RSAConference2015

San Francisco | April 20-24 | Moscone Center

CHANGE

Challenge today's security thinking

SESSION ID: MASH-F03

What Trusted Computing History Teaches Us About Today's Challenges

Robert Bigman

President
2BSecure
@rybbigs

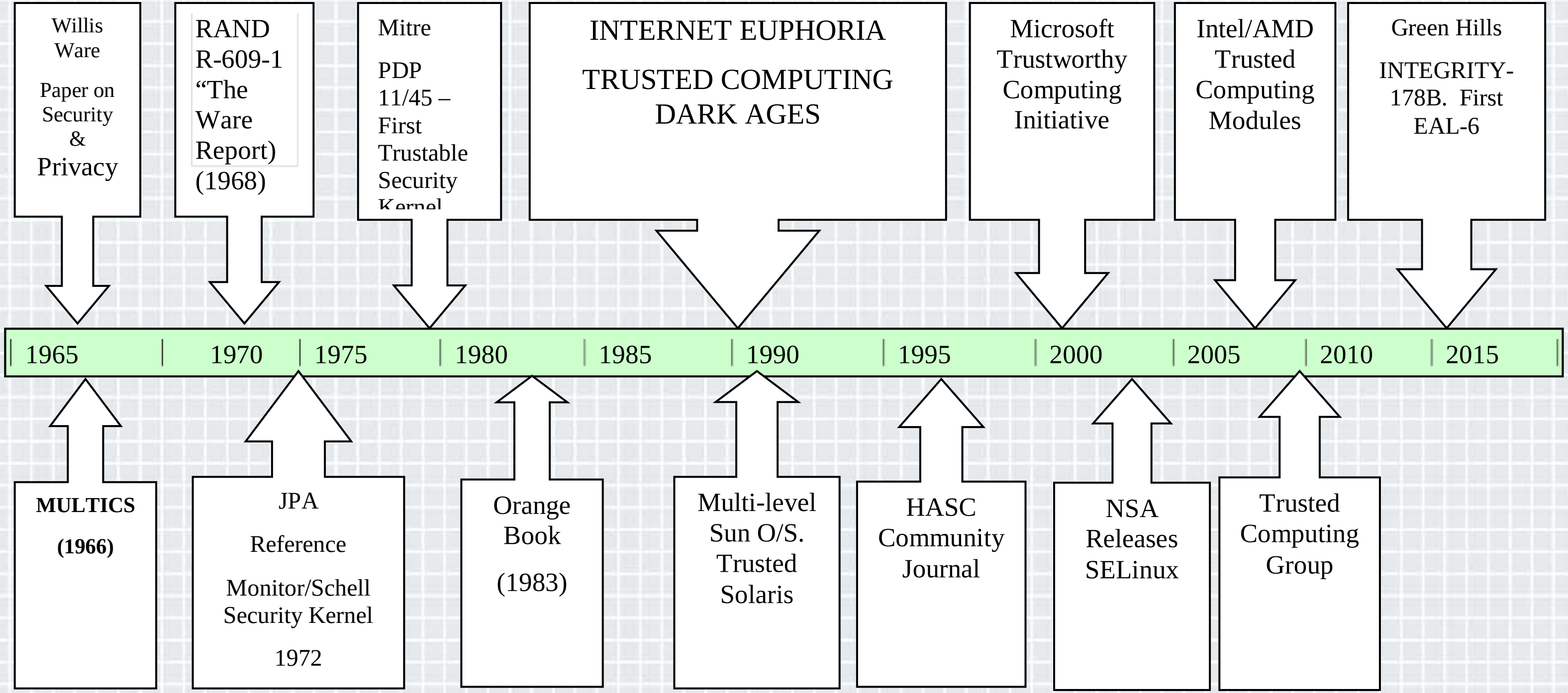
Why Study Trusted Computing History?

