Multiple Independent Levels of Security

for assisting the creation of systems that are resistant to attackers
Safety Critical Systems

What
A system whose failure may result in one or more of the following:
- death or serious injury
- loss or severe damage to property or equipment
- environmental harm

Examples
- Life support systems: space, underwater, ventilators …
- Robotic surgery machines: surgeon does not have to be there!
- Nuclear reactor control systems: 3 Mile Island, Chernobyl
- Amusement rides: Schlitterbahn Waterpark, Kansas City
- Battery management for hybrid vehicles
- Drive by wire: human gestures to computers that control car
- Fly by wire: human gestures
- Air traffic control systems
- Sewage treatment
- Water supply
- Electric power grid
Mathematical evidence (theorem prover) that a system will function exactly as intended at all times

The size and complexity associated with software that monitors, controls, and protects flight critical products continues to grow. This is compounded by an increased use of autonomous systems which are just as complex, if not more so, since many operator responsibilities are supported and replaced by software in unmanned systems.

Further, these systems are subject to cyber-enabled attacks, thereby necessitating another level of complex software to ensure security. GE Research devoted a team to research and develop a new suite of tools to address the challenges with design, development, and verification of these software-intensive products.

The goals were to develop technology, processes, and tools that result in more efficient software and system development as measured by cost and cycle time, and to enable new capabilities such as autonomy and the Industrial Internet.

--- from GE research

High Assurance Systems
Multiple Independent Levels of Security

**What**

high-assurance security architecture based on the concepts of separation and controlled information flow – for safety-critical systems and multi-level data communications

**Diagram:**

- **Enclave 1**
- **Enclave 2**
- **Enclave 3**

MILS architecture transforms data from one classification level to another

Separate data streams (no communication between streams)

Security policies protect classified data

Independent, secure enclaves, different classification levels
Multiple Independent Levels of Security

What

high-assurance security architecture based on the concepts of separation and controlled information flow – for safety-critical systems
implemented by separation mechanisms that support both untrusted and trustworthy components
ensures security cannot be bypassed by an alternate communication path
ensures a system is tamperproof
    unauthorized changes to configuration & data is prevented
ensures a system can be evaluated
    requires: modular components, well specified, compact, simple components, formally provable properties
is always invoked
    every access and message is checked by an appropriate security monitor
Multiple Independent Levels of Security

What

high-assurance security architecture based on the concepts of separation and controlled information flow – for safety-critical systems

implemented by separation mechanisms that support both untrusted and trustworthy components

ensures security cannot be bypassed by an alternate communication path

ensures a system is tamperproof
   unauthorized changes to configuration & data is prevented

ensures a system can be evaluated
   requires: modular components, well specified, compact, simple components, formally provable properties

is always invoked
   every access and message is checked by an appropriate security monitor

employs one or more separation mechanisms: separation kernel, partitioning communication system, physical separation
Multiple Independent Levels of Security

How

MILS is a layered approach with lower layers providing security services to higher layers.

Each layer is responsible for security services in its own domain and nothing else.

The layered approach limits the complexity and scope of security mechanisms so evaluation becomes possible.

Applications – MLS and Non-MLS

Middleware Services
(Device drivers, File Systems, Network communications)

Separation Kernel

Hardware
(MMU, Interrupts)

sep kern → partitioning, scheduling, and secure comm between partitions
Multiple Independent Levels of Security

How
Multiple Independent Levels of Security

Supports Foundational Security Policies:

End-to-end Information Flow
- Policy for checking integrity of data moving from one component to another
- Policy for authorization of movement of information

End-to-end Data Isolation
- Policy says how transparent data is (to other users/processes)
- Data are accurate and coming from official/trusted source
- Has policies for partial disclosure of information (e.g. headers)
- Tradeoff: many-user access means higher concurrency and worse performance

End-to-end Periods Processing
- While sensitive information is being processed, all other applications and data use are prohibited
- After processing, the memory must be sanitized to remove crypto variables and so on

End-to-end Damage Limitation
- Application error damage does not propagate to other partitions
Multiple Independent Levels of Security

What

supports enforcement of one or more application/system specific security policies by authorizing information flow only between components in the same security domain or through trustworthy security monitors which analyze data looking for suspicious behavior or unauthorized system changes

