



MILS

*Architecture Enabling
High Assurance Certifications*

Multiple Independent Levels of Safety/Security

*Architecture Enabling
High Assurance Certifications*

Gordon M. Uchenick

Senior Mentor / Principal Engineer

410-256-7102

gordon.uchenick@ois.com



Agenda

MILS

*Architecture Enabling
High Assurance Certifications*

- What is High Assurance?
- Fail-first, Patch-later
- MILS Architecture
- Guest Operating System Architecture
- Distributed Security Requirements
- High Assurance MILS Workstation



What is High Assurance?

MILS

Architecture Enabling
High Assurance Certifications

- For Airborne Software:
 - One failure per 10^9 (1 Billion) hours of operation
 - How long *is* a Billion hours? Do the math!
 - $1,000,000,000 \text{ hours} \times \frac{1 \text{ day}}{24 \text{ hours}} \times \frac{1 \text{ year}}{365.25 \text{ days}}$
 - **114,077 YEARS!**
- For National Security Systems processing our most valuable data under most severe threat:
 - Failure is *Unthinkable*
- ***How do we implement systems that we can trust to be this robust?***



- *RTCA DO-178B, Software Considerations in Airborne Systems and Equipment Certification*
- *ARINC-653, Avionics Application Software Standard Interface*
- *ISO-15408, Common Criteria for Information Technology Security Evaluation*
- *DCID 6/3, Protecting Sensitive Compartmented Information Within Information Systems*



Assurance Certification Goals

MILS

*Architecture Enabling
High Assurance Certifications*

<i>Common Criteria</i> Basic Robustness (EAL3) Medium Robustness (EAL4+) High Robustness (EAL6+)	<i>MSLS / MLS Separation Accreditation</i> System High Closed Environment System High Open Environment Multi Level Separation
<i>DCID 6/3 Protection Level 5</i>	<i>Multi Nation Coalition Separation Accreditation</i>
<i>DO-178B Level A</i>	<i>Failure is Catastrophic</i>



Fail-first, Patch-later

MILS

*Architecture Enabling
High Assurance Certifications*

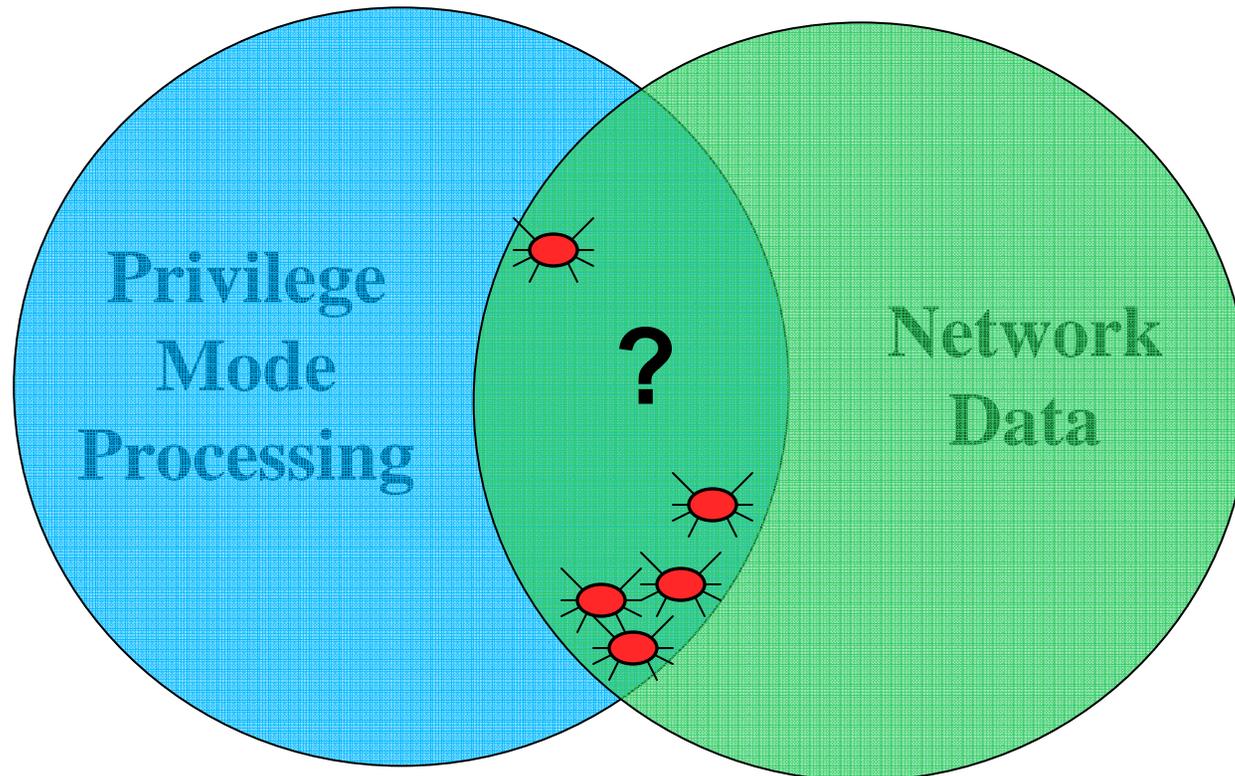
- Most commercial computer security architectures
 - The result of systems software where security was an afterthought
 - Operating systems
 - Communications architectures
 - **Reactive** response to problems
 - Viruses, Worms, and Trojan Horses
 - Hackers and Attackers
 - Problems are only addressed **after** the damage has been done

