Secure Computing Workshop:
Group 2 Brief-out
Scenario – Sensor Fusion

- Controlled sharing
- Key component - MLS combat device
- SCI to unclassified
- A1 criteria
  - Formal policy model - MLS
  - Formal specification
  - Build system that is reliable
  - Hardware mechanism – information hiding and layering
  - Configuration Management of tools
  - Assumption – design is right (kernels, hardware)
Threats

- **Subversion of the infrastructure**
  - Computers – any software or hardware that is part of the infrastructure
    - E.g. life cycle attacks, creation of hardware, distribution of software
  - Separation of concerns
    - Hardware
    - Low level software
    - Firmware

- **Operational Attacks**
  - Active and passive communication attacks
    - Direct attacks – information leakage
      - Wire taps,
      - Communication interference – denial of service
  - Information leakage
  - Jamming communication

- **Integrity attacks**

- **Overrun**
  - Physically taking the combat device and masquerading as friendly device
  - How do we detect this?
  - Reverse engineering – software and hardware
    - Is it possible and non-fatal? How?

- **Design/device loss/theft**
Hard Problems

- Secure hardware abstraction layer
- Trusted Multilevel display interface
- Assured Tool Chain for Trusted Development and Management of HW and low level SW

- Key management – secure (re)generation and storage of keys
- Computing through encryption
- What trusted components do we need so we can build secure devices with a mix of trusted and untrusted components?

- MLS SAN support
- Trusted Dynamic Reconfiguration for fault tolerance and audit
Secure Abstraction Layer

- **Problem:** Define a HW/FW interface that is rich enough to support “full featured” secure operating systems

- **Property:** Software can only access hardware through the abstraction layer
- **Set of primitives to build secure software on the hardware**
- **Need for abstraction layer between hardware components**
Secure Abstraction Layer: Research Status

- Past experiences with evaluating secure systems have identified candidates

- Related Work
  - Phoenix Security Services
  - Intel
  - Alpha PAL code
  - TPM

- All the above are insufficient to support layering, information hiding, MLS services e.g. memory management, support for sharing
Secure Abstraction Layer: Measuring Effectiveness

- Build A1 requirements
- Support for full OS functionality
- Performance – speed, energy, power, etc.
Tool Chains

- **Need:** Assured tool chains for developing trusted systems

- **Example tool chain:**
  - SW: compilers, loaders
  - Hardware: HDL synthesis tools – behavioral, logic, physical design

- **Problem:** How should tool chains be designed/implemented so that they can be certified and their products (HW and SW) are trustworthy

- **Hardest Problem:** Assured tools that work together
Tool Chains

- **Current work**
  - Hardware: none?
  - Software
    - Loaders that encrypt
    - Runtime checks, e.g. stack guard

- **Why Now**
  - Reach
  - Convert design community

- **Measures**
  - Use of tools and concepts – related to performance and acceptance
  - Verification tools
MLS Window Support

Problem: Provide support for window system that
- Provides highly intuitive, visible labeling for windows
- Secure cut and paste
- Presents a standard interface for commercial software

Research Status
- Compartmented Mode Workstation - not a success
- Trusted X - never finished, but a promising approach
- Padilla & Mayer (sp?) - identified useful HW features
- Karger - suggested VMM for the display

Measures of Success
- Support for commercial software
- Some usability metric (Marine test)
- Bandwidth of remaining covert channels