MILS architecture allows for execution of multiple applications at potentially multiple security levels or classifications

Each is protected from others and each may communicate with the others based on mechanisms that support policy enforcement

The old way to get separation was to have physically separate computers, networks, and displays – not practical

The new way to get separation allows enclaves of different classification levels to run on the same processor, even
Multiple Independent Levels of Security

Importance
Military needs systems that are very highly safe and secure

MILS architectures can be evaluated according to the Common Criteria

The US military requires evaluation to high security standards

COTS components that have a very high evaluation are desirable as they can save plenty of money in design and certification costs

Major application: military jets
Imagine: a squadron of planes is suddenly disabled in the air due to enemy intrusion

F-35 Joint Strike Fighter Communications, Navigation, Identification (CNI) system uses a MILS architecture

Major application: control of nuclear power generation

Major application: control of sewage treatment systems
Multiple Independent Levels of Security

Assured Data and Process Separation

Separation kernel –
provides multi-level secure operation on general purpose multi-user systems.

Middleware services -
traditional OS functions are taken from the kernel and put in middleware to make the separation kernel small and evaluable

Partitioning Communication System -
extends MILS software security policies to the network: end-to-end information flow, data isolation, periods processing, damage limitation.

Physical Separation -
Multiple Independent Levels of Security

Separation Kernel

**Purpose:** provide multi-level security on general purpose multi-user systems
Multiple Independent Levels of Security

Separation Kernel

**Purpose:** provide multi-level security on general purpose multi-user systems

Creates an environment which is indistinguishable from that of a distributed physical system
Multiple Independent Levels of Security

Separation Kernel

**Purpose:** provide multi-level security on general purpose multi-user systems

Creates an environment which is indistinguishable from that of a distributed physical system

It must appear as if each enclave is a separate, isolated machine and that information can only flow from one machine to another along known external communication lines
Multiple Independent Levels of Security

Separation Kernel

**Purpose:** provide multi-level security on general purpose multi-user systems

Creates an environment which is indistinguishable from that of a distributed physical system

It must appear as if each enclave is a separate, isolated machine and that information can only flow from one machine to another along known external communication lines

It must be proved that there are no channels for information flow between enclaves other than those explicitly provided
Multiple Independent Levels of Security

Separation Kernel

**Purpose:** provide multi-level security on general purpose multi-user systems

Creates an environment which is indistinguishable from that of a distributed physical system

It must appear as if each enclave is a separate, isolated machine and that information can only flow from one machine to another along known external communication lines

It must be proved that there are no channels for information flow between enclaves other than those explicitly provided

- Data isolation ensures an enclave can't access resources in other enclaves
- Periods processing ensures applications within enclaves execute for the specified duration in the system schedule
- Information flow defines permitted info flows between enclaves
- Fault isolation ensures a failure in one enclave does not impact any other enclave within the system
Multiple Independent Levels of Security

Separation Kernel

Separation Kernel Protection Profile:
High assurance systems require proof that system meets critical safety and security requirements

Protection profile provides a formal notion of system architecture and data flows that can be subjected to formal analysis (theorem provers)

The following can be proved formally from PP:
Protection of all resources from unauthorized access

Separation of internal resources used by (target of evaluation) functions from exported resources made available to subjects

Isolation and partitioning of exported resources

Correct mediation of information flows between partitions and between exported resources

Correct auditing procedures
Multiple Independent Levels of Security

Common Criteria

Original version:
Certification of single products such as processors, operating systems, applications
Adapt the protection profile to the product at a given EAL
Labs or NSA evaluates

Later version:
Allows certification of composed products
Two or more evaluated products can be combined
base component + dependent component
Composition class:
composition rationale
development evidence
reliance of dependent component
base component testing
composition vulnerability analysis

The products may be from different organizations
Multiple Independent Levels of Security

Common Criteria

Later version:

Ensure base component provides at least as high an assurance level as the dependent component

Security functionality in support of security requirements of dependent component is adequate

Description of interfaces used to support security functions of dependent component is provided

Testing of base component as used in composed TOE is performed

Residual vulnerabilities of base component are reported and an analysis of vulnerabilities arising from composition are considered
Multiple Independent Levels of Security

Common Criteria

Later version:
Composition Assurance Packages
Build on results of previously evaluated entities
CAP-A: Structurally composed
  Security functional requirements are analyzed just using the outputs from the evaluations of components