- Inappropriate approach for mission critical systems
 - Does not safeguard information or the warfighter
 - **Proactive** measures are required to **prevent** damage

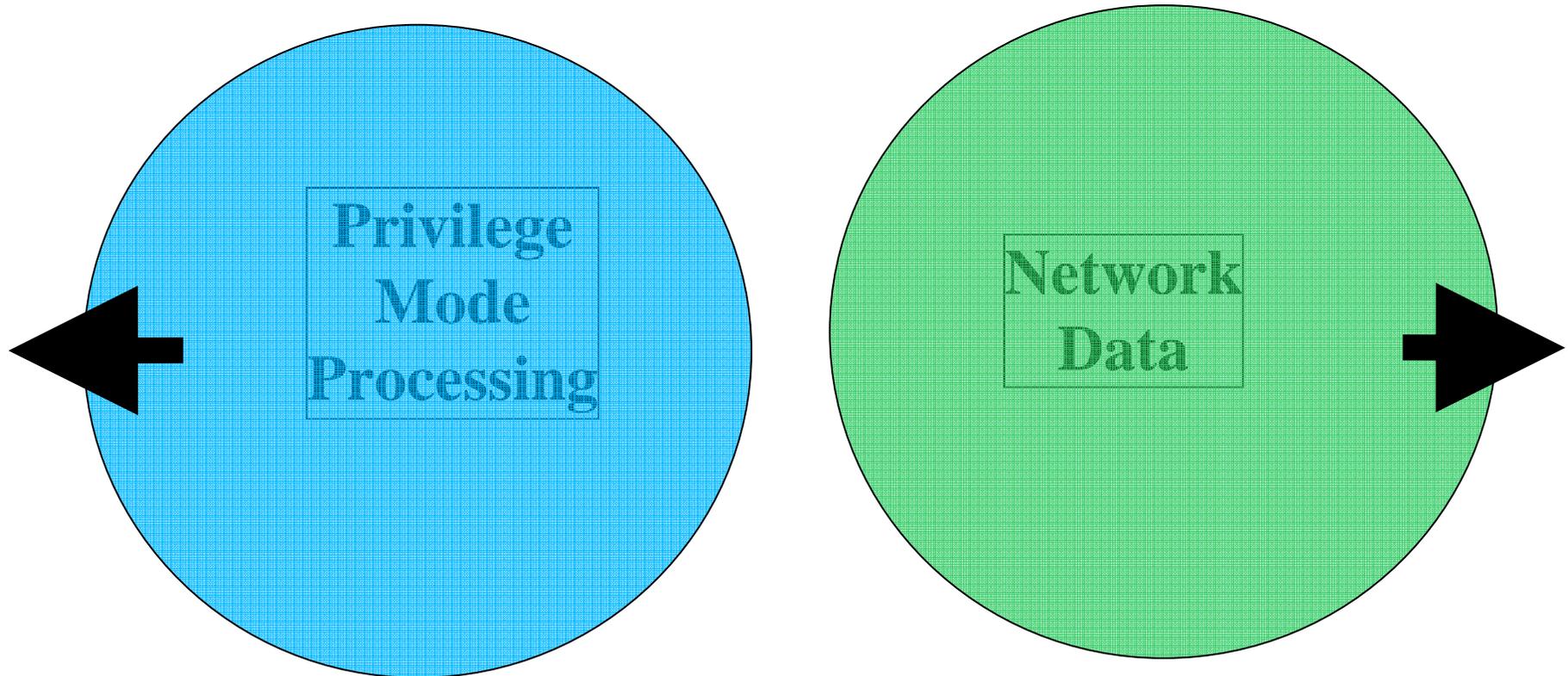


*Breeding Ground for Internet
Wildlife*

MILS
*Architecture Enabling
High Assurance Certifications*



Wild Creatures of the Net: Worms, Virus, . . .



**Under MILS, network header and
privilege mode processing are separated**



The Whole Point of MILS

MILS

*Architecture Enabling
High Assurance Certifications*

Really very simple:

- Dramatically **reduce the amount** of *safety/security critical code*

So that we can

- Dramatically **increase the scrutiny** of *safety/security critical code*



Three distinct layers (John Rushby, PhD)

- **Separation Kernel**
 - Separate process spaces (partitions)
 - Secure transfer of control between partitions
 - Really small: 4K lines of code
- **Middleware**
 - Application component creation
 - Provides secure end-to-end inter-object message flow
 - Device Drivers, File Systems, Network Stacks, CORBA, DDS
- **Applications**
 - Implement application-specific security functions
 - Firewalls, Cryptomod, Guards, Mapplet Engine, CDS, Multi-Nation Web Server, etc.



Separation Kernel

- **Microprocessor Based**
 - Multi-Core Time and Space
Multi-Threaded Partitioning
 - Data Isolation
 - Inter-partition Communication
 - Periods Processing
 - Resource Sanitization
 - Minimum Interrupt Servicing
 - Semaphores
 - Multi-Core Synchronization
Primitives
 - Timers

And nothing else!