- **Because:**
  - Critical security issues identified as early as 1964 have still not been resolved in 2015.
  - Patching, layered firewalls and, now, cyber intelligence is not working (and will not work).
  - Early cyber security pioneers identified *essential* elements for building trusted systems (and actually built some)
  - Today's global IT fabric (think IOT) contain very few truly secure processors.
  - George Santayana was right

From an ACM Presentation in the 1960s

- “*Security is inherently different from other aspects of computing due to the presence of an adversary. As a result, identifying and addressing security vulnerabilities requires a different mindset from traditional engineering. Proper security engineering—or the lack of it!—affects everything . . . .”*

Trusted Computing Timeline



Willis Ware

Paper on Security & Privacy

RAND R-609-1 "The Ware Report) (1968)

Mitre

PDP 11/45 – First Trustable Security Kernel

INTERNET EUPHORIA

TRUSTED COMPUTING DARK AGES

Microsoft Trustworthy Computing Initiative

Intel/AMD Trusted Computing Modules

Green Hills

INTEGRITY-178B. First EAL-6

1965	1970	1975	1980	1985	1990	1995	2000	2005	2010	2015

MULTICS

(1966)

JPA

Reference

Monitor/Schell Security Kernel

1972

Orange Book

(1983)

Multi-level Sun O/S. Trusted Solaris

HASC Community Journal

NSA Releases SELinux

Trusted Computing Group

2BSecure LLC

What Do You Mean: "Trusted Computing"

- ***From Wikipedia***: A system that is relied upon to a specified extent to enforce a specified security policy.
- **A System:**
  - Built to resist **Subversion**
  - Where **Trust** can be **Attested** and **Continuously Proven**
  - That possesses a **Small** and **Verifiable Reference Monitor**
  - That can **Securely Detect** and **Report** subversion
  - That enforces a **Mandatory Access Control** policy
  - Programmed in a **Highly Typed** language
  - With a **Trusted** Supply/Update Channel

Willis Ware - Security And Privacy - 1960s

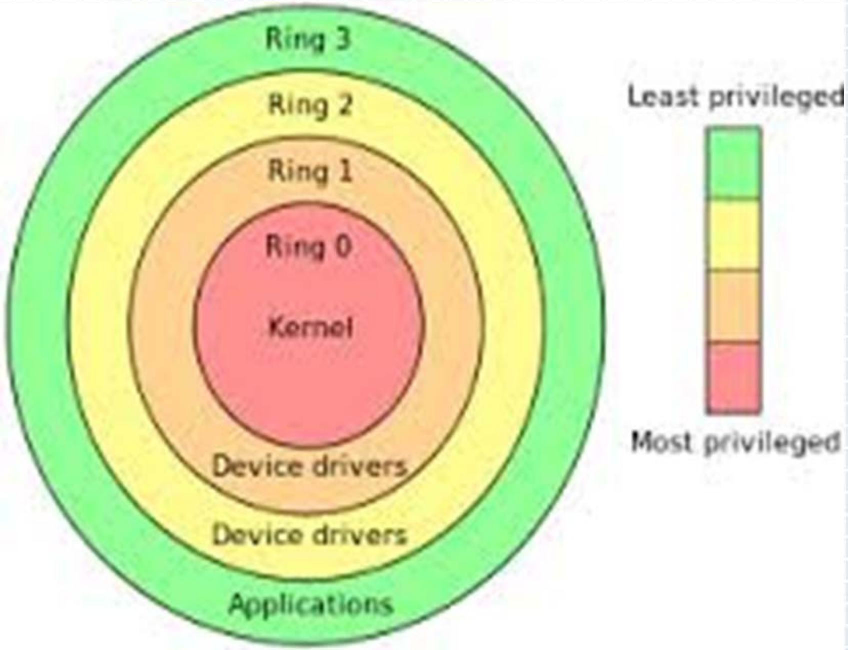
- Protection of central and demountable storage media
- Protection for circuits
- Safeguards for timesharing systems
- Software safeguards to protect access to files
- Software monitoring of users access to files
- Safeguards to protect against software modification
- **Trusting the operating system**
- Safeguards to protect personal data
- Administrative and management controls

