Research Challenges in Cyber Trust

CERIAS 4th Annual Security Symposium
Cyber Security & Safety for the 21st Century

April 9, 2003

Dr. Carl E. Landwehr,
Program Director
National Science Foundation
clandweh@nsf.gov
+1 703-292-8910
Cyber Trust Vision

Society in which

• People can justifiably rely on computer-based systems to perform critical functions
  – national scale infrastructures: water, power, communication, transportation, ...
  – localized systems: cars, homes, ...

• People can justifiably rely on systems processing sensitive information about them to conform to public policy
  – health, banking, libraries, e-commerce, government records

Without fear of sudden disruption by cyber attacks
Homeland Security
Critical Infrastructure Protection
Cyber Security
Cyber Trust

Cyber Trust
There are LOTS of Research Agendas out there!

- 1998 NSF CIP workshop
- 2001: CERIAS/Accenture Call to Action
- 2001: NSF Workshop on Information Technologies for Security
- 2002: NSF Workshop on a Research Issues in Security, Privacy and Trust (Berkeley workshop)
- 2002: CERIAS/Accenture Roadmap to Safer Wireless World
- 2002: CRA Grand Challenges workshop
- 2002: PL 105-307 topic list
- 2003: I3P Research Agenda
- 2003: NSTAC R&D Exchange
- 2003: NITRD: LSN/NRT workshop on scalable cybersecurity
A couple of things we really need to be able to do:

- Reduce software vulnerabilities
- Measure (something!)
Today’s COTS software is full of holes

“Every day in every way my job gets easier and easier”
-- Senior Sandia researcher charged with system penetration exercises
There are several reasons

- Economics drives software to include excess functions
- Customers desire for dancing pigs
  - leads to squeezing last drop of performance
  - leads to lack of internal boundaries
- Some functions will have flaws (or features) that can be exploited by penetrators and that users can’t easily disable
- So penetrators rarely lack an avenue of attack
Some underlying reasons

• Software design that lacks
  – clean internal interfaces
  – strong protection boundaries

• Programmers who lack appreciation for the importance of
  – clean internal interfaces
  – strong protection boundaries
Why are these people celebrating?
"You have to be able to code without making mistakes,"
said Lars Hellsten, a contestant from the University of Waterloo in Ontario, which placed in the top 10 ...

...

Were one to judge the state of computer science education solely by this year's contestants, the best students would appear to be in Eastern Europe and China, not Cambridge, Mass., ... or California, ... and definitely not Japan, which had only two teams in the finals.

...

As if to underscore the lopsidedness of overseas talent, it turned out that the coaches for several of the North American teams were from Poland or Russia.

What to do?

- **Pay extra for custom, high assurance software**
  - e.g., space shuttle, flight critical software, nuclear safety, weapons control, EAL-7

- **Develop system design methodologies that provide desired properties despite flawed software**
  - intrusion tolerant systems
  - anti-virus software

- **Change the economics**
  - regulate developers, certify programmers?
  - product liability?

- **Change the culture**
  - stimulate desire for higher assurance software
  - exploit software engineering advances, provide worked examples, improve the educational process
Possibility?

- Create opportunity for students to
  - build significant software component
  - that must be integrated with software produced by others
  - to produce a system with some quantifiable trustworthiness property

- Encourage competitive developments of same component /system, each producing assurance argument as part of product

- Evaluate the results and report, give prizes?

- Hoped for results:
  - several public, worked examples of how to produce trustworthy software
  - well-trained graduates
  - new knowledge about characteristics of approaches
  - useful trustworthy artifacts
  - public momentum for better software development (e.g., ACM programming contest, Robocup)
Potential artifacts

- Wallet shrinker smart card app
- TCPA platform
- Hackproof multiplayer game
- ...

Measure *(something!)*

Chronic problem for

- Computer Security
- Information Security
- Information Assurance
- Cyber Security
- Cyber Trust

- Am I better off than last year?
- How much?
Some Examples

Stolen from:

- Arbor Networks
- OASIS validation framework
  - presentation by Dr. Jay Lala at HCSS symposium; see www.tolerantsystems.org for further information
- @stake
  - Presentations by Dan Geer, paper by A. Jaquith
1. Measuring the Environment:

Continuing worm re-infection attempts

http://research.arbornetworks.com/up_media/up_files/snapshot_worm_activity.pdf [Arbor Networks, Inc.]

