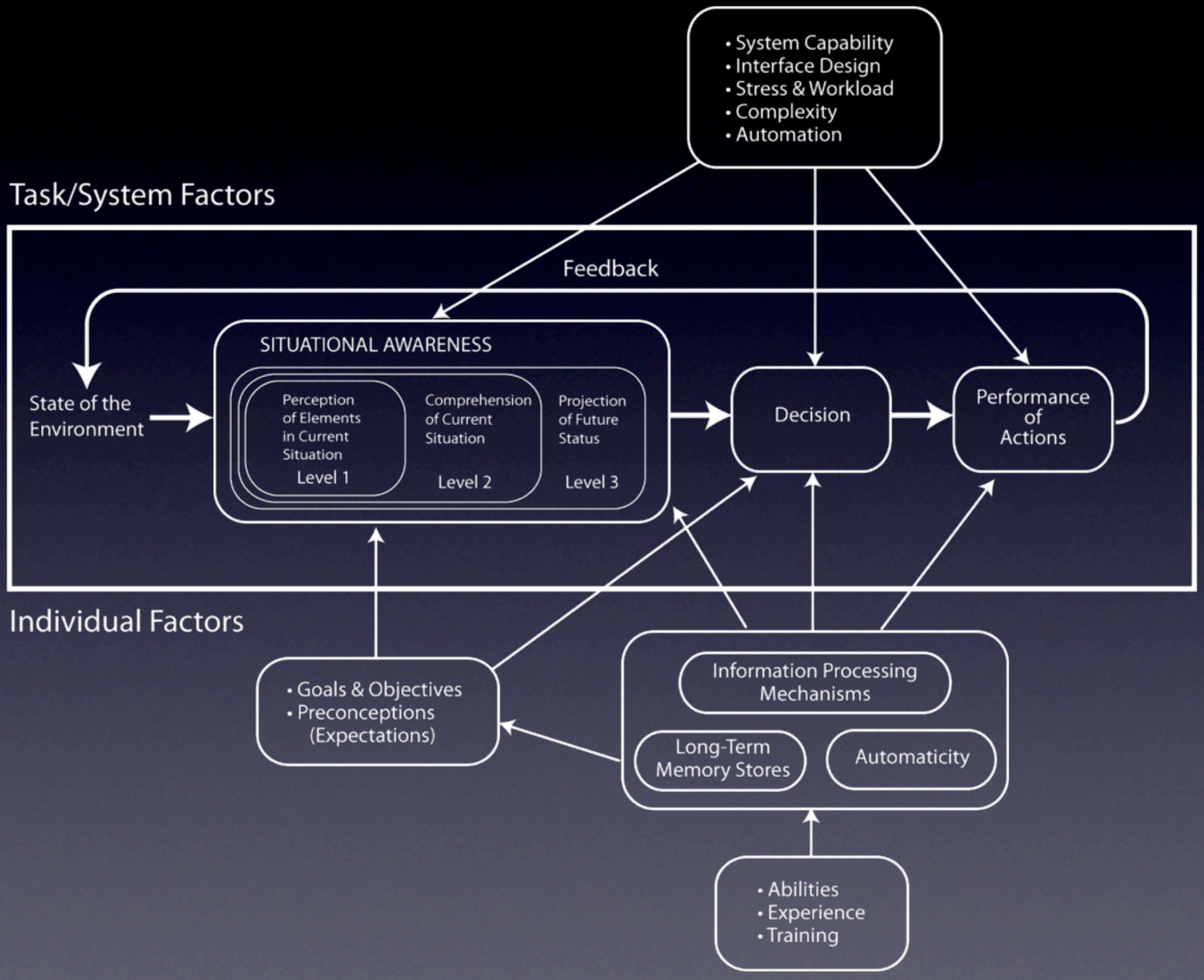
Level 2 SA - Comprehension

- Understanding significance of perceived elements in relation to relevant goals and objectives.
- Involves integration, correlation, knowledge generation.

