



<http://www.rti.com>

## Securing access to Distributed Pub-Sub Information in a System-of-Systems and GIG Environment

Gerardo Pardo-Castellote, Ph.D.  
Gabriela Cretu-Ciocarlie, Ph.D.

**The Real-Time  
Middleware Experts**

OMG Real-time, Embedded and Enterprise-Scale  
Time-Critical Systems workshop,  
May 2010

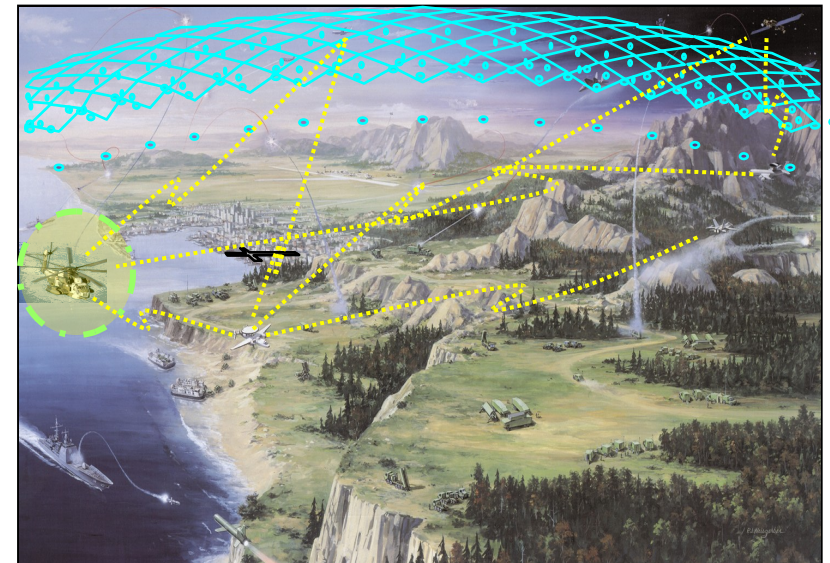
# Background

- Distributed pub/sub systems...
  - are built on top of a shared information space
  - assume information availability where and when it is needed



# Background

- Distributed pub/sub systems...
  - are built on top of a shared information space
  - assume information availability where and when it is needed
  
- As systems expand, we need to prevent adversaries from...
  - compelling DDS entities to publish
  - Fooling DDS entities into subscribing to data that is not authentic
- learning about data outside their purview

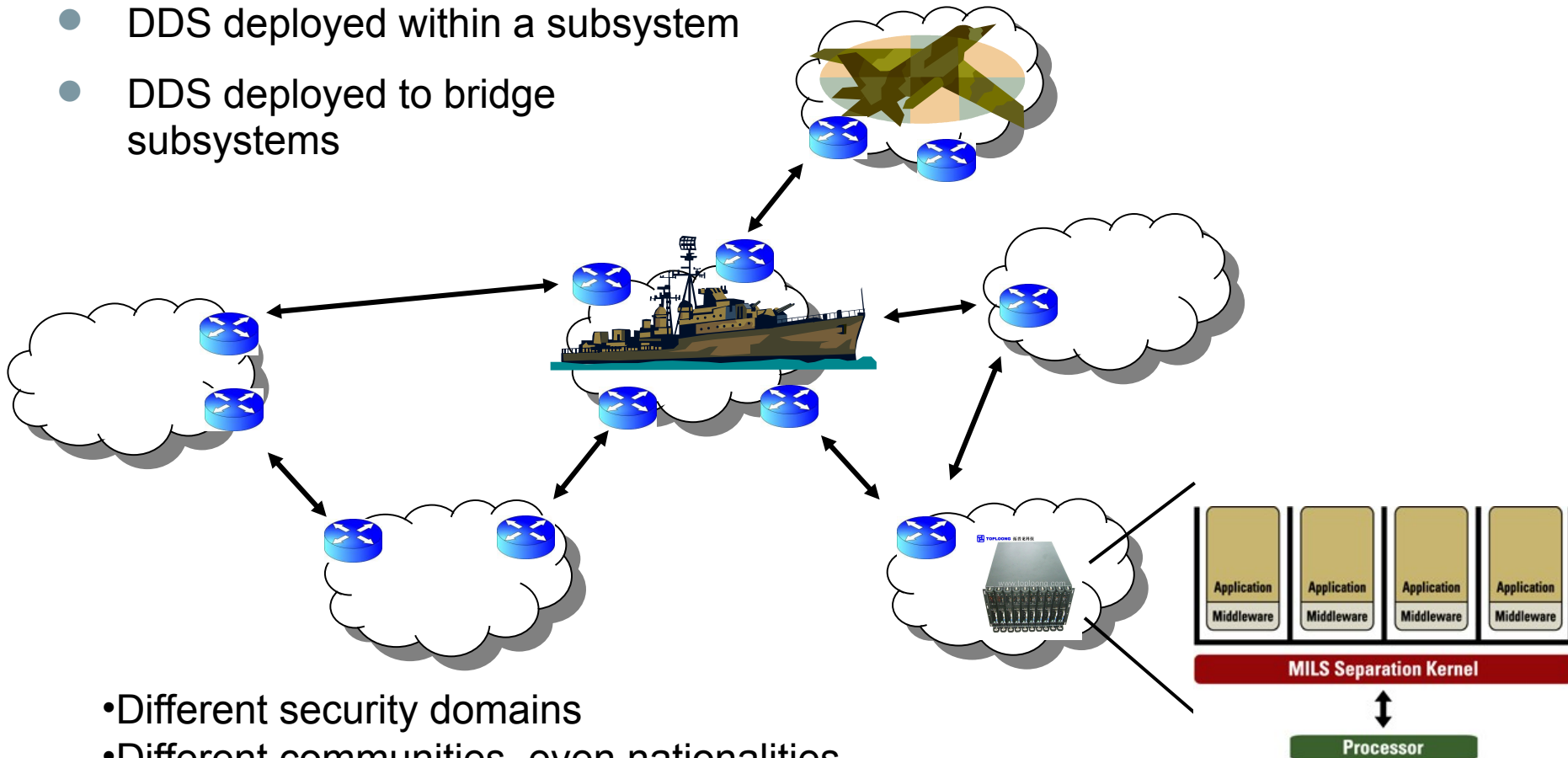


# Scope / Disclaimer

- Security problem is broad...
- GIG deploys/integrates many technologies
- This talk focuses on the aspects that concern/affect systems built using the OMG Data-Distribution Service
- It is anticipated that these systems will need to be integrated into broader security policies...
- ... this latter issue is not directly addressed by the talk.

# GIG is a composition of systems

- DDS deployed within a subsystem
- DDS deployed to bridge subsystems

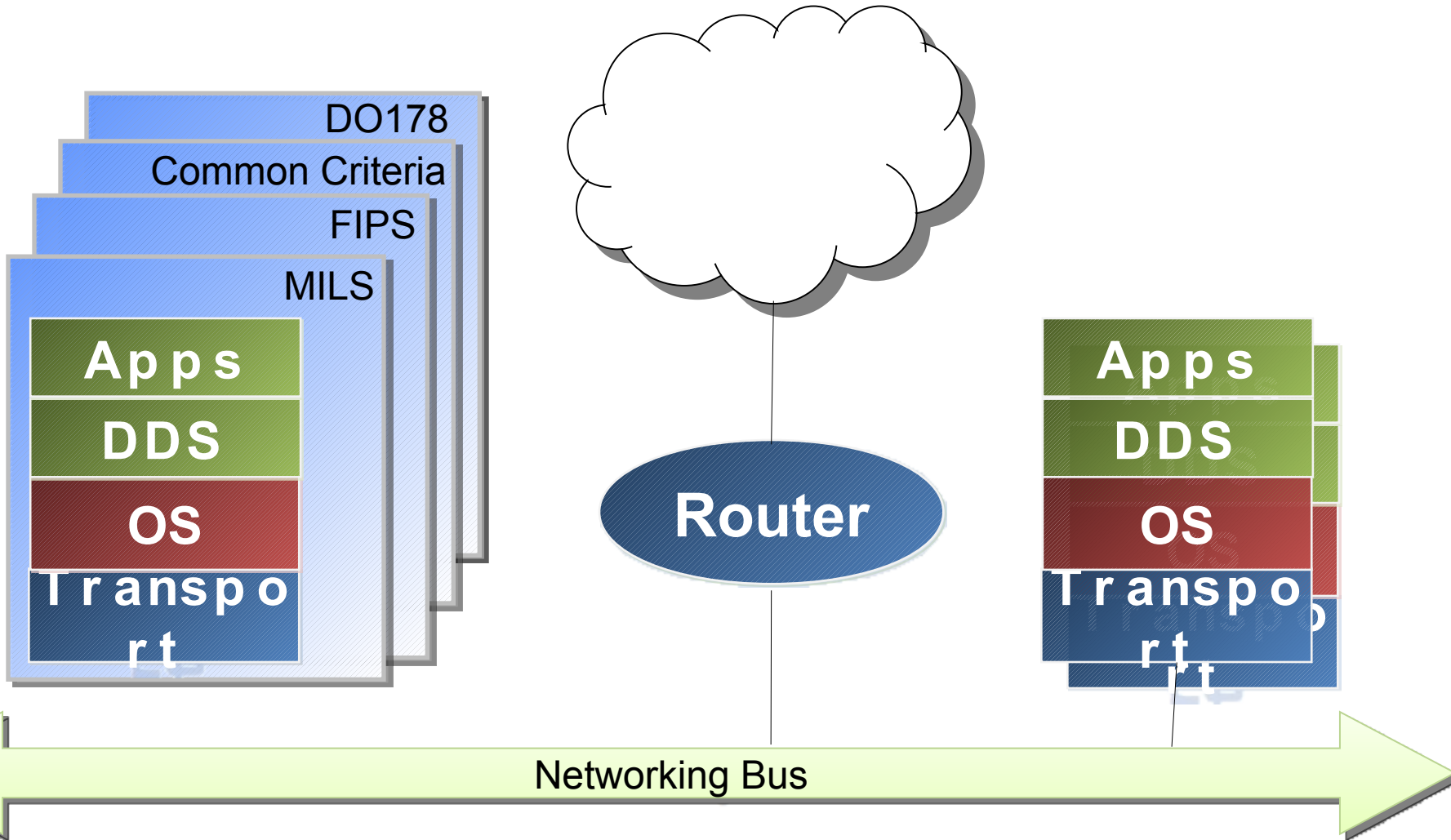


- Different security domains
- Different communities, even nationalities
- Evolving need to share – could depend on mission or political context

