Lecture topics

- Formal system evaluation
- Threat trees
- Penetration testing
- Malicious logic

Formal evaluation of systems

- Trust based on assurance evidence
  - basis for confidence
  - not perfect security
- Helps creating secure systems
- Evaluation methodology features
  - set of requirements for security functionality
  - set of assurance requirements to establish that system meets functional requirements
  - methodology for determining that system meets functional requirements based on analysis of the assurance evidence
  - measure for evaluation (level of trust)

What evaluation guarantees?

- The system is perfectly secure – NOT!
- Lot of effort put on securing system
  - system evaluation
  - security documentation
  - development methodology
  ⇒ Expensive trill
  - cost of external evaluators
  - cost of own work
- Evaluated system less likely to have problems
Evaluation history

- Originated from military and government
- Need for assurance in commerce
  - trustworthy contractors
  - information systems
- Legal requirements, liability
  - cover your back: it was certified

Trusted computer system evaluation criteria (TCSEC)

- U.S. Government

Evaluation classes

- C1, C2, B1, B2, B3, A1 (D for failed)
- mainly information confidentiality
- no requirements for availability

Functional requirements

- discretionary access control (DAC)
- object reuse
- mandatory access control (MAC) (B1)
- label requirements (for MAC) (B1)
- identification and authentication
- trusted path (B2)
- audit mechanism
- architecture

Operational requirements

- separation of roles (B2)
- secure recovery (A1)
- system integrity validation (A1)

Assurance requirements

- configuration management (B2)
- trusted distribution (A1)
- system architecture (C1)
- design specification and verification
  - C1,C2 no requirements
  - B1 informal
  - B2 descriptive top level specification (DTLS, formal)
  - B3 DTLS consistent with security policy
  - A1 formal top level specification (FTLS) with formal methods and mapping to source code
    - testing
    - product documentation
    - internal documentation
TCSEC evaluation classes

D: failed systems that have failed evaluation

C1: discretionary protection identification and authentication

C2: controlled access protection used for commercial products

B1: labelled security protection MAC for set of objects

B2: structured protection MAC, trusted path, least privilege, covert channel analysis

B3: security domains reference validation, requirements for development methodology

A1: verified protection formal methods to evaluate B3 requirements

TCSEC evaluation process

1. Application
2. Preliminary technical review (PTR)
   - readiness review
3. Evaluation
   (a) design analysis
      - based on documentation
      ⇒ requirements for complete and correct
   (b) test analysis
      - coverage assessment
      - vendor-supplied tests
   (c) final review
      - Government-sponsored evaluators
      - Ratings maintenance program

How good TCSEC is?

- Set baseline for evaluations
- Limited scope
  - only for operating systems
  - U.S. military and government needs (no integrity, availability or other business needs)
- Problems with process
  - additional criteria
  - slow process

FIPS 140 (1994–)

- Evaluation of cryptographic modules
- Based on levels
  1. FIPS-approved algorithm; software or hardware
  2. physical security, role-based authentication, EAL2
  3. enhanced physical security, EAL3
  4. detecting and responding to physical access
Areas of requirements (11 total)
- cryptographic module specification, parts, interfaces
- roles, authentication
- logical model, design
- physical security (EMI/EMC)
- operating environment
- mitigating attacks

<table>
<thead>
<tr>
<th>level</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIPS 140-1</td>
<td>9</td>
<td>66</td>
<td>143</td>
<td>86</td>
</tr>
<tr>
<td>FIPS 140-2</td>
<td>-</td>
<td>40</td>
<td>89</td>
<td>88</td>
</tr>
</tbody>
</table>

Common Criteria (1998–)
- International followup to TCSEC, etc.
  - Common Criteria Recognition Agreement (CCRA)
  - ISO 15408
- Three parts
  - CC documents
  - CC evaluation methodology
  - national scheme
- Two types of evaluation
  PP: protection profile implementation independent set of requirements for set of products or systems
  ST: security targets evaluation of single product or system
- Requirements
  - security functional
    * 11 classes with 2–16 families
  - assurance
    * 10 classes with 2–8 families

CC Levels build on assurance
1. Functionally tested
2. Structurally tested
3. Methodically tested and checked
4. Methodically designed, tested and reviewed
5. Semi-formally designed and tested
6. Semi-formally verified design and tested
7. Formally verified design and tested

<table>
<thead>
<tr>
<th>TCSEC</th>
<th>CC</th>
<th>count</th>
<th>other</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C1</td>
<td>EAL1</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>C2</td>
<td>EAL2</td>
<td>62 need for FIPS 140-2 L2</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>EAL3</td>
<td>43 need for FIPS 140-2 L3</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>EAL4</td>
<td>63 need for FIPS 140-2 L4</td>
</tr>
<tr>
<td></td>
<td>B3</td>
<td>EAL5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>A1</td>
<td>EAL6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>EAL7</td>
<td></td>
</tr>
</tbody>
</table>
ISO 17799

- “A comprehensive set of controls comprising best practises in information security”
  - code of practise
  - specification for information security management system (BS7799-2)
- Generic information security standard
- Control IT system risks
  - product auditing, . . .
  - policy must cover all areas
- Plan-Do Check-Act