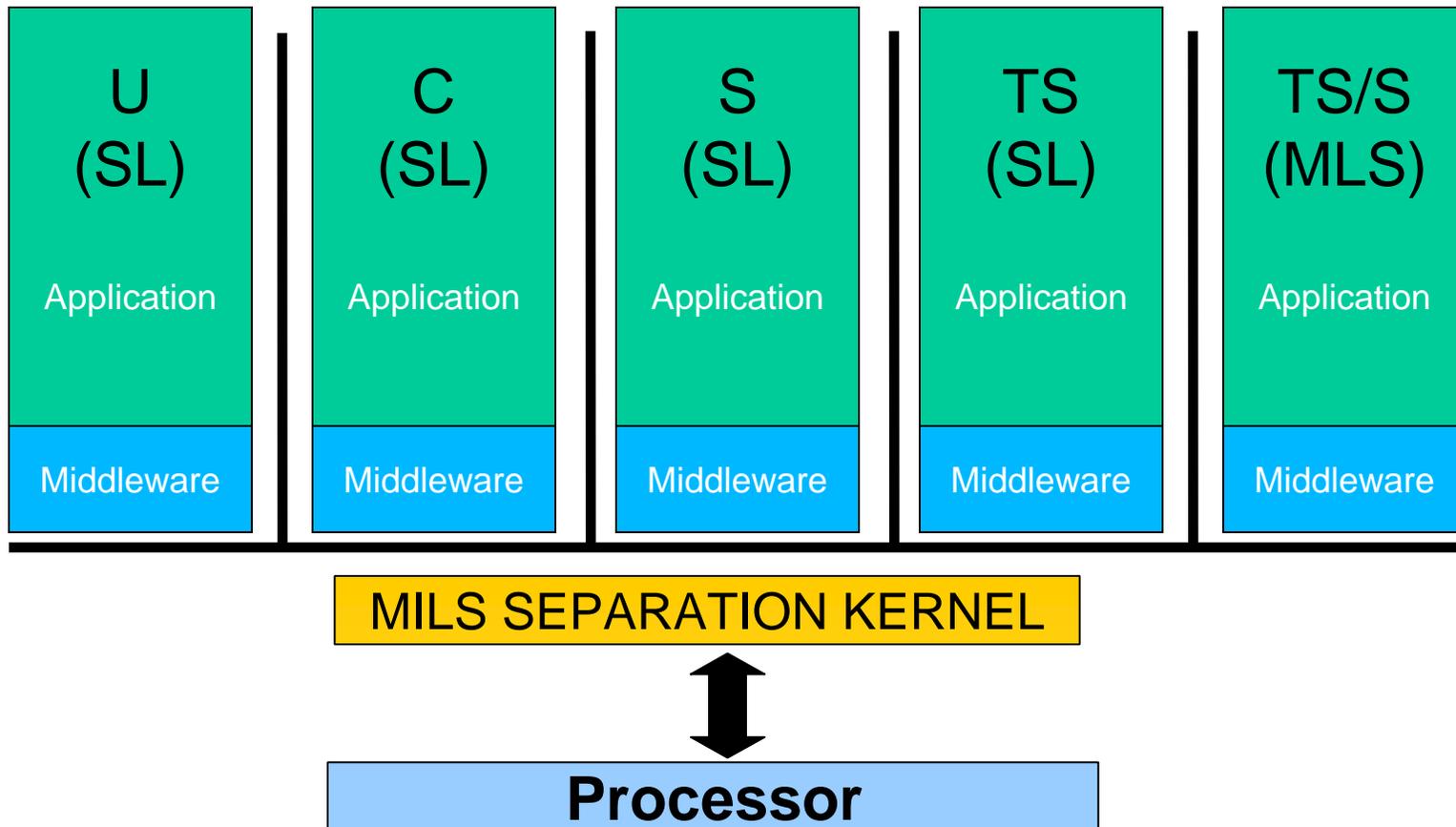
MILS Middleware

- **Traditional RTOS Services**
 - Device Drivers
 - File Systems
 - Token and Trusted Path
- **Traditional Middleware**
 - CORBA (Distributed Objects)
 - Data Distribution (Pub-Sub)
 - Web Services
- **Partitioning Communication System (PCS)**
 - Global Enclave Partition Comm
 - TCP, UDP, Rapid-IO, Firewire,
...
 - Partition Based Attestation