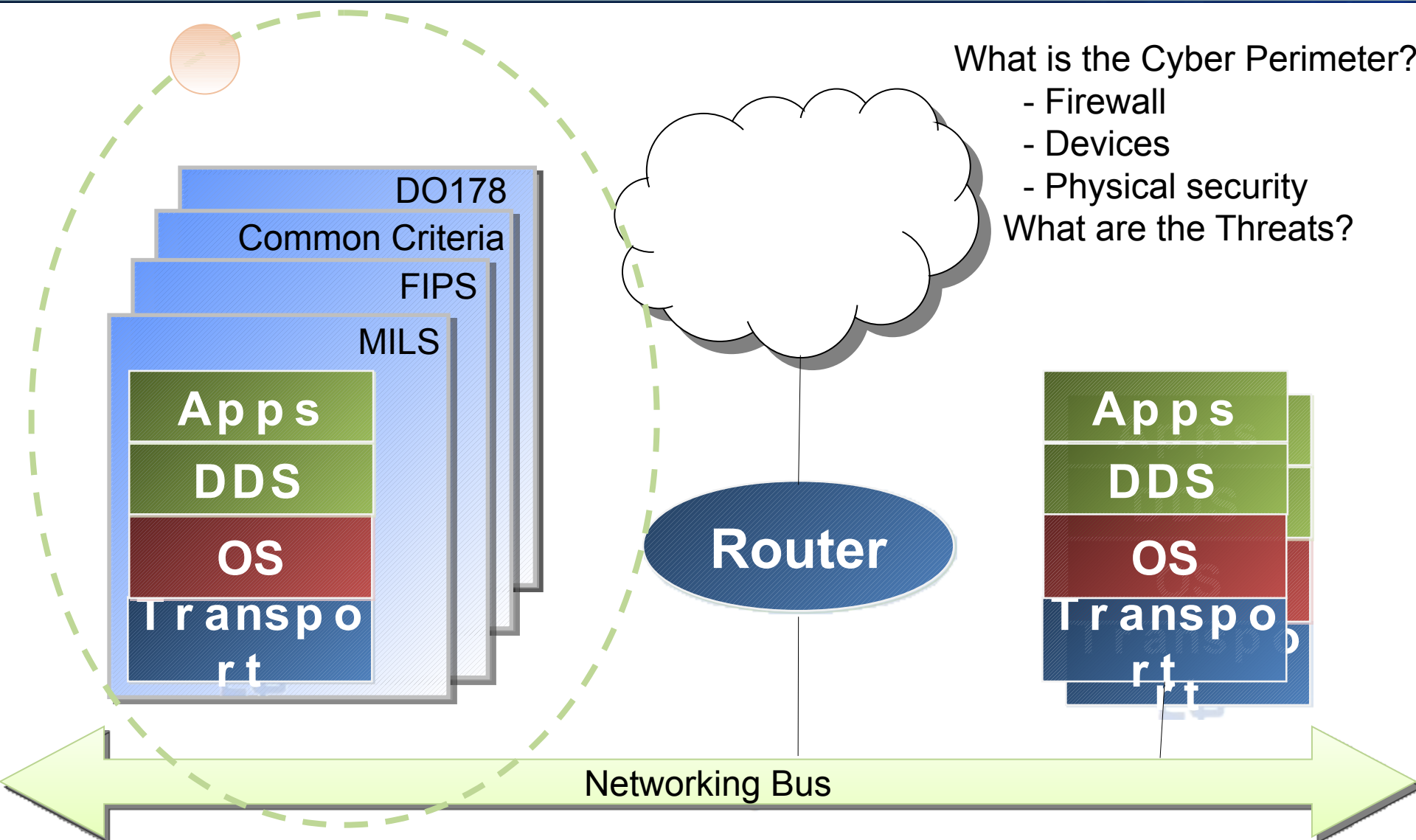
# Outline

- **Background:**
  - **Mechanisms Available**
- Securing DDS applications in a (partitioned) OS
- Securing DDS applications within a DDS Domain
- Securing DDS applications across DDS Domains
- Conclusions

# Available Mechanisms



# Available Mechanisms

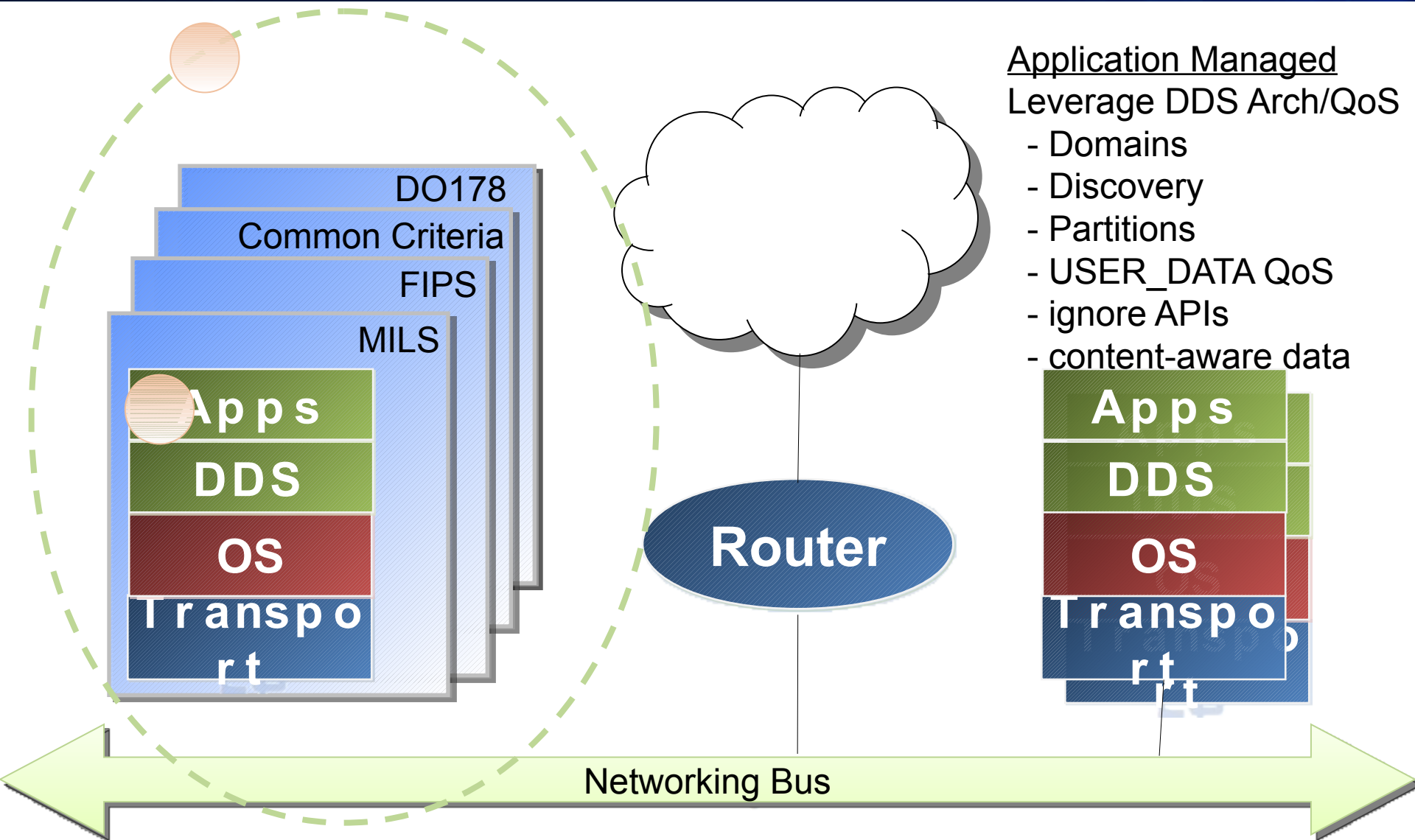


What is the Cyber Perimeter?

- Firewall
- Devices
- Physical security

What are the Threats?