Recurrence of initial “Code Red” worm launched 7/12/01
2. Measuring the System in the Environment: System Validation from DARPA’s OASIS program

- OASIS idea: develop technologies to support survivable information systems -- operate through attacks

- Problems:
  - How to convince the management there’s an improvement?
  - How to explain the scope of the technologies addressed by the whole program?

- Approach: create validation framework to structure how PI’s characterize their technologies
Define terms based on prior Dependability work (IFIP WG 10.4)

- A system or more generally a technology has certain functional goals over a domain of application along with certain supporting information assurance and survivability attributes for protection.

- The system or technology may not be able to achieve its functional goals because of certain vulnerabilities or attacks (or threats).

- However, the system or technology may counter the vulnerabilities or attacks by protection mechanisms/means that are intended to provide for its particular attributes and assure that it achieves its functional goals.
Framework for describing each project

1. What is the technology and what information assurance/survivability (IA/S) problem is being addressed?

2. What assumptions does the technology make about the environment, system, user, hardware, etc.?

3. What are the vulnerabilities/attacks the technology is trying to address? (design time, implementation time, operation)

4. What IA/S attributes is the technology trying to assure?
## Sample validation matrix

### IA/S properties

<table>
<thead>
<tr>
<th>Design</th>
<th>Operation</th>
<th>AV</th>
<th>I</th>
<th>C</th>
<th>AU</th>
<th>NR</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAV-1.1</td>
<td>TAV-3.1</td>
<td></td>
<td>A2, M5¹</td>
<td></td>
<td></td>
<td></td>
<td>M1, M3, M6²</td>
</tr>
<tr>
<td>TAV-1.2</td>
<td>TAV-3.2</td>
<td></td>
<td>M4³</td>
<td></td>
<td></td>
<td></td>
<td>M1, M3, M6²</td>
</tr>
<tr>
<td>TAV-2.1</td>
<td>TAV-3.3</td>
<td></td>
<td>TCB⁴</td>
<td></td>
<td></td>
<td></td>
<td>M1, M3, M6²</td>
</tr>
<tr>
<td>TAV-2.2</td>
<td>TAV-3.4</td>
<td></td>
<td>M4³</td>
<td>M4⁸</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAV-3.5</td>
<td></td>
<td></td>
<td>A3, M7, note⁹</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAV-3.6</td>
<td></td>
<td></td>
<td>M8, note⁹</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Allow some visualization of “coverage”**
Details (20+ sample project validations) available: www.tolerantsystems.org

Just the beginning -- still need better:

- Concepts and terms to succinctly express IA domain issues
- Threat, attack and vulnerability taxonomies
- Security models and models of attacker intent, objectives, and strategies
- Work factor metrics, survivability metrics, operational security metrics, cryptographic protocol metrics
- Methods for testing and validating protection mechanisms
- Security and survivability requirements specifications
Dan Geer, @Stake

Application Penetration Testing Approach

- Define Target Application(s)
  - Hypothesize Threats
    - Build Test Environment (as req.)
      - Generate Findings
        - Document Findings
- Understand Architecture
  - Analyze Component
    - Identify Risks
      - Analyze Risks
        - Conduct Proof of Concept (as req.)
  - Discuss Vulnerability Risk
    - Understand Technical and Business Context
      - Develop Action Plan for Improvement
- Document Findings
  - Develop Plan

Up-to-date Vulnerability/Threat Knowledge
Findings (1/4): Security Defects Are Common