MULTICS - 1966

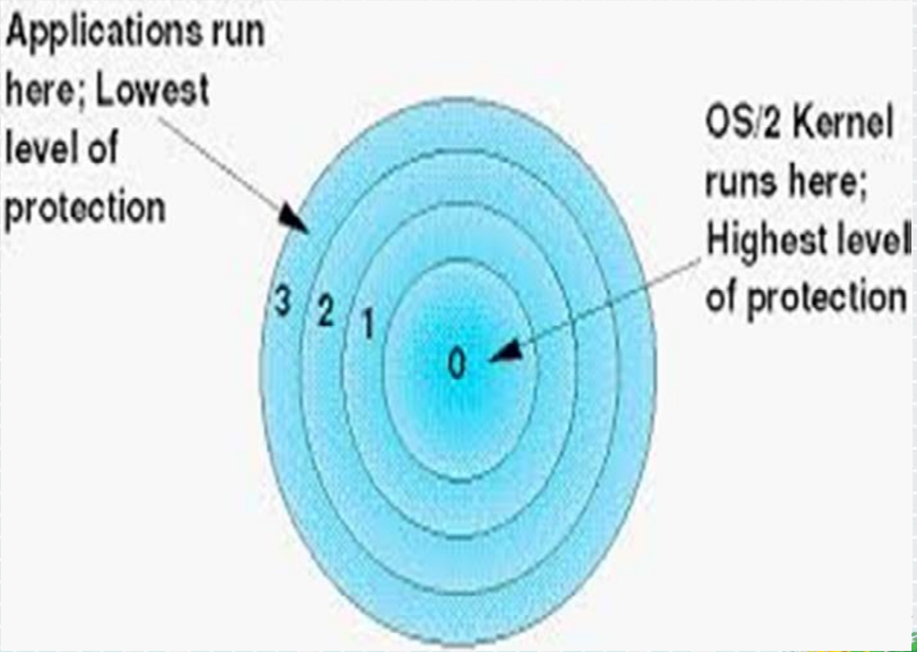
- Many concepts found today in Unix/Linux releases (just not security)
- First time-sharing system built with a security model
- First system built with a Mandatory Access Control (MAC) policy
- USAF upgrades led to TCSEC use-case for B2 systems
- Programmed in PL/1 (highly typed)
- Apps. had to satisfy security model not vice versa
- Hardware segregated ring oriented architecture (Honeywell 6180)
- Ring 0 is 628K
- Used by both government and industry to securely share data.

Coincidence – I Don’t Think So!