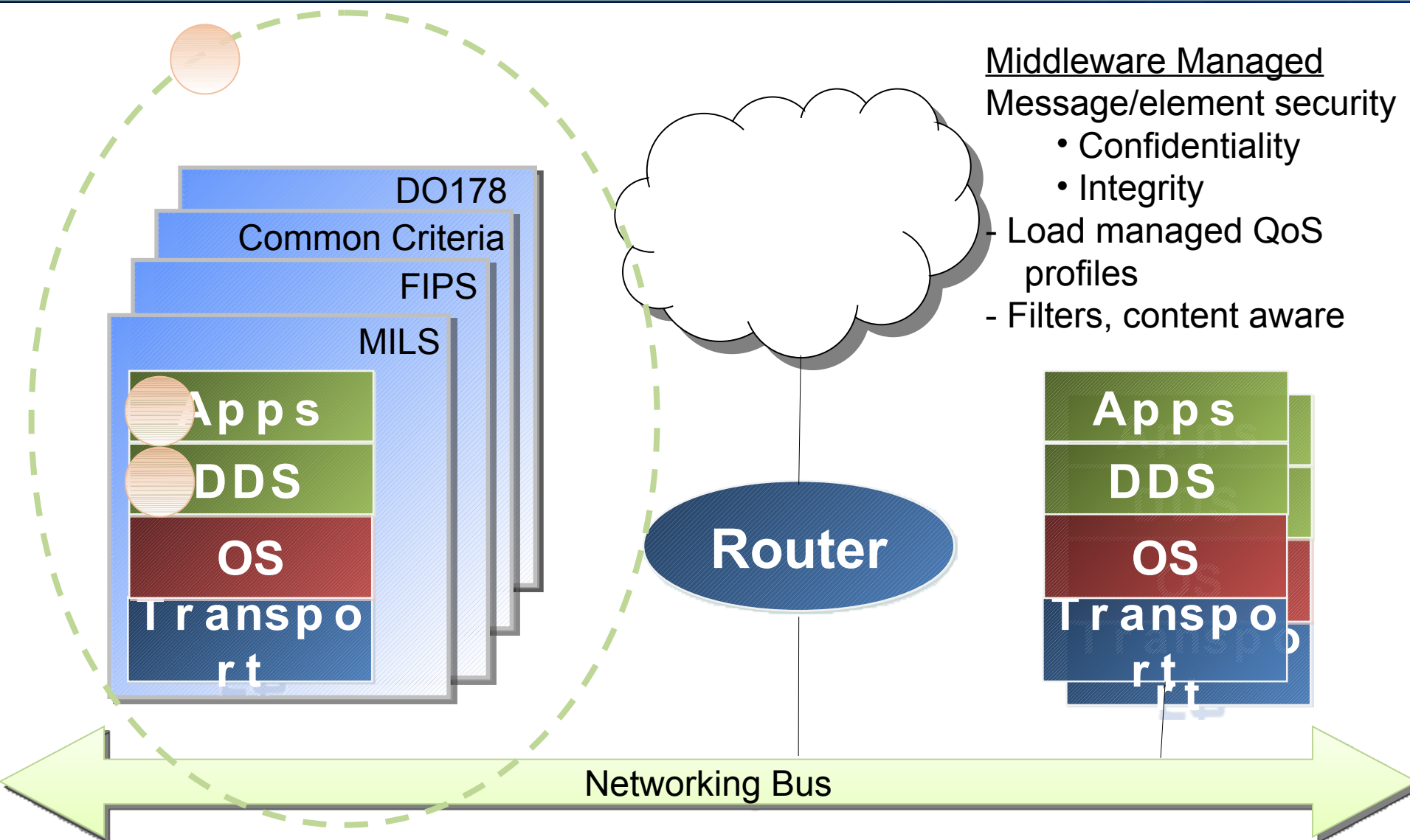
# Available Mechanisms



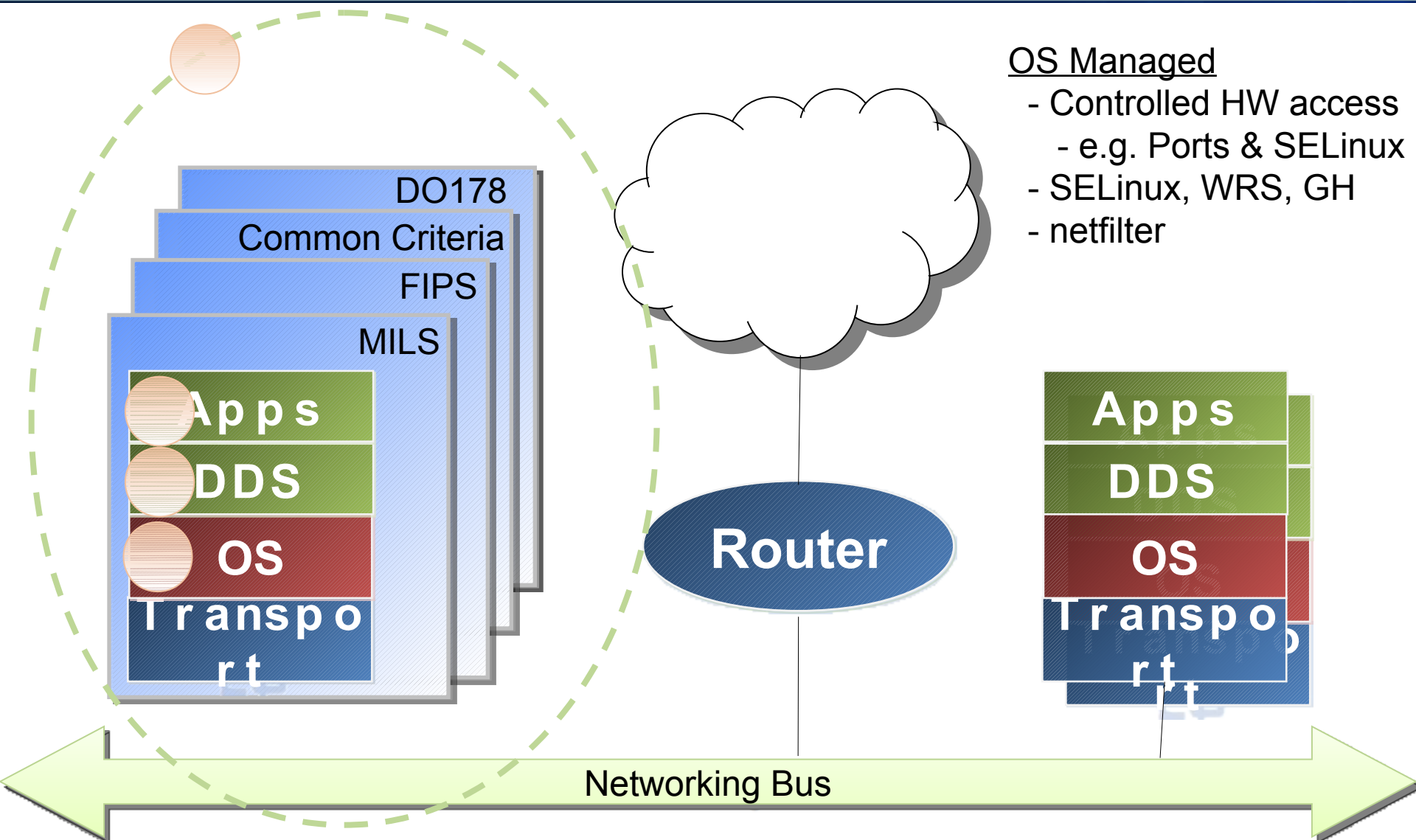
Application Managed  
Leverage DDS Arch/QoS

- Domains
- Discovery
- Partitions
- USER\_DATA QoS
- ignore APIs
- content-aware data