Level 3 SA - Projection of Future Status

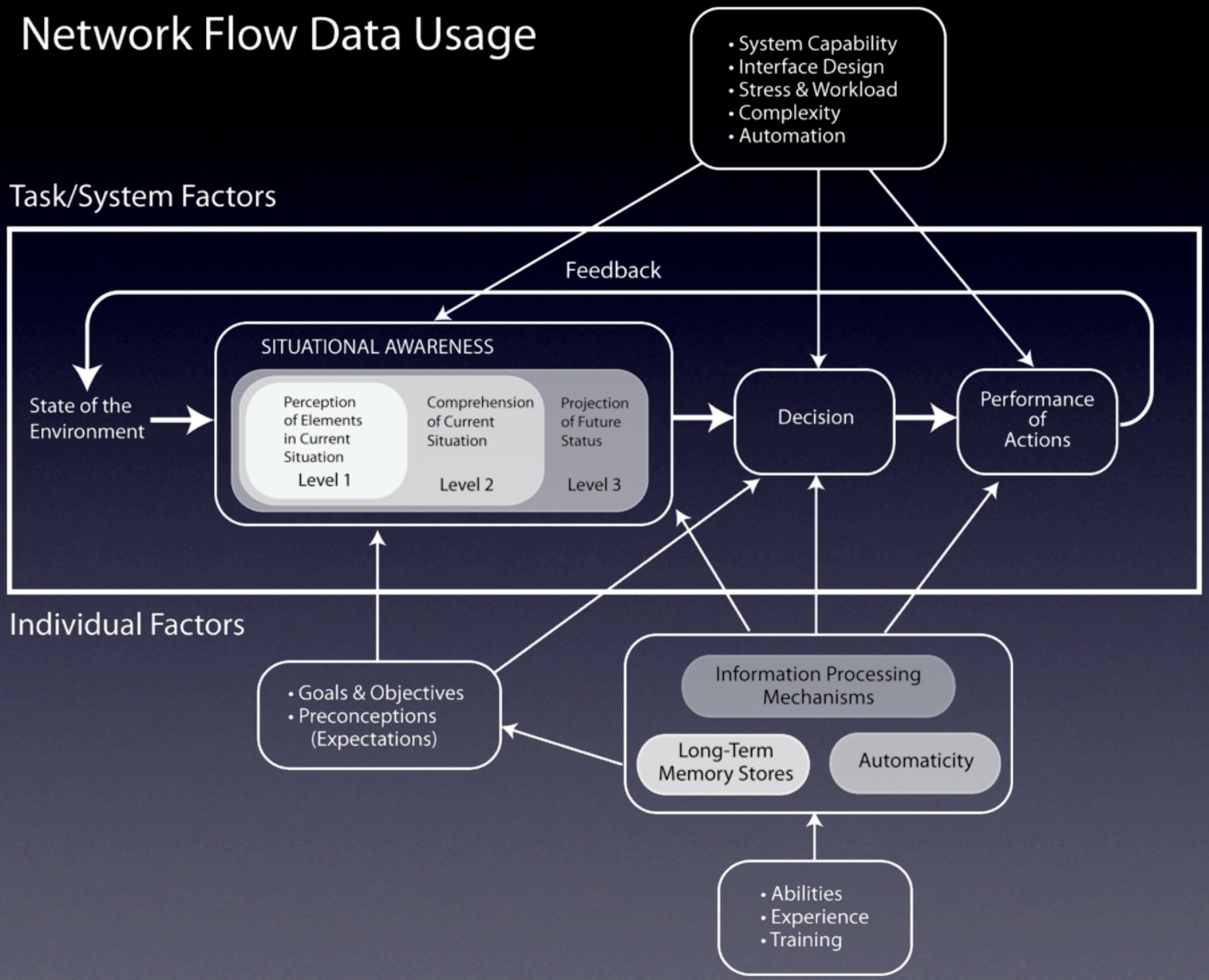
Endsley, M. R. (1995b). Toward a theory of situation awareness in dynamic systems. Human Factors 37(1), 32-64.