Threat modelling

- Target: understand and document security threats
- Large number of possible threats
  ⇒ Ad-hoc treat searching incomplete
  ⇒ Must be methodological
- System threat profile described
- Characterisation of system security
- Threat is *not* vulnerability
  - vulnerability is unmitigated threat
  - attack classification important

Collecting information

- How system will be used
- What system depends on (environment)
- What assumptions are made on implementation
- How system interacts to environment
- Basics of internal design decisions

Modelling system

- What are entry points
  - network, services, user interface, files, disk system
- Assets to protect
- Trust levels
  - user groups, unidentified
  - access to assets
- Data flow
  - how data flows from entry to processes
Determining threats

- Threat identification
  - shadow corners of valid-but-malicious data
  - invalid data
- Analyse threat
  - does it result a vulnerability
  - how it can be mitigated

Threat effect classification: STRIDE

Spoofing allows obtaining false identity
Tampering modifies data for goal
Repudiation not providing evidence to point guilty
Information disclosure for unauthorised user
Denial of service for legitimate users
Elevation of privilege for higher trust level

Risk of vulnerability: DREAD

Damage potential if exploited
Reproducibility of vulnerability
Exploitability how easily vulnerability can be exploited
Affected users if exploit is widely available
Discoverability is likelihood that vulnerability will be found

Assign weight for each category, calculate average.

Threat tree [1]

- Goal as tree root
- An attack is decomposed to sub-goals
  - AND all sub-goals must be meet
  - OR any of subgoals is sufficient
- Attack costs or pre-requirements can be assigned
  - helps to determine seriousness
- Reuse of attack patterns
Survivability Compromise: Disclosure of ACME proprietary secrets

OR 1. Physically scavenge discarded items from ACME
    OR 1. Inspect dumpster content on-site
    2. Inspect refuse after removal from site
2. Monitor emanations from ACME machines
    AND 1. Survey physical perimeter to determine optimal monitoring position
    2. Acquire necessary monitoring equipment
    3. Setup monitoring site
    4. Monitor emanations from site
3. Recruit help of trusted ACME insider
    OR 1. Plant spy as trusted insider
    2. Use existing trusted insider
4. Physically access ACME networks or machines
    OR 1. Get physical, on-site access to Intranet
    2. Get physical access to external machines
5. Attack ACME intranet using its connections with Internet
    OR 1. Monitor communications over Internet for leakage
    2. Get trusted process to send sensitive information to attacker over Internet
    3. Gain privileged access to Web server
6. Attack ACME intranet using its connections with public telephone network (PTN)
    OR 1. Monitor communications over PTN for leakage of sensitive information
    2. Gain privileged access to machines on intranet connected via Internet

Survivability Compromise: Disclosure of ACME proprietary secrets

ACME web server

How to blackhole traffic
How to establish unauthorised BGP session

Establish Unauthorized BGP Session with Peer

- Establish session with permissive router
- Compromise router and reconfigure to allow peering session
- Simulate BGP Session from non-router

Find available local BGP speaker
Port Scan
Social Engineering

Find available remote (EBGP-multi) speaker
Port Scan
Social Engineering

Establish peering relationship with discovered router

Configure router to allow peering from attacking router

How to originate unauthorised prefix attribute into peer route table

Originate Unauthorized Prefix/Attribute into Peer Route Table

- Send from valid Router
- Send from invalid Router
- Send spoofed BGP Update from Non-Router
- Craft BGP Message
- Validate rogue router (Assume IP)
- Conduct TCP Sequence Number Attack
- Establish unauthorised BGP session wrapper

Kill Router
Steal IP Addr
Steal MAC
ARP Spoof
Steal MAC

Power Off/Physical Layer
Crash and prevent reboot
Conduct denial of service against router

Penetration tests

- Experimental evaluation of system security
- Layered models: different threats
  1. external attacker without knowledge about system
  2. external attacker with access to system
  3. internal attacker with access to system
- Flaw hypothesis methodology
  1. information gathering
  2. flaw hypothesis
  3. flaw testing
  4. flaw generalisation
  5. flaw elimination
- Unsuccessful penetration does not prove system secure
  - cracking contests mostly useless publicity stunts

Malicious logic

Trojan horse user unintentionally executes program

- documented effect (what user expects)
- covert effect (malicious)
- Trojan in compiler 2
• “free” software add-ons (spyware, adware)
• may replicate itself

**Virus** inserts itself to file
- may have malicious actions
  - corrupts files
  - destroys equipment
- loss of performance
- several subtypes by infection, implementation method

**Worm** propagates between systems
- may have an impact on network
- most current malicious logic
  - massmailers
  - chat
  - p2p networks

**Rabbits/bacteria** exhaust resources quickly
```c
main(){for(;;)fork();} (DO NOT run code on public systems...)
```

**Logic bombs** event triggers malicious action
- disgruntled employee

**Protection against malicious code**
- Too coarse access control
  - any program has access to all my data
  - root/administrator omnipotent
- Enforcing principle of least privilege
  - sandboxing
  - capacity model
- Code signing not a solution

**Summary**
- Formal evaluation supports system development
  - for higher levels must be integral
  - overall quality assurance
- Threat evaluation needed in development
- Threat trees help large system evaluation
  - even if components are certified, network maybe insecure
- Penetration testing practical evaluation
  - needs high level of skill and experience

**References**