# Available Mechanisms



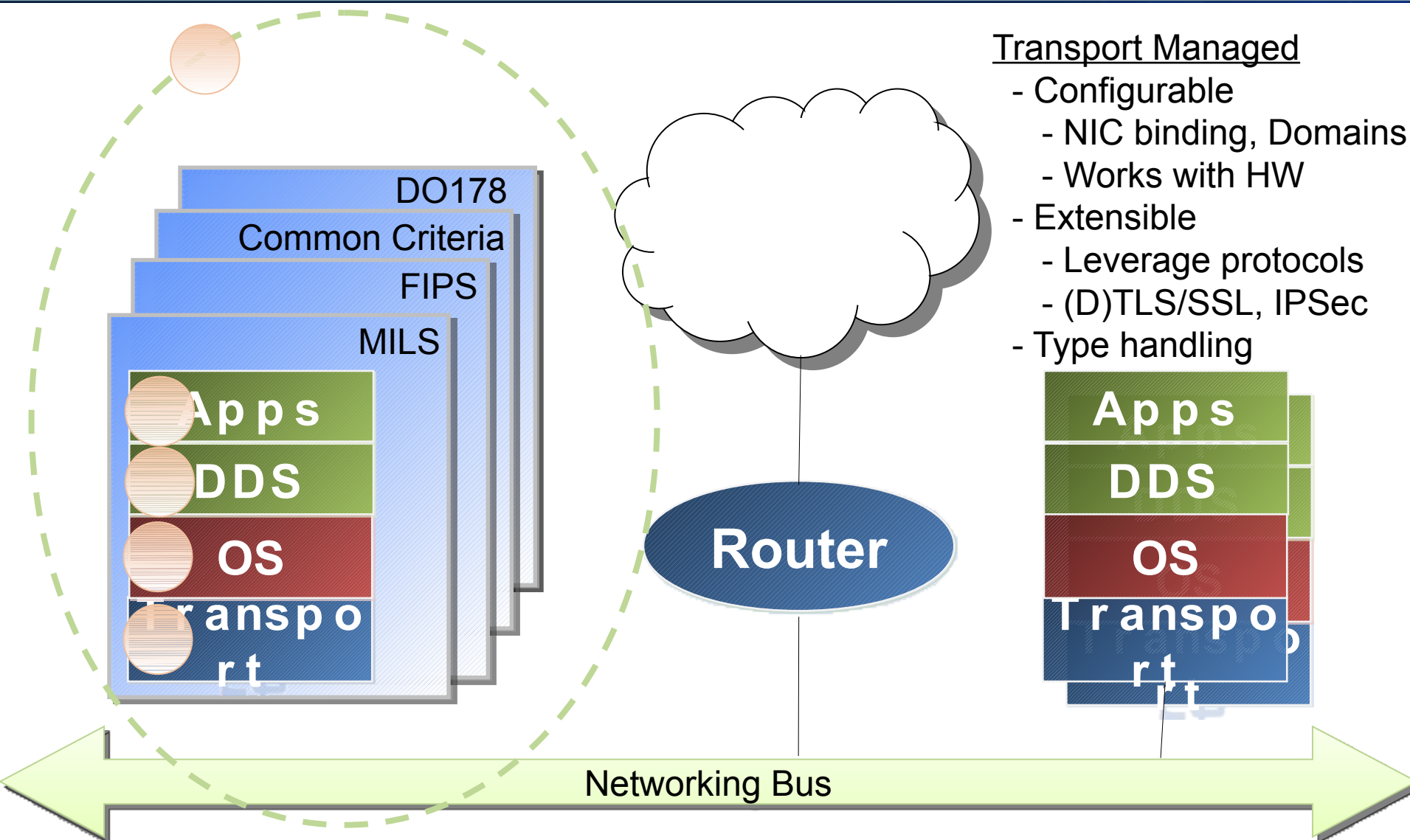
# Available Mechanisms



## OS Managed

- Controlled HW access
- e.g. Ports & SELinux
- SELinux, WRS, GH
- netfilter

# Available Mechanisms

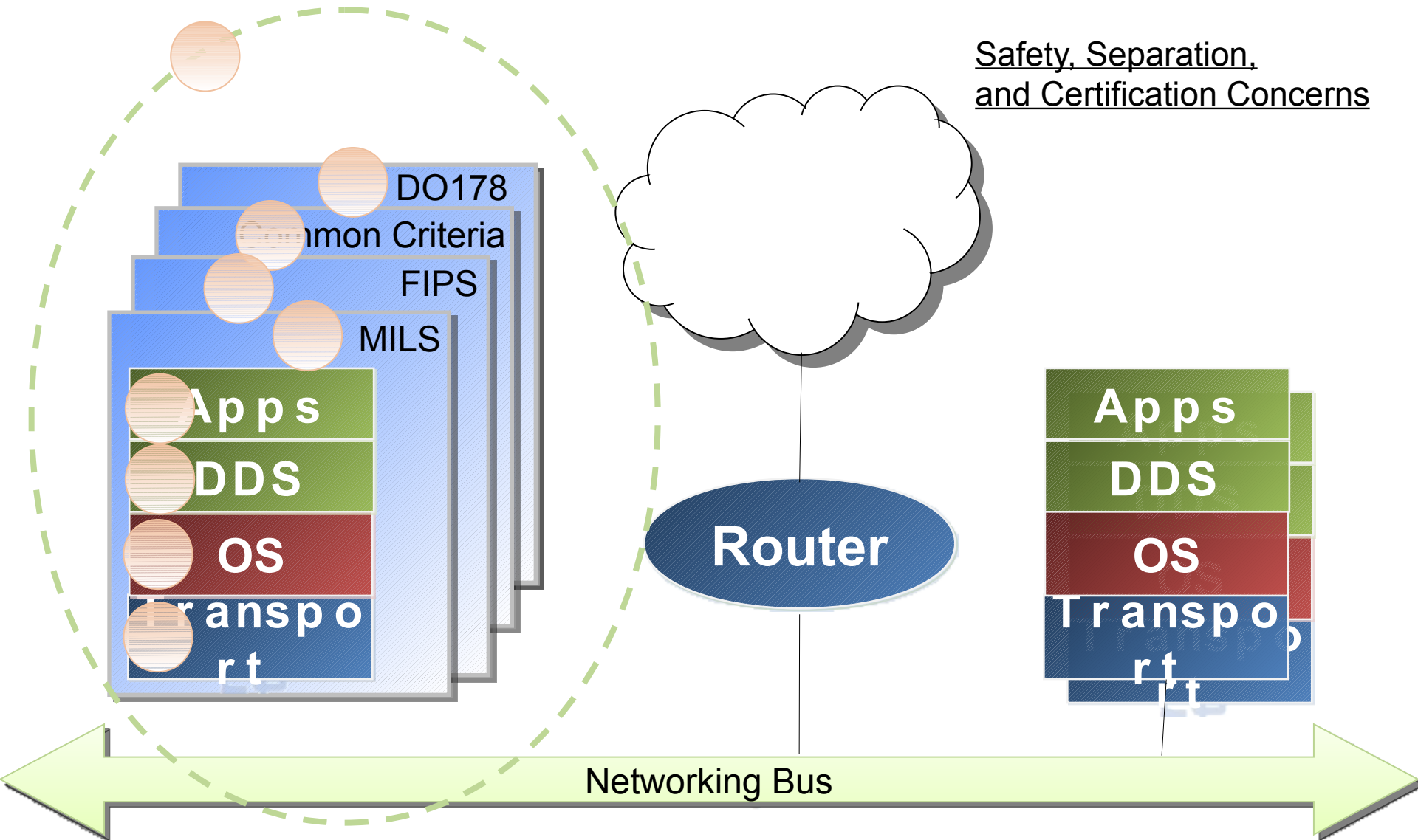


## Transport Managed

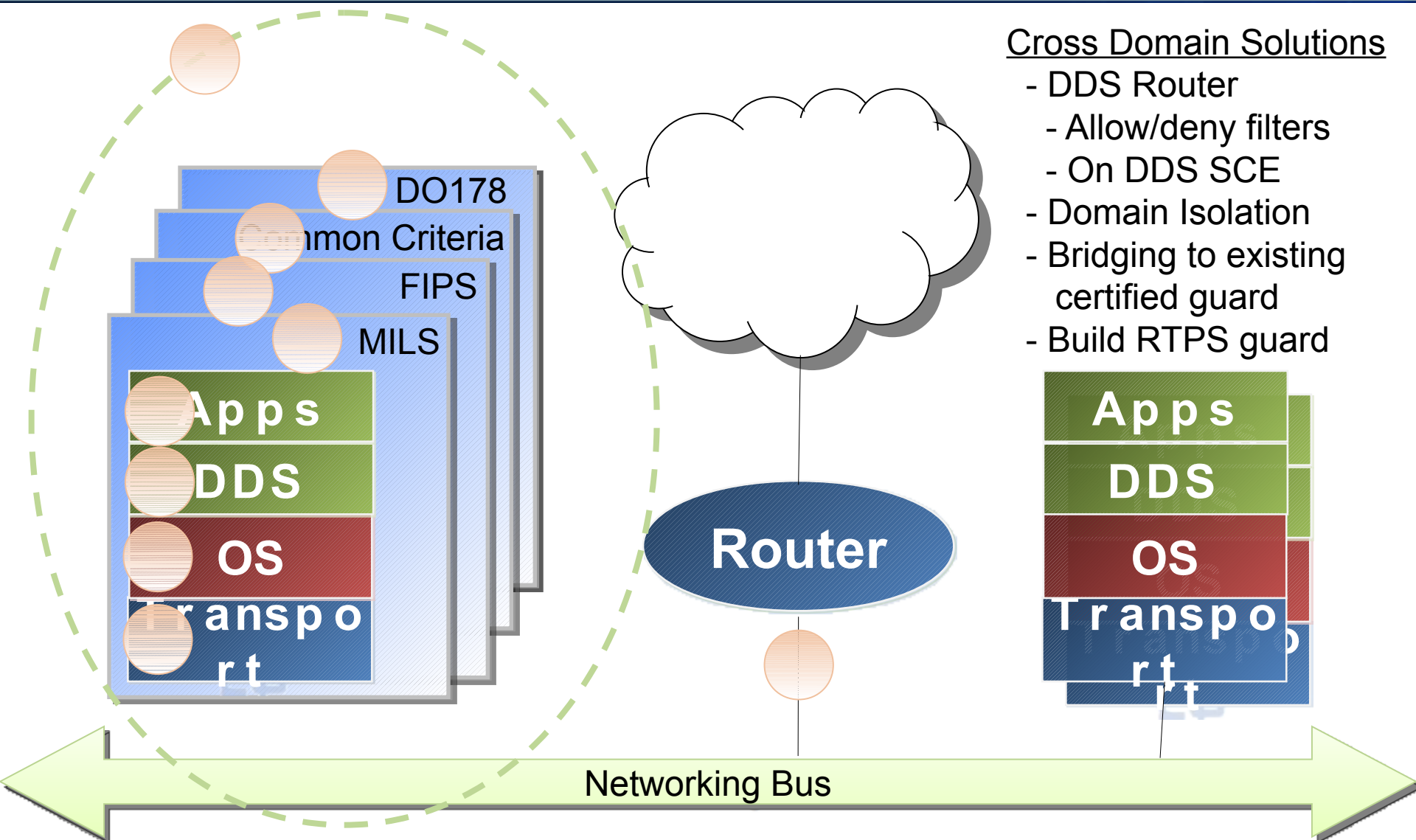
- Configurable
- NIC binding, Domains
- Works with HW
- Extensible
- Leverage protocols
- (D)TLS/SSL, IPsec
- Type handling

# Available Mechanisms

Safety, Separation,  
and Certification Concerns



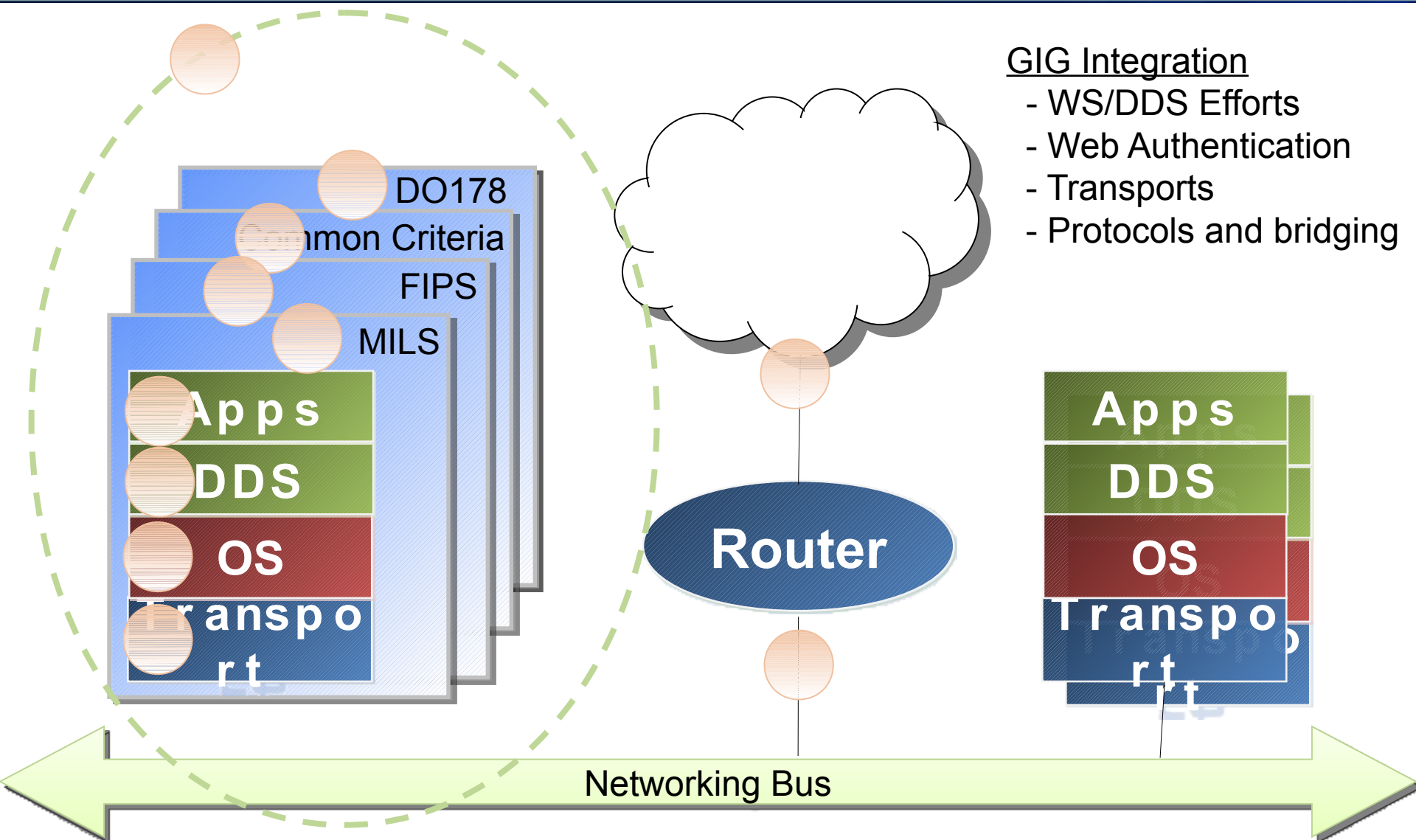
# Available Mechanisms



## Cross Domain Solutions

- DDS Router
- Allow/deny filters
- On DDS SCE
- Domain Isolation
- Bridging to existing certified guard
- Build RTPS guard