CAP-B: Methodically composed
  Security functional requirements are analyzed using outputs from component evaluations, specification of interfaces and high level component design of the composed system

CAP-C: Methodically composed, tested & reviewed
  CAP-B + involvement of the base component developer
Multiple Independent Levels of Security

Common Criteria

MILS is a good fit for the Common Criteria certification:
MILS was designed as a component architecture
Components are designed by multiple vendors
Components are certified at multiple EAL levels
Components assist with security policy enforcement

Example: Separation kernel & MILS Message Router (MMR):
base: Separation kernel
dependent: MMR

Evaluate Separation Kernel
  PP exists, security target exists, target: EAL 6+
Evaluate MMR
  No PP, artifacts reviewed, target: EAL 5
Evaluate Composed MILS Components
  Define a Security Target for the composed system
  Decide on a Composition Assurance Level (CAP)

If done right, certification results for combined system can be re-used by multiple vendors
Multiple Independent Levels of Security

Separation Kernel

Available from

Green Hills Software  https://www.ghs.com/
  Integrity 178B RTOS used in F-16, F-22, F-35, Airbus 380
  Very tiny kernel – 4K lines
  Kernel is evaluated to NSA EAL 6+ (semi-formally verified)

Lynx Software Technologies  http://www.lynx.com/
  LynxSecure separation kernel and embedded hypervisor
  LynxOS-178 RTOS (LynxOS on Atari 1040ST in 1986-1989)

SYSGO  https://www.sysgo.com/
  PikeOS – small set of privileged services
  Used in products certified by the French NIS Agency

Wind River Systems  https://www.windriver.com/
  VxWorks MILS platform compliant with Separation Kernel Protection Profile (SKPP) from the NSA

OK Labs  https://en.wikipedia.org/wiki/Open_Kernel_Labs
  OKL4 microkernel – in billions of mobile devices
Multiple Independent Levels of Security
Partitioning Communications System (PCS)
A communications security architecture compliant with an information flow separation policy

Extends the MILS architecture to network flows

Works with a separation kernel to ensure
  System security channels cannot be bypassed
  System can be evaluated
  Is always invoked – policies are always checked
  System is tamperproof

Supports (a kind of) formal proof of correctness
Multiple Independent Levels of Security

Formal Proof of Correctness

Introduce and define States of a system in terms of security

Define transition rules from State to State based on various kinds of triggers (e.g. input or clock timer firing)

Check that the initial State is considered secure

For each transition from State A to State B, check that if A is considered secure then B can be considered secure

Then we have a proof that the system is secure
Multiple Independent Levels of Security

Formal Proof of Correctness

Operation:
Triple: (subject, object, operation)
Example: (franco, sshd, execute)

Subjects and Objects labeled with security levels in partial order

But each subject has a current security level and a maximum security level
Thus subjects can be 'downgraded' in security temporarily

Access control matrix (M): gives permissions for a given operation (o) on particular sets of security levels (l)

A State: (o,M,l)
Multiple Independent Levels of Security

Formal Proof of Correctness

Policy types (discretionary and mandatory):

Discretionary: access may be permitted (i.e. (s,o,op))
No read-up: subject may not read object at higher security level
No write-down: subject can't write to object at lower level

Subjects are processes, memory is an object
Subjects have access to memory
Subjects can act as channels by reading one memory object
and writing that information to another memory object

Trusted subjects are exempt from no write-down policy
Subjects can be 'downgraded' in security temporarily to loosen
the mandatory restrictions

A State is secure if all current access triples (s,o,op) are
permitted by the policies above

A State transition is secure if it is between two secure States
If the initial State is secure and all transitions are secure then
the system is secure
Multiple Independent Levels of Security

Formal Proof of Correctness

Operations for a real-time OS:

- **Execute:**
- **Read:**
- **Write:**
- **Read and write:**
- **Get-read:** requests read access to an object
- **Release-read:** release an object
- **Give-read:** grant read access to another process
- **Rescind-read:** withdraw read permission given to another process
- **Create-object:** OS has to check write access on the object directory is permitted and the security level of the object dominates the security level of the process
- **Change-subject-current-security-level:**
- **Change-object-current-security-level:**