Model of Situational Awareness in Dynamic Decision Making



Model of Situational Awareness in Dynamic Decision Making

Network Flow Data Usage



Situational Awareness System

Basic design is local sensing, data collection and management, with local near real time data processing and large scale data sharing to support multi-dimensional network activity comprehension.

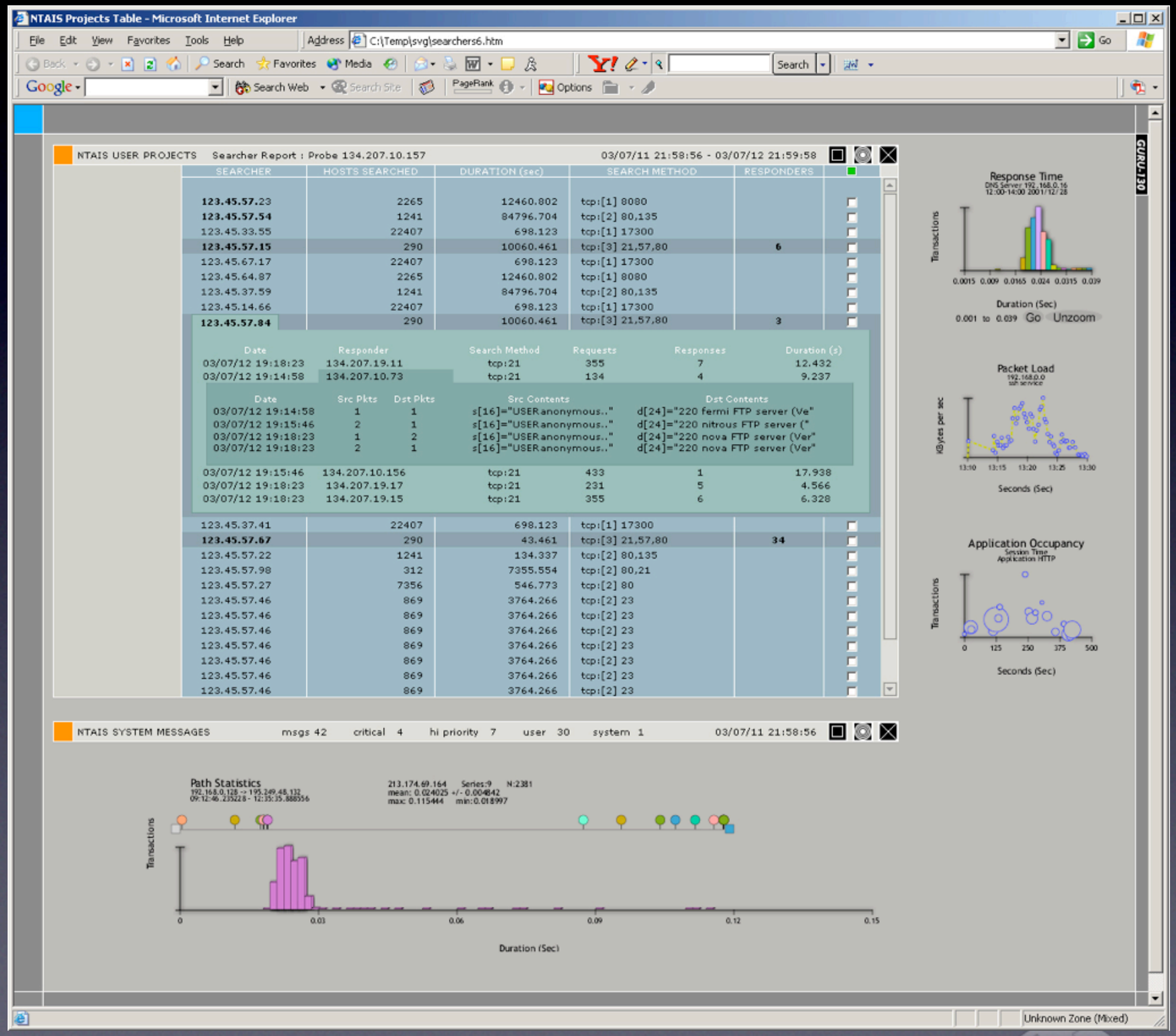
- Federated Database Model
  - Access controlled by local administrative domain (scoping)
  - Cloud-like distributed processing and query support
  - Flexible data management strategies
  - Large numbers of simultaneous users
- Near real-time information availability
  - Register for information of interest
  - Complex data processing / aggregation / enhancement / advertisement
  - Large scale data correlation processing
  - Anonymization
- Substantial historical data access