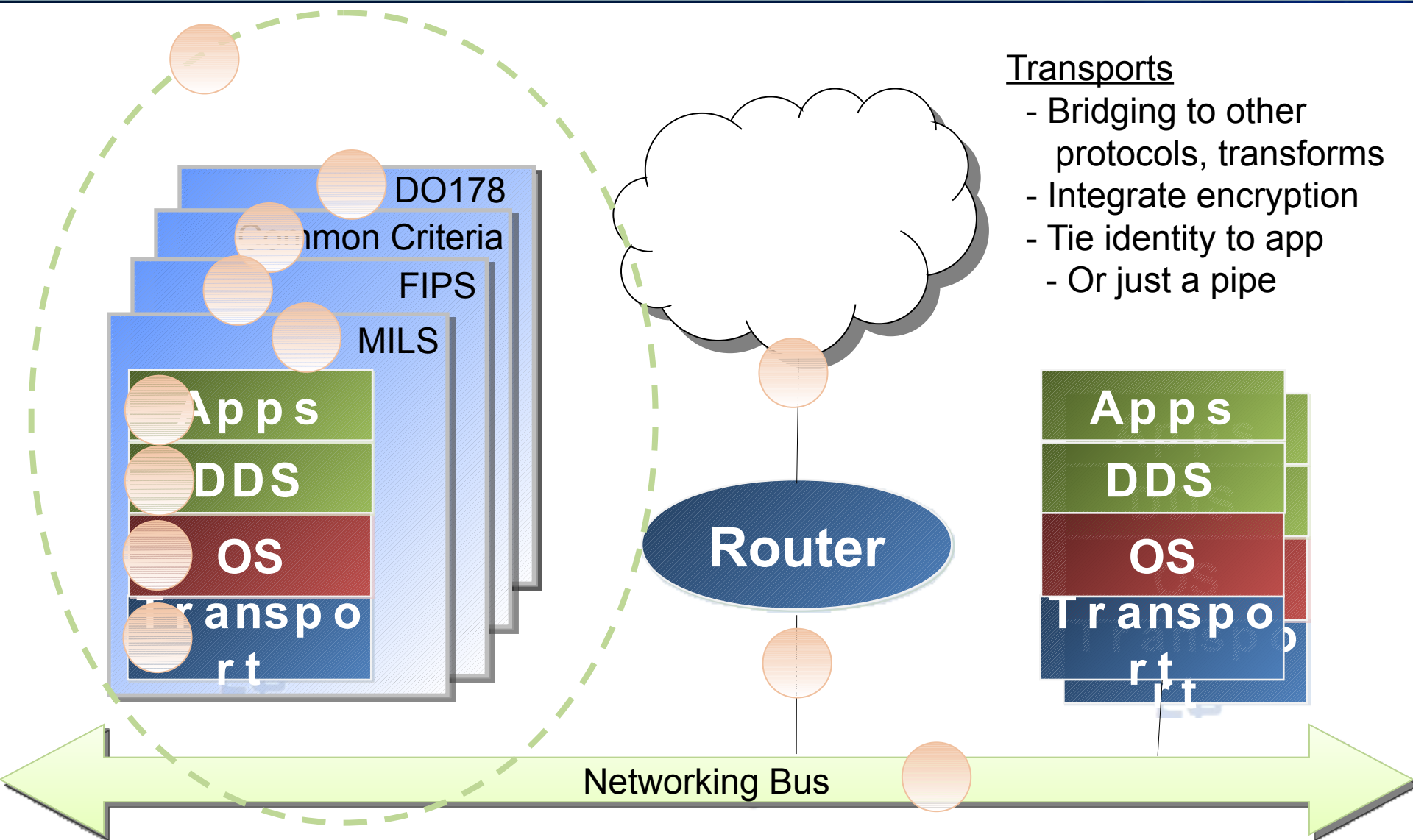
# Available Mechanisms



## GIG Integration

- WS/DDS Efforts
- Web Authentication
- Transports
- Protocols and bridging

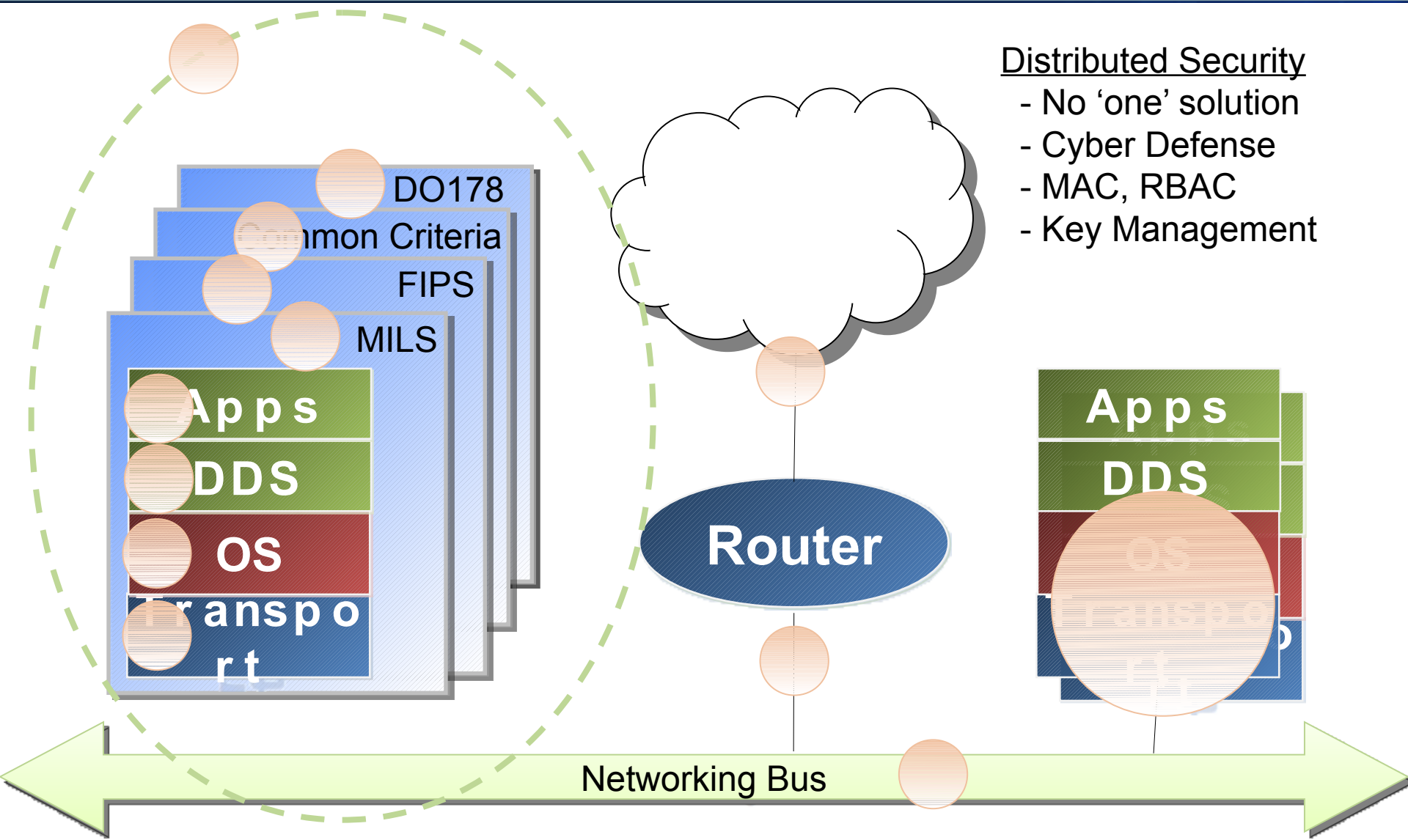
# Holistic Security



## Transports

- Bridging to other protocols, transforms
- Integrate encryption
- Tie identity to app
- Or just a pipe

# Available Mechanisms



- Distributed Security
- No 'one' solution
  - Cyber Defense
  - MAC, RBAC
  - Key Management

# Outline

- Background:
  - Mechanisms Available
- **Securing DDS applications in a (partitioned) OS**
- Securing DDS applications within a DDS Domain
- Securing DDS applications across DDS Domains
- Conclusions

# Operating System Security Integration

- Trusted operating systems (such as SELinux, Trusted Solaris) provide fine-grained access control for system resources:
  - Network ports
  - Inter-process communication (IPC) objects
- These policies can be mapped to the required resources to join a DDS domain
  - DDS Wire Protocol standard specifies ports used for discovery, default ports for user data
  - For a configured system, there is a required set of ports and/or IPC objects that must be used to join the domain
- This requires no changes to DDS, and provides another mechanism for domain separation

# Integration with SE Linux

- Linux feature primarily developed by NSA
- Integrated into the kernel version 2.6. Available out-of-the-box in distributions from RedHat, Ubuntu, etc.
- Provides hybrid of concepts and capabilities drawn from:
  - mandatory access controls,
  - mandatory integrity controls,
  - role-based access control (RBAC)
  - type enforcement architecture.
- Enforces mandatory access control policies
  - Confine user programs and system services to the minimum amount of privilege they require to do their jobs.
- Configured by means of policies
  - Reputation of being hard to program... Require expert consulting

