DDS in Multi Level Security Environments for System F6

Abhishek Dubey, Andy Gokhale, Gabor Karsai, William R. Otte; Vanderbilt University/ISIS
Paul Calabrese, Adam Mitz; OCI

This work was supported by the DARPA System F6 Program under contract NNA11AC08C.
Future Fast, Flexible, Fractionated, Free-Flying Spacecraft (F6)

- Objective: Develop and demonstrate a satellite architecture where the functionality of a traditional monolithic spacecraft is replaced by a cluster of wirelessly connected modules

- Advantages:
  - Increased flexibility during design and acquisition
  - Reduced development and launch costs
  - Increased adaptability and survivability of space systems on-orbit
  - Potential to apply economies of scale to satellite design & manufacture

- Key program objective is the promulgation of open interface standards for hardware and software.
High Level System F6 Technical Goals

1. Distributed system with network addressability
   - Everything and anything (modulo security permissions) can be accessed and addressed

2. Cope with highly variable network quality & availability
   - Inter-satellite links are highly unreliable with unpredictable bandwidth; ground links are infrequent and flow

3. Dynamism
   - Dynamically deployed applications, security configurations, and cluster architectures

4. Resource sharing
   - Specific resources can be shared across applications: CPU, communication links, memory, services

5. Fault tolerance
   - Faults in components, services, communication links, computing nodes are detected, isolated, and their effects mitigated

6. Multi-level security
   - The architecture enforces mandatory access control based on MLS
Solution: F6MDA (Model-driven Architecture)

Layered architecture supported by a model-driven development toolchain
Solution: F6OS

- The ‘Operating System’ that provides
  - Restricted OS calls for application actors
  - Privileged calls for platform (‘service’) actors
  - All system calls are time-bounded

- Provides messaging services
  - All component interactions are via messages
  - No other interactions are possible

- All component interactions are facilitated by a ‘secure transport’ that verifies security labels on messages

- Resource management functions
  - CPU time: temporal partitioning for actors, utilization cap per actor within partition
  - Memory: space partitioning, limit caps
  - Network bandwidth: diffserv, bandwidth budget, differentiated routing

- F6OS is part of the TCB

F6OS enforces Mandatory Access Control (MAC) on all inter-actor interactions
F6OS Secure Transport (1/2)

**Goals:**
- Secure information sharing (confidentiality, integrity, authenticity)
- No unauthorized information flows
- (Application) Actors do not have to be trusted
- Support for real-time (‘timely’) communication and fault tolerance
- Individual actors should be addressable

**Approach:**
- Communication connections are explicitly set up by a privileged and trusted actor
  - *Information flows must be set up by trusted entities.* Communication endpoints are addressed in a manner that is independent of the underlying node’s identity.
  - *Allows the actors to be relocated at runtime without disrupting the application level logic.*
- Actors are given the endpoints of connections at initialization time
  - *Actors cannot simply create endpoints, they must be given to them by a trusted entity*
- Actors supply security labels when communicating
  - *Actors can handle multiple communications simultaneously on different labels, hence they must inform the kernel which one is needed*
- F6OS verifies labels against labels independently supplied in actor’s meta-data
  - *Trusted entity verifies label matching (or compatibility)*
All inter-actor communication happens via endpoints that are logically connected via flows.

Endpoints …
- Are like sockets (connect an actor to another endpoint – of another actor)
- But can be created by only privileged actor(s)

Flow….  
- Associations between one source and one or more destination endpoints  
- Special case: destination endpoint can be multicast group  
- Configured by only privileged actor(s)

Endpoint/flow configuration is part of the ‘Deployment Plan’

Labels
- Generated and assigned by a trusted party (system integrator)
  - There has to be one authoritative source of labels
- Both actors and endpoints have labels
  - Multi-label actors communicating on multi-label endpoints are possible
- Write Equal/Read Equal and Write Equal/Read Down are supported
Solution: Middleware

- The ‘middleware layer’ that provides:
  - Synchronous and asynchronous point-to-point communication with call/response semantics (→ Subset of CORBA RMI)
    - Location transparency
    - Request (de)multiplexing
    - Message (de)marshalling
    - Error handling
    - Support for QoS (client timeouts, reliable one-ways)
  - Anonymous publish/subscribe communications with one/many-to-many data distribution patterns (→ Subset of DDS)
    - Datatype specification
    - Static discovery

CORBA and DDS are complex standards; certification as part of the Trusted Computing Base (TCB) would be prohibitive
Implications for DDS

- DDS Implementation must be label aware
  - Must use secure transport APIs to transmit data samples
  - Must associate labels with DDS entities
- Restricts Quality of Service properties that may be supported
  - Low label actors can not be aware of high label actors
  - Any QoS that requires such knowledge can not be supported
- Complicates conventional DDS discovery process
  - Participants can not spontaneously connect to peers for discovery
  - Unrestricted discovery creates a significant covert channel
Security Label Associations (1/2)

- **Transport Object**
  - Interacts with a F6ST Endpoint
  - May have multiple labels; subset of endpoint labels

- **Writer**
  - Assigned a single label at creation
  - Label may not change, and it applied to all samples

- **Reader**
  - Unlabeled
  - Each sample’s label is communicated via Sample Info mechanism
Security Label Associations (2/2)

- Topics and domains may have multiple labels
- **Write-Equal/Read-Equal**
  - Publishers/Subscribers are the same label
  - Publishers and Subscribers may mutually “know” each other
  - Allows for both unicast and multicast communication
  - Nearly full range of DDS QoS supported
- **Write-Equal/Read-Down**
  - Publishers have a lower label than subscribers
  - Publishers *may not* have any knowledge of subscribers
  - Only allows multicast communication
  - QoS requiring such knowledge is forbidden
    - Reliability
    - Ownership
F6 Discovery Participants

- **Actor Home**
  - Generic process that hosts application business logic hosted in components
  - Hosts the DDS implementation

- **Deployment Manager**
  - Trusted deployment infrastructure local to the node on which an Actor runs
  - Responsible for configuring Secure Transport
  - Part of the TCB

- **Cluster Dictionary Manager**
  - Trusted cluster repository of DDS entities and topics
  - Responsible for matching publishers and subscribers
  - Part of the TCB
Process is similar for subscriber entity
Proxy objects provide a trusted mediator between discovery service and untrusted actor
Discovery services is label aware
Only matches entities of compatible labels
Proxy objects intercept discovery events and configure Secure Transport before informing entity of matches
F6 Discovery generates “fake” meta data (GUID, etc) when matching low writers to high readers
Lessons Learned

- DDS is a useful and popular platform for future space systems
  - Most submitted designs for the System F6 IAP featured DDS as a communication mechanism
  - Shown to be an effective tool to write distributed flight software for fractionated spacecraft
- Segregating Discovery substantially simplifies integrating DDS in MLS systems
  - DDS need not be part of the TCB, substantially simplifying analysis and design
  - Reduces possibility of covert channels between low and high actors
- Proprietary F6 discovery process could present a interoperability concern

More Information:
http://www.isis.vanderbilt.edu/projects/F6