Security Defects by Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Engagements observed</th>
<th>Design related</th>
<th>Serious design flaws*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrative interfaces</td>
<td>31%</td>
<td>57%</td>
<td>36%</td>
</tr>
<tr>
<td>Authentication/access control 62%</td>
<td>62%</td>
<td>89%</td>
<td>64%</td>
</tr>
<tr>
<td>Configuration management</td>
<td>42%</td>
<td>41%</td>
<td>16%</td>
</tr>
<tr>
<td>Cryptographic algorithms</td>
<td>33%</td>
<td>93%</td>
<td>61%</td>
</tr>
<tr>
<td>Information gathering</td>
<td>47%</td>
<td>51%</td>
<td>20%</td>
</tr>
<tr>
<td>Input validation</td>
<td>71%</td>
<td>50%</td>
<td>32%</td>
</tr>
<tr>
<td>Parameter manipulation</td>
<td>33%</td>
<td>81%</td>
<td>73%</td>
</tr>
<tr>
<td>Sensitive data handling</td>
<td>33%</td>
<td>70%</td>
<td>41%</td>
</tr>
<tr>
<td>Session management</td>
<td>40%</td>
<td>94%</td>
<td>79%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>45</strong></td>
<td><strong>70%</strong></td>
<td><strong>47%</strong></td>
</tr>
</tbody>
</table>

*Scores of 3 or higher for exploit risk and business impact

Top 10 Application Security Defects

<table>
<thead>
<tr>
<th>Defect</th>
<th>Assessments where encountered, percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session replay/hijacking</td>
<td>31%</td>
</tr>
<tr>
<td>Password controls</td>
<td>27%</td>
</tr>
<tr>
<td>Buffer overflows</td>
<td>27%</td>
</tr>
<tr>
<td>File/application enumeration</td>
<td>27%</td>
</tr>
<tr>
<td>Weak encryption</td>
<td>24%</td>
</tr>
<tr>
<td>Password sniffing</td>
<td>24%</td>
</tr>
<tr>
<td>Cookie manipulation</td>
<td>20%</td>
</tr>
<tr>
<td>Administrative channels</td>
<td>20%</td>
</tr>
<tr>
<td>Log storage/retrieval issues</td>
<td>20%</td>
</tr>
<tr>
<td>Error codes</td>
<td>20%</td>
</tr>
</tbody>
</table>
Findings (2/4): *Leaders Have Fewer Defects*

Average defects per engagement, by risk category

- **Administrative interfaces**: 0.3 (First quartile), 2.7 (Fourth quartile)
- **Authentication and access control**: 0.7 (First quartile), 6.5 (Fourth quartile)
- **Configuration management**: 1.2 (First quartile), 3.3 (Fourth quartile)
- **Cryptographic algorithms**: 0.3 (First quartile), 0.5 (Fourth quartile)
- **Information gathering**: 1.0 (First quartile), 1.3 (Fourth quartile)
- **Input validation**: 1.3 (First quartile), 3.5 (Fourth quartile)
- **Parameter manipulation**: 0.2 (First quartile), 1.8 (Fourth quartile)
- **Sensitive data handling**: 0.3 (First quartile), 3.3 (Fourth quartile)
- **Session management**: 0.7 (First quartile), 3.3 (Fourth quartile)

Source: 2002 @stake - The Hoover Project (n=23)
Findings (3/4): *Leaders Carry Less Risk*

Average business-adjusted risk (BAR) index per engagement, with breakdown by risk category.

- **Business-adjusted risk index**
  - **Bottom quartile**: 331.8 score
  - **Top quartile**: 60 score
  - **Risk reduction**: 82%

- **Administrative interfaces**: 36.2 score
- **Authentication/access control**: 85.2 score
- **Configuration management**: 36.3 score
- **Cryptographic algorithms**: 6.8 score
- **Information gathering**: 11.0 score
- **Input validation**: 46.3 score
- **Parameter manipulation**: 31.5 score
- **Sensitive data handling**: 34.5 score
- **Session management**: 44.0 score

Source: 2002 @stake - The Hoover Project (n=23).

BAR index = sum of all defects' individual BAR scores, where each defect's score = exploit risk (5 point scale) x business impact (5 point scale).
Findings (4/4): *Fixing security defects earlier pays off*

- Although benefits can be found throughout the lifecycle, earlier involvement is most beneficial.
- Vulnerabilities are harder to address post-design.
- System-wide changes may be required at later stages.
- Enabling improvements can be made at design state.

![Security ROI by Phase](image-url)

*Source: 2002 @stake - The Hoover Project*
Concluding thoughts

1. Regardless of the threat, today’s systems are way too vulnerable

2. There is an opportunity now for increased investment in this area
   - Trusted Computing
   - Trustworthy Computing
   - CSRDA
   - DHS

3. Unless we can demonstrate measurable improvements, the investments will be short-lived
Thank you

- Questions?