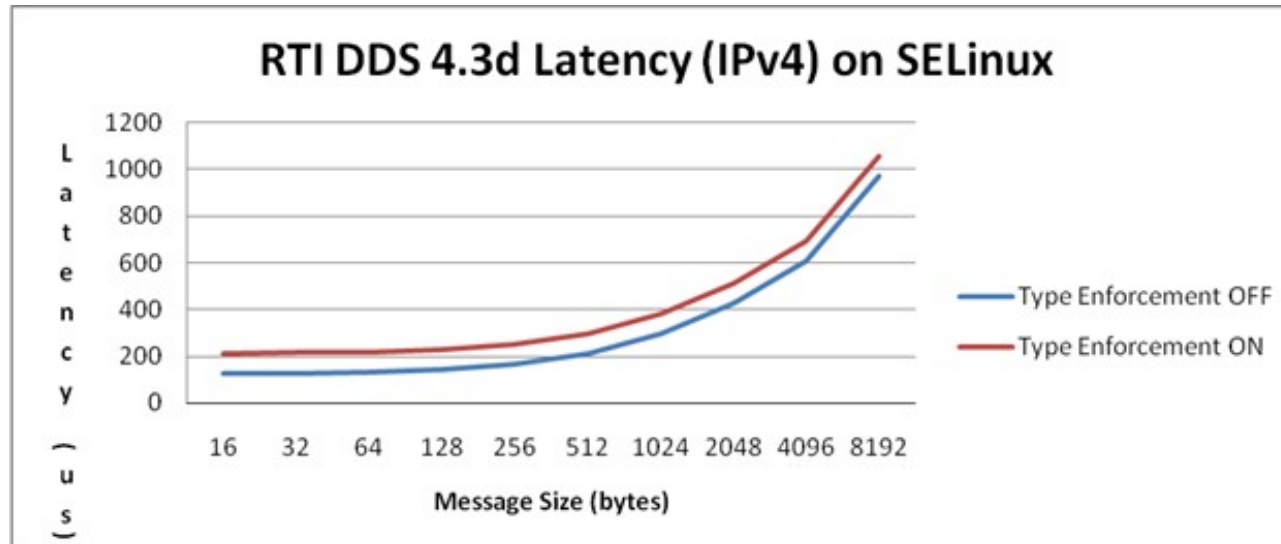
# Example: Using SE Linux to Secure DDS Domains



- SE Linux Type Enforcements can be used to restrict access to DDS domains
  - DDS maps domains to ports and SE Linux can restrict access which processes can open a given port number
  - Once in the cloud, all is available without other precautions
- SE Linux can also be used to restrict Topic access
  - DDS Interoperability allows tying Topics to separate ports
  - SE Linux generally does not have enough granularity to restrict writing to a Topic
- Provides hybrid of concepts and capabilities drawn from:
  - mandatory access controls,
  - mandatory integrity controls,
  - role-based access control (RBAC)
  - type enforcement architecture.
- Enforces mandatory access control policies
  - Confine user programs and system services to the minimum amount of privilege they require to do their jobs.

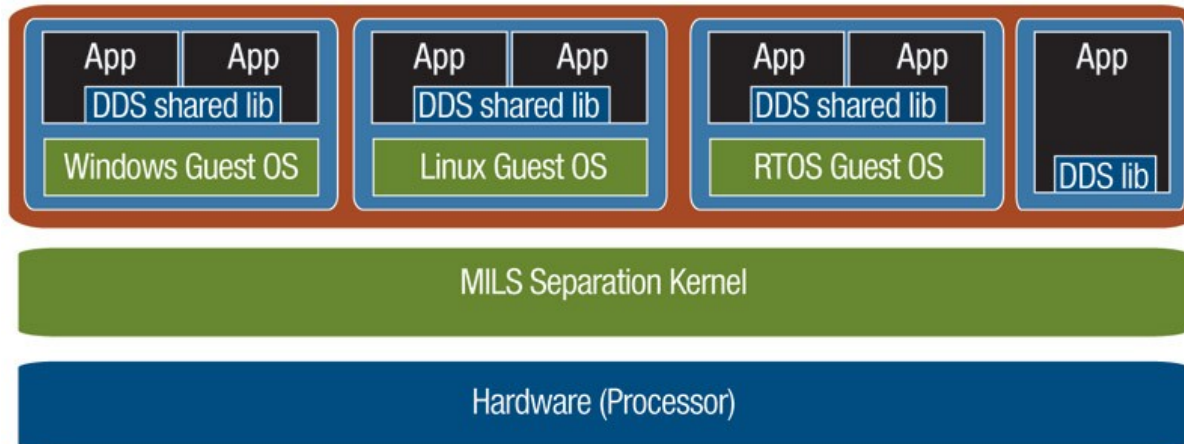
# SE Linux and DDS

- SE Security policies can be developed to limit with ports a process might send to or receive from
- Mapping of DDS wire protocol to ports allows SE Linux to enforce access control to DDS Domains within a computer.
- RTI has partnered with Tresys to develop DDS SE-Linux policies
- Performance impact of type enforcement is reasonable small:



# DDS in a MILS Kernel

- One of the MILS Kernel goals is to provide an isolated sandbox where Guest-OS and applications can run
  - Ideally the Guest OS and application runs without modification
  - MILS Kernel manages network access ensuring separation even when accessing common network hardware.
  - Cryptographic separation typically enforced by hardware TOEs
  - Information flows between different level partitions brokered by security guards (hardware/software)
- MLS partitions can be problematic for the middleware
  - Unclear whether this is required by applications or MLS partitions with be dedicated to upgrades/downgrades
- Area of active research

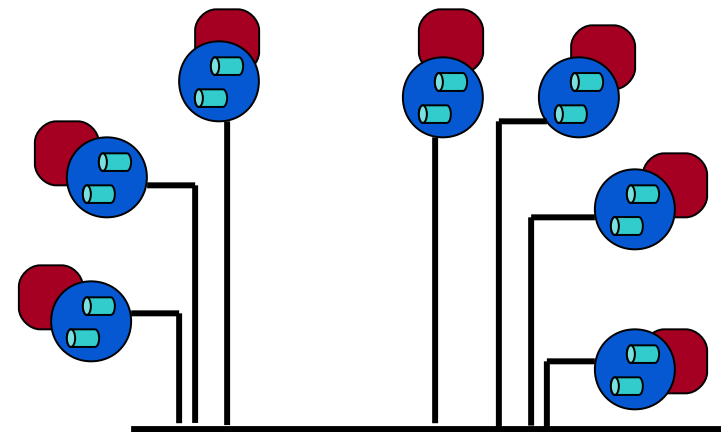
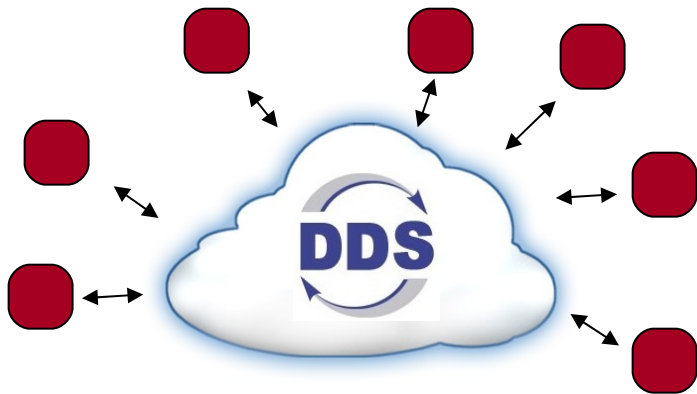


# Outline

- Background:
  - Mechanisms Available
- Securing DDS applications in a (partitioned) OS
- **Securing DDS applications within a DDS Domain**
- Securing DDS applications across DDS Domains
- Conclusions

# Security within a DDS Domain

- Applications built on DDS access the network OS stack
- OS-provided protection can limit access to a specific domain from within the OS
- Is this enough?  
... NO ..  
The OS can only protect from applications running within  
Moreover complex systems require finer grain of control



*RTPS / UDP / Ethernet*

# Standard security concerns must be addressed with the right granularity



- Authentication
  - Verify the identity of an entity (e.g., a DDS participant)  
... Must occur independently of where / which OS this is running on
- Authorization
  - Ensure that only allowed entities can perform operations  
... What are the relevant operations to protect within a DDS system?
- Data integrity
  - Data is not altered between sender and receiver
- Data confidentiality
  - Unauthorized entities cannot view data on the wire, (i.e., encryption)  
.. What is the granularity for this authorization (Domain, Topic, Content)?
- Availability
  - Data is available at all times

# Authentication

- In general authentication comes down to...
  - Something you have (e.g. certificate)
  - Something you know (e.g. password)
  - Who you are/do (e.g. as identified by a biometric characteristic, signature, etc.)
  - Or combination: Multifactor authentication (e.g. Secure ID)
- For DDS applications we must authenticate Computers / Applications / DDS participants
  - Certificates seem the most practical approach
  - Alternatively could be the combination of an IP address of a trusted computer and IPSec-type labeling provided by that computer.
  - Or in case of an interactive application it could also rely on authenticating the user that runs the application

# Authorization

- Authorization and access controls fit naturally into the publish/subscribe paradigm
- Two approaches:
  - DDS Application/ DDS Participant (User) identity
    - Traditional role-based access control (RBAC)
    - Each application is given a set of roles
    - Each role has well-defined permissions for publishing and subscribing to various topics
    - Complexity scales with the number of unique applications and roles
    - Works well in mixed-privilege environments
  - Topic-driven
    - Attribute or credential-based per topic
    - Each application given credentials to the Topics /attributes it can access
    - Complexity scales with the number of unique attributes and topics
    - Useful for strict isolation enforcement within and among DDS domains

# Authorization (2)

- Capabilities and attestations
  - Think of them as documents signed by a trusted authority that certify DDS privileges
  - “User *A* is allowed to publish topic *X*” or “Any entity holding tokens *A* and *B* are allowed to publish topic *Y*”
  - Useful for applying access control in a decentralized manner
    - Reduces communications complexity
- Labeling and/or tagging data
  - Cross domain solutions or guards, e.g., radiant mercury, often require security markers to enable Multi-Layer Security (MLS)
  - Providing hooks for data labeling facilitates interoperability in these environments
    - E.g., networks carrying both classified and unclassified data

# Underlying proposed DDS Security Model



- The approach based in two ideas:
  - Secure Domains
  - Confidential Topics
- Secure Domains
  - Limit each Global Data Space (DDS **Domain**) to contain information at a single level of security
  - Only authorized participants are allowed to join a Domain
  - All domain communications are confidential
  - All information is accessible to all members of the domain
- Confidential need-to-know Topics
  - Within a Global Data Space allow participants to read and write **Topics** on a “need to know basis”
  - Separate “right to read” from “right to write”
  - Each Topic is authorized and protected separately