MULTICS



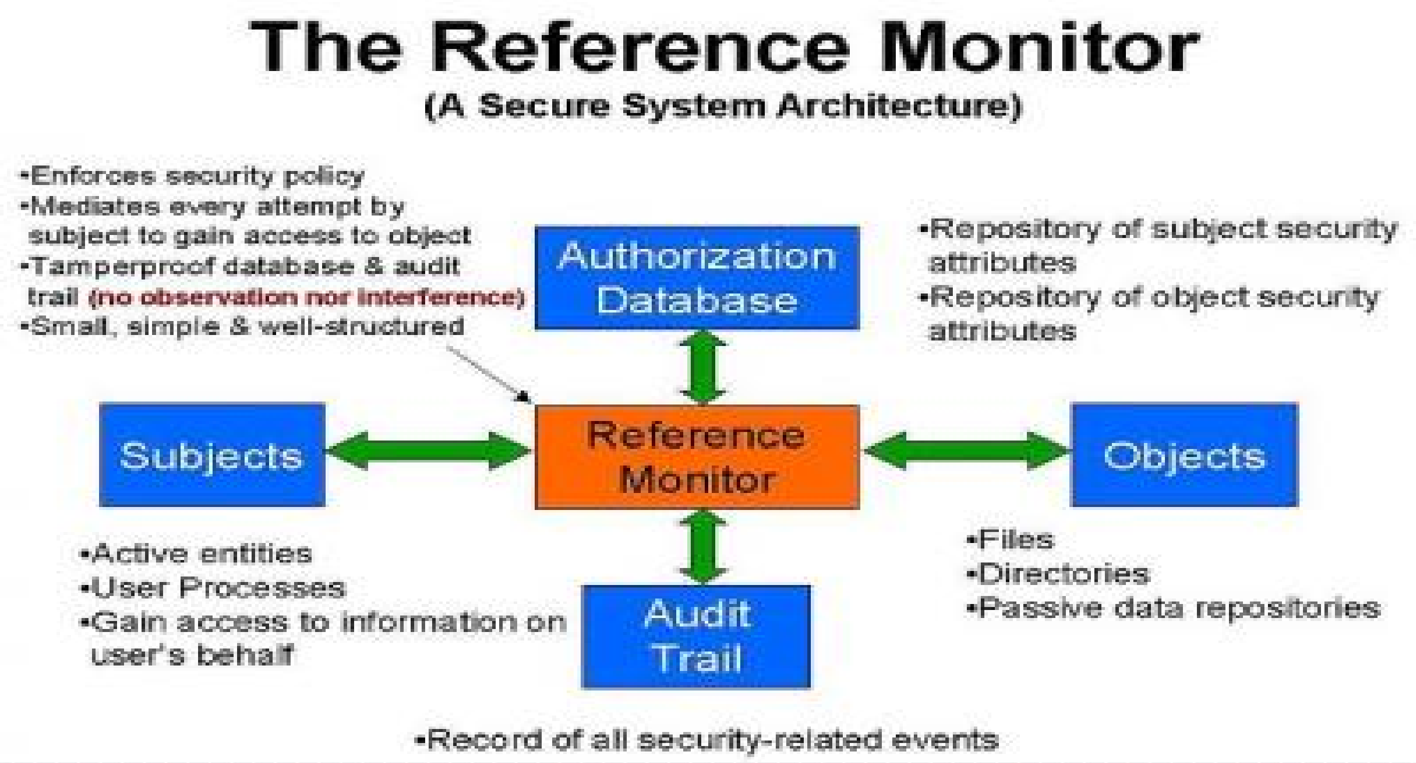
OS/2


Applications run here; Lowest level of protection
OS/2 Kernel runs here; Highest level of protection
3
2
1
0

The Concept Of Trusted Computing – 1970s

- James P. Anderson’s Computer Security Technology Planning Study and the reference monitor
- Roger Schell and the security kernel (e.g., Project Guardian):
  - Complete mediation
  - Tamperproof
  - Verifiable
- The security kernel in action (Mitre’s DEC PDP 11/45)
- The hypervisor as a kernel/reference monitor (UCLA’s IBM’s VM 370)

The Concept Of Trusted Computing – 1970s



The Reference Monitor

(A Secure System Architecture)

- Enforces security policy
- Mediates every attempt by subject to gain access to object
- Tamperproof database & audit trail (no observation nor interference)
- Small, simple & well-structured

- Repository of subject security attributes
- Repository of object security attributes