Situational Awareness

- ratop
  - Develop near-realtime view of what is going on in this network, right now.
  - Complex aggregation rule set for data representation
  - Allows for historical and current data stream comparisons
- rasqlinsert
  - ratop functions with state cached in mysql
  - goal is to have the last X minutes available
  - interface is an argus data stream

Situational Awareness

- GUI Strategies
  - Real Time Access
  - Web Based vs Native
  - Drill Down
  - Complex Data Methods
    - Time Series
      - Traditional
    - Spatial Information
  - Complex Visualizations
  - Generally not well used



Network Perception Goals

- Total Semantic Capture (Comprehension)
  - State initializations and transitions
  - Policy dissemination and enforcement
  - Topology, resource allocations, error conditions
- Context Awareness
  - Multi-Layer Identifiers (ethernet, MPLS labels, etc....)
  - Globally sychronized uSec timestamps
- Enable Near Real-Time State Awareness
  - Large scale access and data sharing
  - Multi-dimensional Correlation
- Complete Historical Reconstruction

Network Sensing Strategy

- Third-party Control/Data Plane monitor/sensors
  - Can't rely on the network switch/router vendors to do this.
- Each network device must provide complete Control Plane packet capture
  - Any packet that originates from or terminates on the device must be captured in its entirety.
  - Data must include port of origination/transmission, direction and UTC time stamp.
  - Before and after any encryption/decryption.
- Data plane flow data available for sharing, status reporting, and archival.
- Now we have the data we need to drive Data/ Control Plane Situational Awareness.

Network Information Model

- Multi-tiered Information Model
  - Not every application needs the same type of information
  - System needs to allow "customer" to define what it wants
- And, as conditions change, level of detail and frequency of status reports also needs to change

1. Data/Control Plane Service Existence Flow Strategy
   1.1. Matrix Flow with Service Identifiers
   1.2. Operational/Security Fault Status Flow Records
2. Data/Control Plane Service Performance Strategy
   2.1. Transactional Flow with Ops and Performance Attributes
   2.2. Operational Fault Status Flow Records
3. Total Packet Content Flow Strategy
   3.1. Transactional Flow with Aggregated Content
   3.2. Complete Remote Packet Capture

Packet / Flow Strategies

- Packet data for complete comprehension
- Flow data provides multi-tiered data model.
  - Data Reduction / Semantic Preservation
    - Service Oriented Transactional Abstractions
    - Complex Data Representations
    - Flexible Compression Strategies
    - Multiple Flow Content Representations
  - Semantic Access Control Schemes
    - Inter/Intra Domain Data Sharing
    - Complex Data Aggregation Scoping
    - Anonymization
  - Cross Domain/Dimensional Correlation
    - Unified Object Specifications
    - Self-Synchronization Methodologies
    - High Resolution Timestamping

Discovery Detection

Network scan detection is not as important as it was decades ago, but understanding who responds to scans, and what they respond with, is still a very important thing to know.

- radark.pl
- Track IP addresses that attempt to connect to non-existent hosts (a network explorer)
- If these network addresses ever get a response from existing nodes on non-public service SAPs, then report these accesses.
- Include what the responder responded with.

Conclusions

- http://qosient.com/argus
- argus-info@lists.andrew.cmu.edu
- carter@qosient.com