# Underlying proposed DDS Security Model: Secure DDS Domains



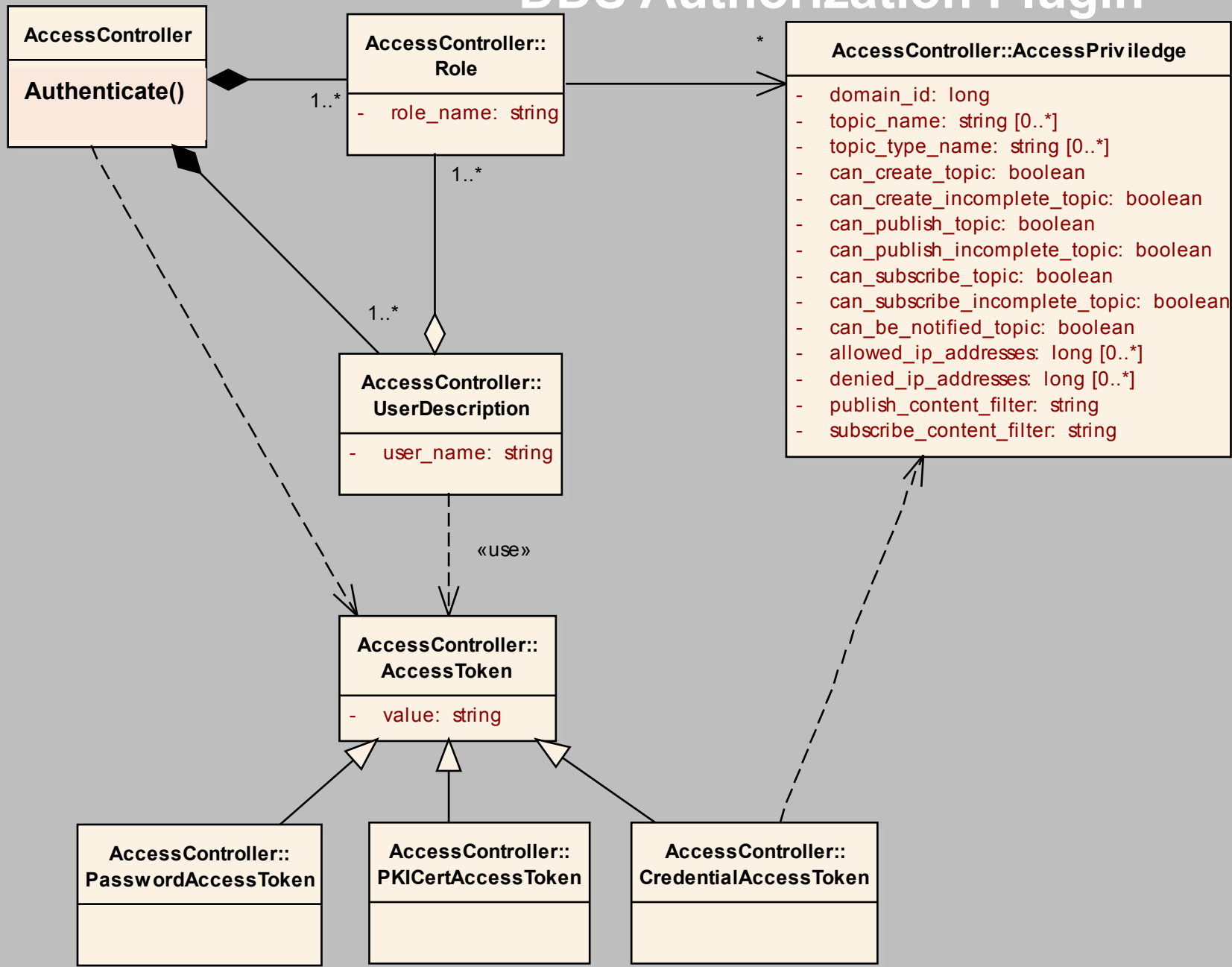
- Each DDS **domain** implements MAC at a single security level.
- Each DDS domain is configured with a CA. The PK of the CA is compiled into each application
- The CA gives certificates to each Participant that is allowed to join the domain
- When participants discover each other they use the pre-configured CA PK to verify their peer's certificate to join the domain
- Result:
  - Limits access to the domain only to the principals (the DDS **Participants**) that have the proper security clearance.
  - Mandatory Access Control (MAC) is therefore applied to each DDS **Domain** separately.

# Underlying proposed DDS Security Model: Confidential DDS Topics



- The RBAC model applied to DDS Topics
- Each **Topic** is assigned a list of “reader roles” and a list of “writer roles”.
  - ‘reader roles’ are the roles of the principals that can read the Topic
  - ‘writer roles’ are the roles of principals that can write the Topic
- Each **Participant** attached to the **Domain** is assigned a set of roles.
  - Write access to the **Topic** given only to **Participants** that have a role that appears in the list of “writer roles” for the **Topic**
  - Read access to the **Topic** given only to **Participants** that have a role that appears in the list of “reader roles” for the **Topic**
- Result:
  - Limits access to the Topic only to the principals (the DDS **Participants**) that have the “need to know” for that Topic
  - A single domain can carry traffic of multiple security levels