- Active entities
- User Processes
- Gain access to information on user's behalf

- Files
- Directories
- Passive data repositories

- Record of all security-related events

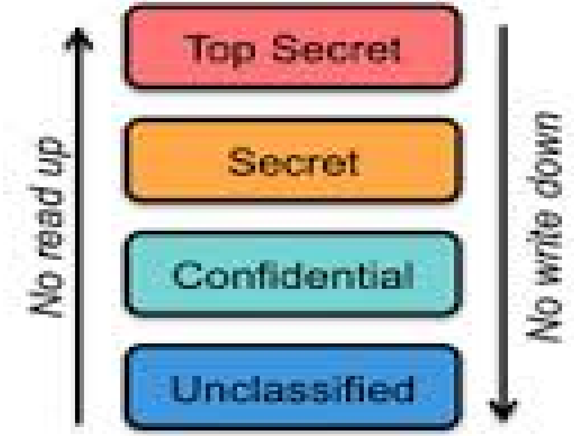
2BSecure LLC

We're From The Government And We're Here To Help You – 1980s

- 1983 - Trusted Computer System Evaluation Criteria (TCSEC) – aka "Orange Book:"
  - Implemented Bell-Lapadula security model
  - Confidentiality was paramount
  - Enforces both mandatory/discretionary access restrictions
  - Required accountability (identity, authentication, audit)
  - Required assurance (operational, life-cycle, continuous)
  - Required lots of documentation
- Divisions and classes (D, C1, C2, B1, B2, B3, A1)

We're From The Government And We're Here To Help You – 1980s

- **Bell Lapadula Security Model**



Confidential cannot read Secret
Confidential cannot write Unclassified

From R&D To Implementations – 1990s

- Microsoft Windows NT 4.0 (C2+)
  - DAC; object reuse; accountability; auditing; trusted path
- Sun MLS/Trusted Solaris/Trusted Solaris Extensions (B1+)
  - Kernel “zone;” MAC/DAC; labeled file-system/networks/desktop/printing; RBAC; storage encryption
- DEC/VAX/SVS (A1)
  - VMM security kernel; MAC/DAC; TCB enforcing Bell-Lapadula and Biba integrity models; layered design; covert signal/band analysis
- ASEC GEMSOS (A1 on an X86 platform)
- BAE’s STOP MLS B3 Guard

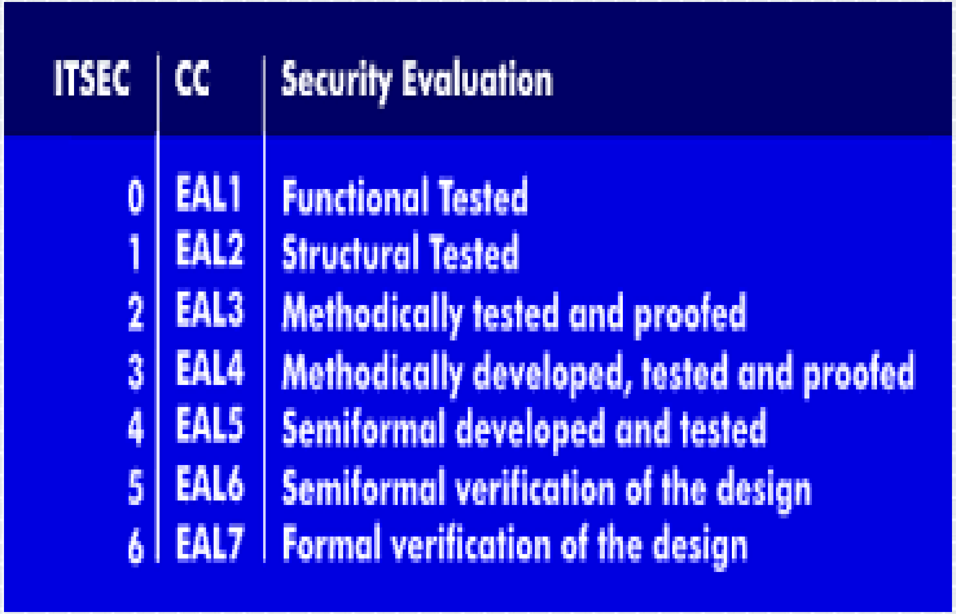
From R&D To Implementations – 1990s

- So, why did the Government Trusted Computing Initiative Fail:
  - Written by the DOD/IC community, for the DOD/IC community with only the DOD/IC community in mind
  - Too focused on Bell-Lapadula and Biba security models.
  - Underemphasized issues like identification/authentication and denial of service
  - Topics like trusted supply chain never matured into standards
  - Expense and time to have systems certified
  - Most user interfaces were clumsy and complicated