The MILS Architecture

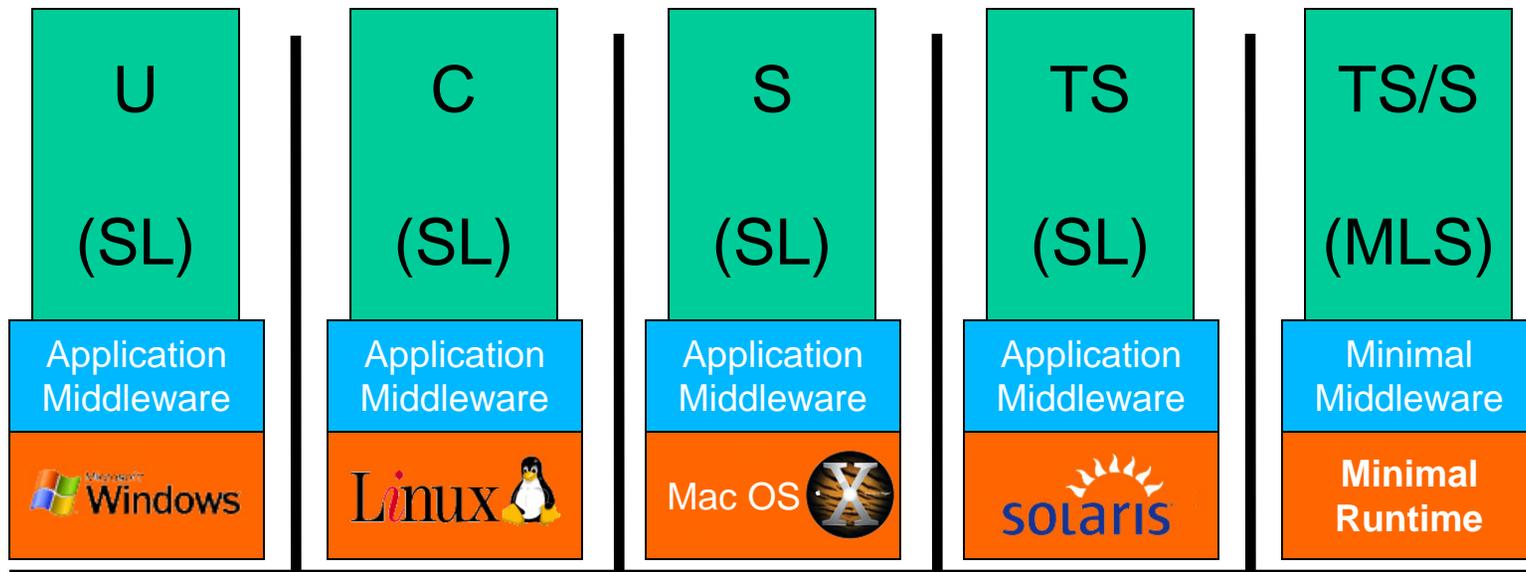
MILS
*Architecture Enabling
High Assurance Certifications*





Guest OS Architecture

MILS
Architecture Enabling
High Assurance Certifications



A MILS Workstation? (later...)



Processor



Why Does Neatness Count?

MILS

*Architecture Enabling
High Assurance Certifications*

**N
E
A
T**

Safety and Security enforcing functions must be:

- **N**on-bypassable
 - Enforcing functions cannot be circumvented
- **E**valuatable
 - Enforcing functions are small enough and simple enough for mathematical verification
- **A**lways Invoked
 - Enforcing functions are invoked each and every time
- **T**amperproof
 - Subversive code cannot alter the enforcing data or functions