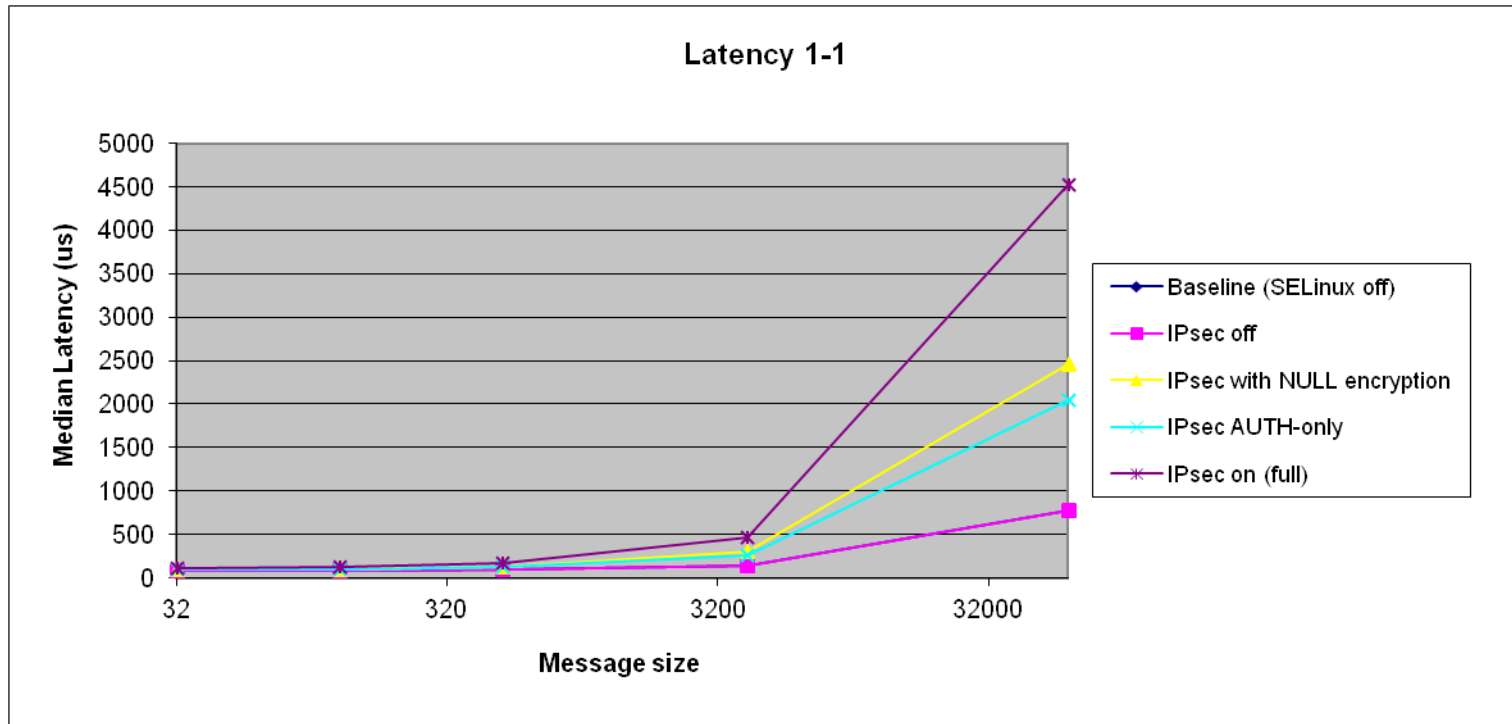
# DDS Authorization Plugin



# Confidentiality

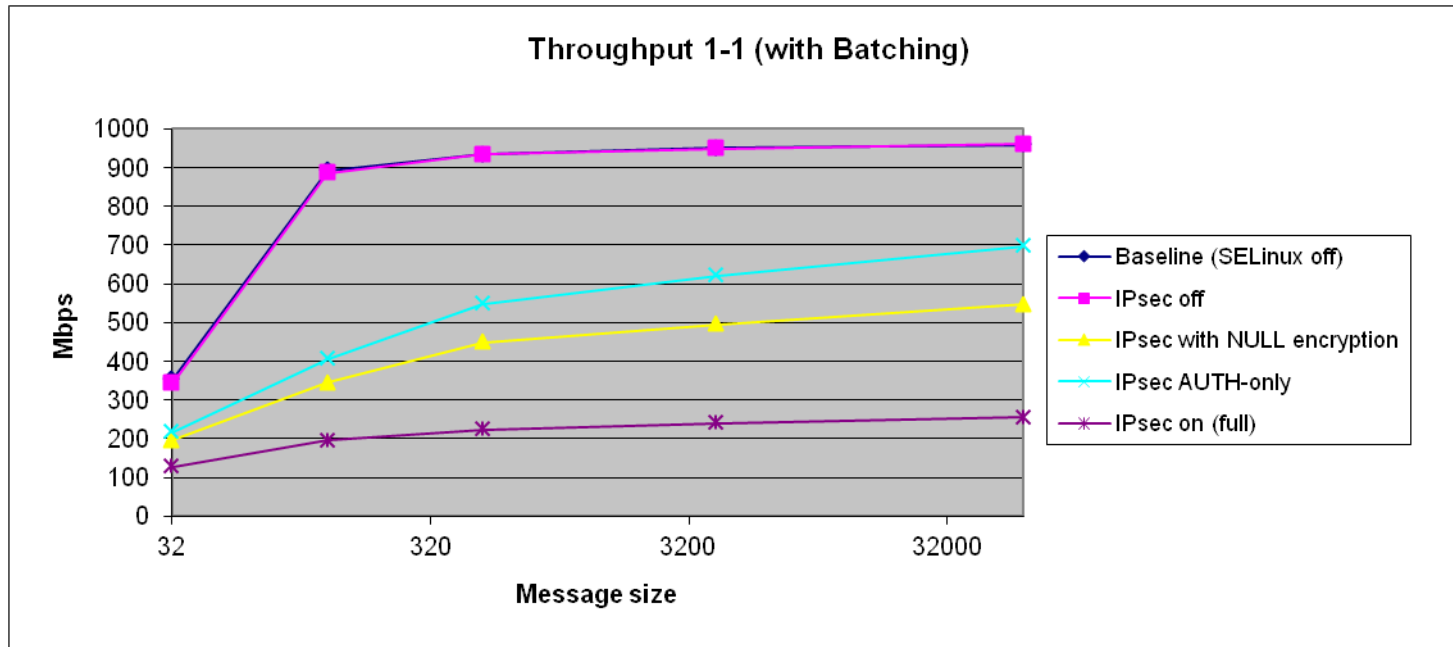
- Granularities
  - Data/Element:
    - Symmetric and asymmetric encryption possible from application layer via data marshalling/serialization handlers
  - Message:
    - IPsec
    - SSL/(D)TLS
  - Inter-domain:
    - DDS Routing Service, TCP Transport (SSL/TLS)
- Considerations:
  - Key management and distribution
  - Performance implications
  - Multicast restrictions (e.g., asymmetric encryption)

# Performance Impact Study: Median Latency



- Note: Performance is CPU bound and will vary by type
- Latency of data is very similar for the baseline and SELinux cases
- Overhead for IPsec is not fixed
  - The cost increases as the packet size increases

# Performance Impact Study: Throughput



- Batched throughput is very similar for the baseline and SELinux cases
- Significantly lower throughput with full IPsec
  - Maximum throughput is reached quickly due to batching of small packets
  - Maximum difference in throughput between baseline and SELinux: 15.6 Mbps

# Data and Message Integrity

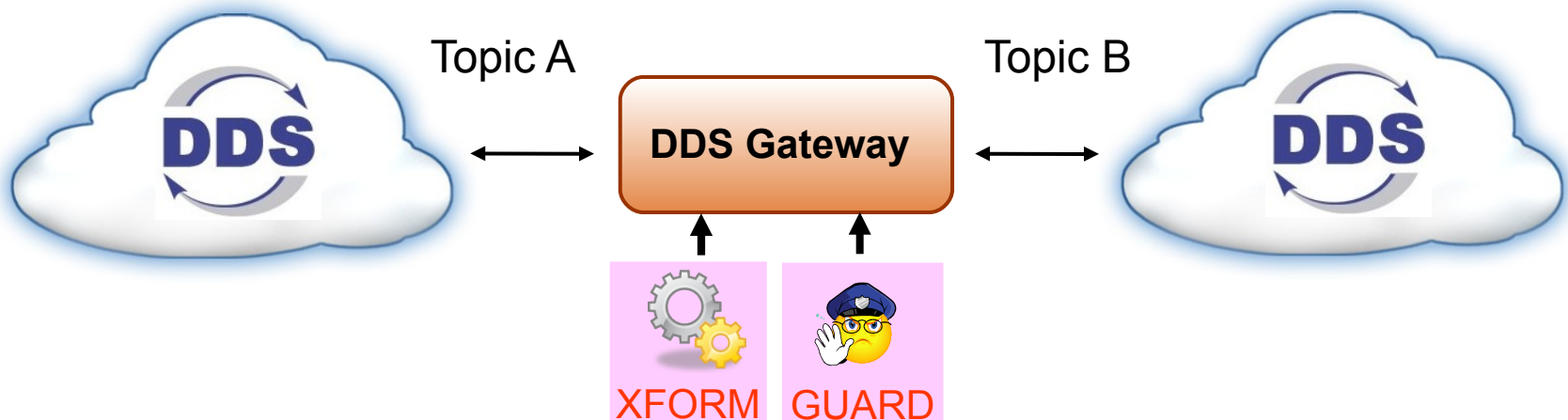
- Data Integrity via Application Layer
  - Symmetric crypto:
    - Message Authentication Code (MAC) computed on data elements or payload in marshalling/serialization handler
  - Asymmetric crypto:
    - Digital signatures computed on digest of payload or element(s)
    - Provides non-repudiation
- Integrity via Transport/Network Layer or Routing Service
  - Integrity and non-repudiation included “for free” with IPsec, TLS/SSL encryption
- Similar performance and key/identity management considerations from previous slide

# Outline

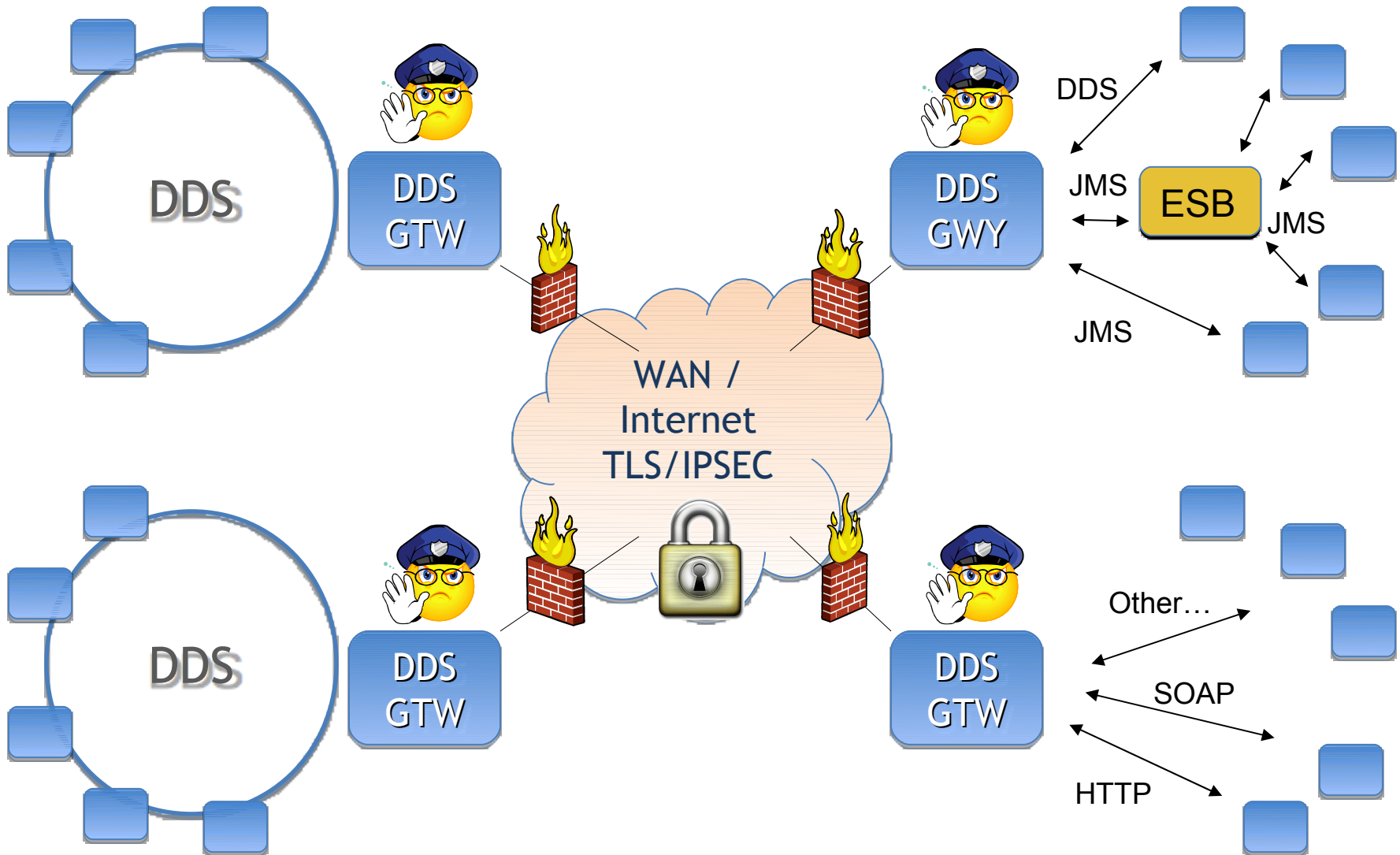
- Background:
  - Mechanisms Available
- Securing DDS applications in a (partitioned) OS
- Securing DDS applications within a DDS Domain
- **Securing DDS applications across DDS Domains**
- Conclusions

# DDS Domain Authentication, Isolation

- Same authentication techniques can be applied to DDS “DDS Gateway” applications bridging the Domains
  - DDS Routing Service (Bridge)
  - Web-Enabled DDS Gateway
- Additional access control policies might be applied at the DDS Gateway
  - Expose only Certain Topics
  - Remove fields
  - Introduce delays
  - Introduce Noise



# Domain Isolation in a System of Systems



# DDS Security: What is missing?

- What should DDS security look like?
  - A security model defined for DDS applications
  - A flexible set of extensions/interceptors in the DDS API
  - Standard security mechanisms built-in to DDS-RTPS protocol
  - Standard mapping of DDS-RTPS to TLS
- Adding these mechanisms to the middleware does not obviate or undermine the aforementioned security mechanisms outside the middleware
  - Security mechanisms in the middleware can be used in conjunction with other mechanisms
  - However, if the goal is to secure middleware messages and participation, it should be defined in the middleware

# Conclusions

- There are some mechanisms within DDS that can be used achieve common security features
  - Unfortunately, effectively securing DDS systems in this manner requires a lot of system tuning and specialized knowledge
  - Defending and managing systems using these mechanisms can be ad-hoc and will not be interoperable
- What do we want instead?
  - Standardization of DDS mechanisms and security model
    - Added convenience
    - Reduce development time and complexity
    - Encourage security engineering
    - Achieve interoperability
    - Integrate with external security infrastructure