From Prescribing Requirements to Validating Features – 2000-2010

- “Globalizing” a Common Criteria
- Recognizing a broader range of “trustability”
- The Evaluation Assurance Level
- The National Information Assurance Partnership (NIAP)
- NIAP Common Criteria Evaluation/Validation Scheme for IT Security
- Most commercial operating systems at EAL 4/4+ (a TCB rating of around C2)
- Relies on a specific set of configuration settings (think GPOs) for a one-time event
- Relies on self testing and proofs

From Prescribing Requirements to Validating Features – 2000-2010



ITSEC	CC	Security Evaluation
0	EAL1	Functional Tested
1	EAL2	Structural Tested
2	EAL3	Methodically tested and proofed
3	EAL4	Methodically developed, tested and proofed
4	EAL5	Semiformal developed and tested
5	EAL6	Semiformal verification of the design
6	EAL7	Formal verification of the design

The Trusted Computing Legacy 2010>
(Partial List)

- Trusted Solaris Extensions
- SELinux/Extensions
- General Dynanmic’s PitBull (EAL 4+)
- BAE STOP (EAL 4+)
- Green Hills Software's INTEGRITY RTOS (Samsung Knox)
- Green Hills INTEGRITY®-178B (EAL 6)
- The Trusted Computing Group (TCG) Consortium
- Intel’s Trusted Execution Technology
- AMD’s Trusted Execution Technology
- The Trusted Platform Module

You Are Here – 2015
Apply What You Have Learned Today

- Understand that today’s offerings of truly “trustable” systems is sparse and incomplete
    - EAL 4+ doesn’t worry the sophisticated hackers!
- Understand that adding layers 2-7 security software and even cyber threat intelligence does not compensate for vulnerable security kernels
- Understand that we need to establish a new public-private partnership to mandate higher levels of trust in our IT networks, systems and applications
- Understand that you can play a role by influencing cyber security industry associations (e.g., ISACs) and Congress to focus more attention to the need for higher levels of trust.
    - Congress wants to talk about intelligence sharing, insurance and “hack-back!”

Time For A New Public-Private Partnership

- Should be sponsored by the White House Cybersecurity Coordinator
- Use NIST framework to establish a public-private partnership infrastructure
- Include representatives from international governments, vendors, user communities, academia, standards organizations and privacy organizations
- Publish requirements for building next generation trusted systems
- Integrate requirements into Government and industry acquisitions
- **Hold a NIST/NSF sponsored competition (similar to crypto. competition) to motivate international IT private interests to build operational models**
- **Establish a new Common Criteria Recognition Arrangement program to test and rate systems based on requirements**