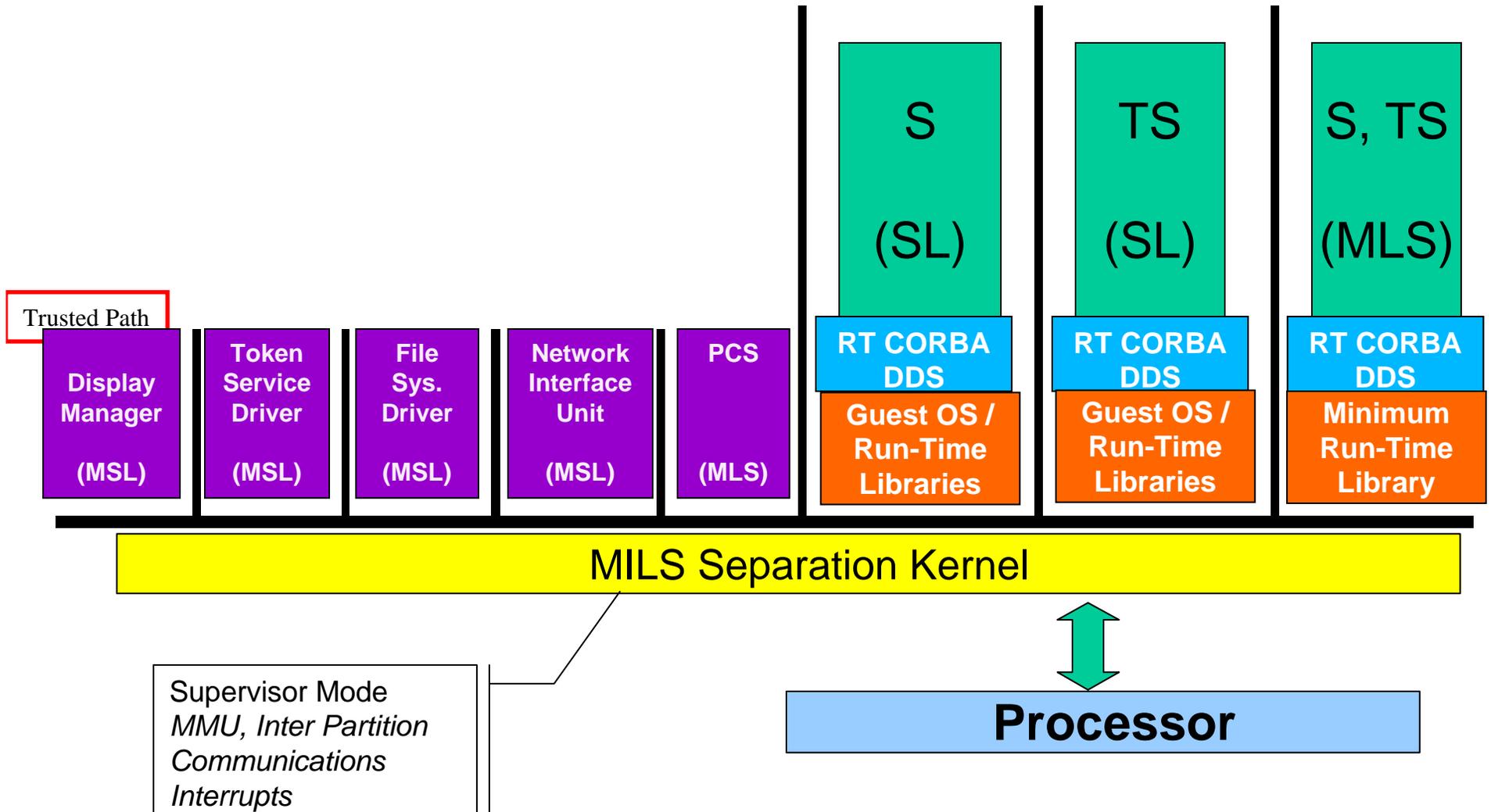
- Extend single node enforcement to multiple nodes
- Do not add new threats to data Confidentiality or Integrity
- Enable distributed Reference Monitors to be **NEAT**
- Optimal inter-node communication
 - Minimizing added latency (first byte)
 - Minimizing bandwidth reduction (per byte)
- Fault tolerance
 - Infrastructure must have no single point of failure
 - Infrastructure must support fault tolerant applications



