Dan Geer

geer@stake.com
+1.617.768.2723
Art v. Science
Characterization and Specialization
Time Line and Drivers
Put up or shut up...
Applications are where the action is

- Security trends say so
- Business realities say so
- Risk management needs quantitative decision support
- Application pen-tests can yield that support
Security trend 1
*Applications are federating*

- **Distributed applications have multiple security domains**
  - **The firm:** client service & administrative functions
  - **External providers:** front-end Web farms and application hosting
  - **Partner interfaces:** data streams (inventory, payment, real-time feeds)

- **Applications get ever more moving parts**
  - Mainframe $\rightarrow$ client-server $\rightarrow$ $n$-tier $\rightarrow$ Model 2 (J2EE and .Net)

- **Network service stratification**
  - Bandwidth, hosting, provisioning, delivery
Security trend 2

*Perimeter defense is increasingly diseconomic*

- “Shared wire” supplants “shared model”
  - XML is the great equalizer
  - SOAP and XML-RPC specifically designed to go through firewalls
  - Emerging web services

- **Firewalls stop nuisance attacks, not application traffic**
  - Everyone leaves ports 80 and 443 open

- **As a result, the threat model mutates**
  - More attacks through HTTP, at application level
  - More attacks targeted at specific application components
  - Attacks on applications require lower skill levels
Security trend 3
*Data, data everywhere*

- Data storage needs increasing exponentially
  - More new data produced in next 3 years than in all of human history
  - Corporate IT spending 4% in 1999 v. 17% in 2003 (Forrester)

- Form factors proliferating
  - Local storage
  - Storage arrays
  - Appliances/network-attached storage

Moore’s Law, 18mo doubling
Storage, 12mo doubling
Bandwidth, 9mo doubling
Corresponding business realities

- Risk management has won
- Anticipate failure or be damned
- Demand for security expertise exceeding supply

*But most importantly,*

- The future belongs to the quants
Quantitative decision support for risk management

- **Annualized Loss Expectancy**
  \[ = \sum (\text{probability} \times \text{business impact}) \]  
  \{ Before investment, and after \}

- **Net Present Value**

  **Increased Revenues**
  - Improved Uptime
  - Transactional Frequency
  - New Referrals

  **Decreased Direct Costs**
  - Developer Re-work
  - System Administrator Labor
  - Patch Release Costs
  - Customer Retention

  **Cost Avoidance (soft costs)**
  - Media/Legal

  \[ = \text{Net Investment Return} \]
Treat application security as you would quality

<table>
<thead>
<tr>
<th>Relative cost to fix issues, by stage</th>
<th>Software development costs, by stage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design</td>
<td>Design</td>
</tr>
<tr>
<td>Implementation</td>
<td>Implementation</td>
</tr>
<tr>
<td>Testing</td>
<td>Testing</td>
</tr>
<tr>
<td>Maintenance</td>
<td>Maintenance</td>
</tr>
<tr>
<td>1</td>
<td>15%</td>
</tr>
<tr>
<td>6.5</td>
<td>60%</td>
</tr>
<tr>
<td>15</td>
<td>25%</td>
</tr>
<tr>
<td>100</td>
<td>15%</td>
</tr>
</tbody>
</table>

Source: Implementing Software Inspections, IBM Systems Sciences Institute, IBM, 1981
Source: Architectures for Software Systems, course Notes, Garlan & Kazman, CS, CMU, 1998
A little example of pooled data

Security evaluation of major applications treated as a source of summary numbers and shared intelligence

All data are real, pooled and hence anonymized within a trust relationship, and modeled as normative
Application Penetration Testing Approach

1. Define Target Application(s)
2. Understand Architecture
3. Hypothesize Threats
4. Build Test Environment (as req.)
5. Analyze Component
6. Identify Risks
7. Analyze Risks
8. Conduct Proof of Concept (as req.)
9. Generate Findings
10. Discuss Vulnerability Risk
11. Understand Technical and Business Context
12. Develop Action Plan for Improvement
13. Document Findings
14. Implement Plan

Up-to-date Vulnerability/Threat Knowledge
## Finding 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</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>

Source: 2002 @stake - The Hoover Project (n=45)
Finding 2/4: *Leaders have fewer defects*

**Average defects per engagement, by risk category**

- **Administrative interfaces**
  - Fourth quartile: 23.0
  - First quartile: 4.8
  - Average: 10.6

- **Authentication and access control**
  - Fourth quartile: 6.5
  - First quartile: 0.7
  - Average: 3.6

- **Configuration management**
  - Fourth quartile: 3.3
  - First quartile: 1.2
  - Average: 2.2

- **Cryptographic algorithms**
  - Fourth quartile: 2.7
  - First quartile: 0.3
  - Average: 1.5

- **Information gathering**
  - Fourth quartile: 1.3
  - First quartile: 1.0
  - Average: 1.1

- **Input validation**
  - Fourth quartile: 3.5
  - First quartile: 1.3
  - Average: 2.4

- **Parameter manipulation**
  - Fourth quartile: 1.8
  - First quartile: 0.2
  - Average: 1.0

- **Sensitive data handling**
  - Fourth quartile: 3.3
  - First quartile: 0.3
  - Average: 1.8

- **Session management**
  - Fourth quartile: 3.3
  - First quartile: 0.7
  - Average: 2.0

Source: 2002 @stake - The Hoover Project (n=23)
Finding 3/4: *Leaders carry less risk*

<table>
<thead>
<tr>
<th>Risk Category</th>
<th>Bottom quartile</th>
<th>Top quartile</th>
<th>Risk reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business-adjusted risk index</td>
<td>331.8 score</td>
<td>60 score</td>
<td>82%</td>
</tr>
<tr>
<td>Administrative interfaces</td>
<td>36.2</td>
<td>4.0</td>
<td>89%</td>
</tr>
<tr>
<td>Authentication/access control</td>
<td>85.2</td>
<td>10.3</td>
<td>88%</td>
</tr>
<tr>
<td>Configuration management</td>
<td>36.3</td>
<td>8.7</td>
<td>76%</td>
</tr>
<tr>
<td>Cryptographic algorithms</td>
<td>6.8</td>
<td>2.5</td>
<td>63%</td>
</tr>
<tr>
<td>Information gathering</td>
<td>11.0</td>
<td>8.8</td>
<td>20%</td>
</tr>
<tr>
<td>Input validation</td>
<td>46.3</td>
<td>14.5</td>
<td>69%</td>
</tr>
<tr>
<td>Parameter manipulation</td>
<td>31.5</td>
<td>3.3</td>
<td>89%</td>
</tr>
<tr>
<td>Sensitive data handling</td>
<td>34.5</td>
<td>2.5</td>
<td>93%</td>
</tr>
<tr>
<td>Session management</td>
<td>44.0</td>
<td>5.3</td>
<td>88%</td>
</tr>
</tbody>
</table>

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

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).
Finding 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

Source: 2002 @stake - The Hoover Project
Repeating: *Applications are where the action is*

- Security trends say so
- Business realities say so
- Risk management means quantitative decision support
- Application pen-tests can yield that support

*And if they don’t, what’s the point?*
Questions?