Thank You

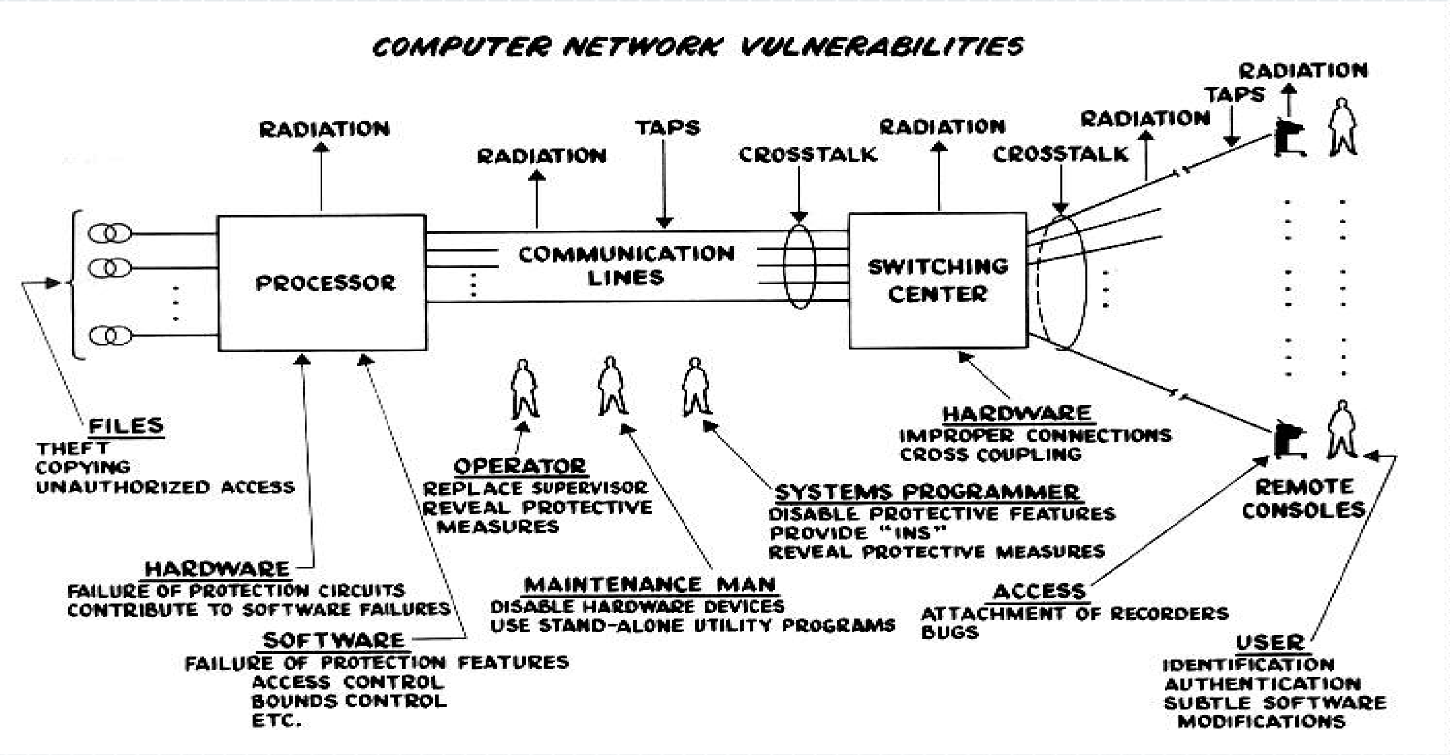
- The following people offered their time to help with this presentation:
  - Steve Lipner
  - Roger Schell
  - Ron Ross
  - Gene Spafford
  - Richard “Dickie” George
  - Mike Jacobs
  - Charles Sherupski
  - Joseph Bergmann
  - William Studeman

QUESTIONS

- **Robert Bigman**
- **2BSecure**
- **Rybbigs@Gmail.com**
- **@rybbigs**

- BACKUP SLIDES

Willis Ware - Security And Privacy - 1960s



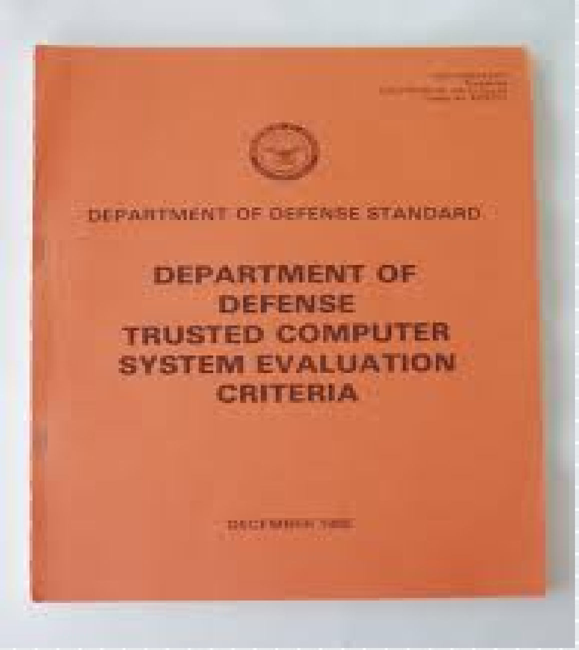
Agenda

- *Why Study Trusted Computing History?*
- *What Do You Mean: “Trusted Computing”*
- *Timeline Of Seminal Events*
- *Willis Ware - Security And Privacy - 1960s*
- *Multics And CP-67*
- *The Concept Of Trusted Computing – 1970s*
- *We’re From The Government And We’re Here To Help You – 1980s*
- *From R&D To Implementations – 1990s*
- *From Prescribing Requirements To Validating Features - 2000-2010*
- *The Trusted Computing Legacy 2010>*
- *Lessons From Trusted Computing History*
- *You Are Here – 2015*
- *Time For A New Public-Private Partnership*