High Assurance MILS Workstation

MILS
Architecture Enabling
High Assurance Certifications

Application (User Mode) Partitions

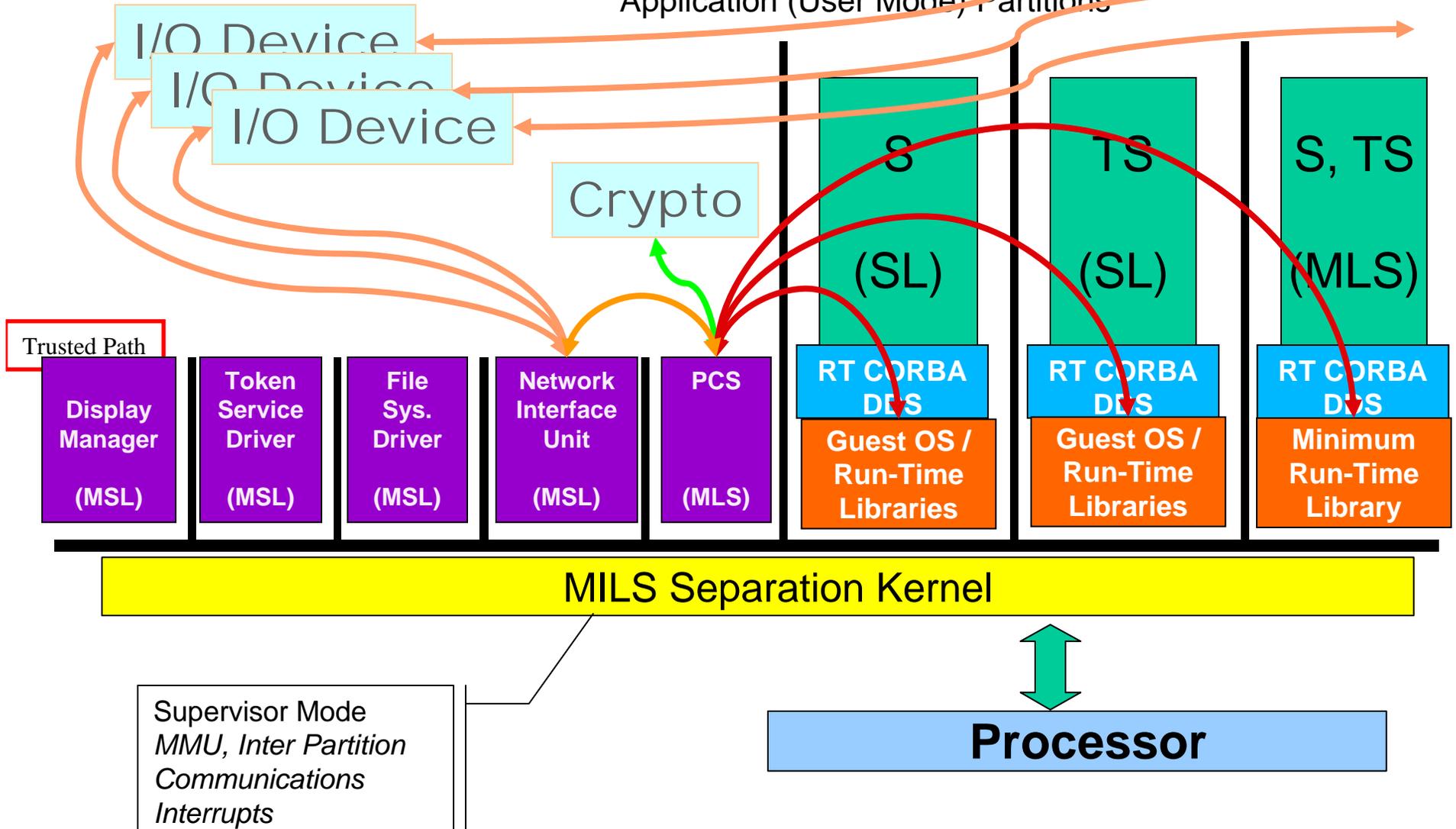




MILS Workstation Network Access

MILS
Architecture Enabling
High Assurance Certifications

Application (User Mode) Partitions





Really very simple:

- Dramatically **reduce the amount of safety/security critical code**

So that we can

- Dramatically **increase the scrutiny of safety/security critical code**

To make

- Development, certification, and accreditation more **practical, achievable, and affordable.**