IBM’s CP-67

- First successful virtual machine platform
- Strong hardware-enforced architectural separation of virtual machines
- Full isolation of user experience
- Paged memory
- Virtualized device I/O
- Bare-metal hypervisor (before the word hypervisor was used)
- CP-67 kernel was 80KB

We’re From The Government And We’re Here To Help You – 1980s



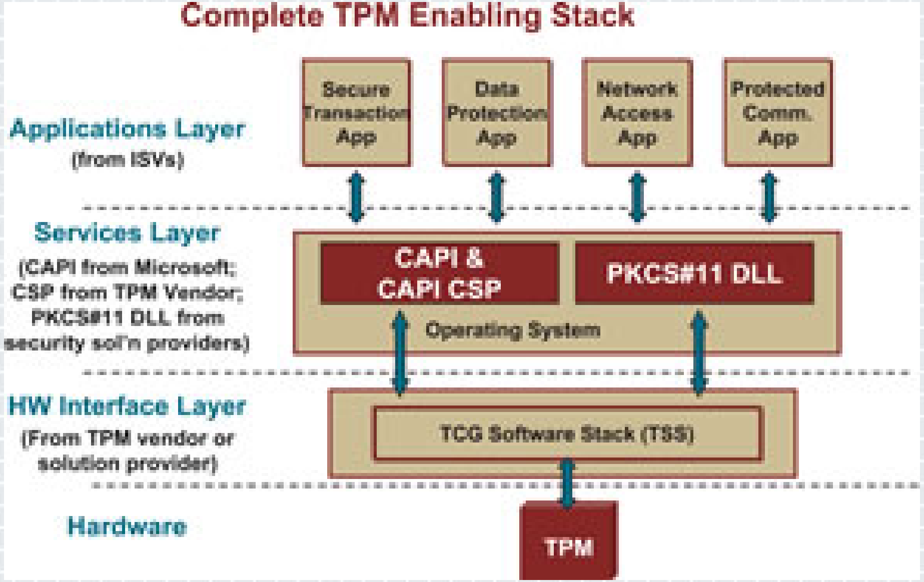
[illegible]

DEPARTMENT OF DEFENSE STANDARD

DEPARTMENT OF DEFENSE TRUSTED COMPUTER SYSTEM EVALUATION CRITERIA

DECEMBER 1985

The Trusted Computing Legacy 2010>



Lessons From Trusted Computing History

- Begin with a **Security Model**
- **Establish**, **Attest** and **Maintain O/S Trust** (in an "untrustable" environment.
- Ensure a **Small/Simple**, **Verifiable Reference Monitor**
- Establish "**Trustable**" system coding principles
- Establish **Mandatory Access Control** rules
- Ensure **Complete** mediation of rules
- Ensure "**Trustable**" event **Audit**
- Establish "**Trustable**" **Supply Chain**

- Applying the Lessons of Trusted Computing History:
  - Today’s systems lack most (if not all) the attributes to truly protect private information, process sensitive financial transactions and safely perform automated command management (e.g., IOT).
  - No amount of added security features and third party security products can substitute for a trusted computing base.
  - Trusted computing history teaches us that systems must be designed and operated with a security model that establishes and sustains a level of trust to reject subversion.
  - We need a new international public-private partnership that builds on the lessons of our trusted computing history and challenges a new generation of scientists and engineers.
  - The government can lead but industry and academia must propose solutions.