Security + Agile = FAIL

Adrian Lane
Securosis, L.L.C.
What We’ll Cover

• What’s great about Agile
• Why it fails for Secure Code Development
• Contrast other processes
• The People Problem
• Recommendations
• Q&A
People & Process
Make Things Simple
What’s Great About Agile

• Focus
  – Granular approach reduces complexity
  – Natural ‘divide & conquer’ approach

• Efficiency
  – Always working on highest priority item
  – Mistakes found, reversed faster
  – Decentralization and less bureaucracy

• Communication
  – Scrum put everyone face to face
  – Peer pressure
Efficiency vs. Security
Agile does …

• Reduce complexity
• Focus on high priority issues

Agile does not …

• Deal with the abstract (undefined tasks, ROI)
• Inherently enforce code reviews
• Mandate secure code style
• Help prioritize security
Web App Security

Breaking Trust Relationships

Browser -> Cross Site Scripting

Browser -> Cross Site Request Forgery

Browser -> SQL Injection

Server
Threat Modeling

Remote DB Access

Web Application Servers

Application Servers
Batch Jobs

Archive / Export

Internal Users
Sprint Duration
<table>
<thead>
<tr>
<th><strong>Foo-1529</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure the code.</td>
</tr>
<tr>
<td>Status: In Progress</td>
</tr>
<tr>
<td>Module: All</td>
</tr>
<tr>
<td>Version: Iteration 4284</td>
</tr>
<tr>
<td>Remaining: $2 \times 10^{16}$</td>
</tr>
<tr>
<td>Assigned: Barry</td>
</tr>
</tbody>
</table>
User Stories vs. Attacker Stories

This one time, at Hacker camp ...
Chickens & Pigs

- Security specialists become ‘witness’
- Typical waterfall release gates removed
Squeezed Testing

- Fuzzing
- Black Box
- Security Regression
- Transaction consistency
- Repudiation
- Trust relations (privilege)
- Perpetual Beta
5lbs of Stuff in 2lb Bag

- No one creates ‘stories’ for attacks
- Loosely defined requirements
- Assessment’s are not a neat 4 hour task
- Threat modeling is systemic; rarely modular
- Code dependencies & trust relationships
- Sprint too short for meaningful testing
- Design & coding shoved into same sprint
Rock Beats Scissors

People Beat Process
The People Problem

• We use process to guide people
• Requirements invariably are more, better and faster – lends itself to agile
• Developers paid to build – not destroy
• Coding practices, threat modeling, undermining systems is not part of the mindset
• Motivation the same, but manifestation different
Security Decisions by PM

Less conceptual planning by this guy

More security decisions by this guy
Project Management

- Least qualified to make security decisions
- Just in time planning
- Sprint detritus avoidance (i.e. security backlog)
- COTS vs. Bootstrap approach
- Metrics different
Let’s Not Forget Coder

- The Wallflower – mum in the scrum. They’re not really working. Week four the cram like it’s midterms.

- The ‘Expert’ – Knows everything about everything, and argues effectively, so they get their way. Quotes Gosling, Ritchie, Matsumoto. Volunteers to do code reviews, hates it if you critique their code.

- The False Guru – admired by peers because they ‘know the code’ but is personally responsible for all the buried corpses you find.

- Mrs. By The Book – “That’s the way Schwaber did it”.
## Who Got Skillz

<table>
<thead>
<tr>
<th>Experience</th>
<th>Perception of Role</th>
<th>Process Knowledge</th>
<th>Product Security Knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Newbie (0-1 years)</td>
<td>“I fix bugs.”</td>
<td>“What?”</td>
<td>“We use passwords”</td>
</tr>
<tr>
<td>Apprentice (1-3 years)</td>
<td>“I work on the web services back end”</td>
<td>“Waterfall. I think”</td>
<td>“We use a firewall to protect the application”</td>
</tr>
<tr>
<td>Skilled Individual</td>
<td>“I designed and built the web services application on a struts framework, leveraging ...”</td>
<td>“We use Agile with Scrum on a 4 week sprint and daily meetings”</td>
<td>“The development team uses unit tests and code reviews, while QA uses security regression and fuzzing ...”</td>
</tr>
<tr>
<td>Skilled Manager</td>
<td>“I organize development activities across design, development and release management groups for commerce applications”</td>
<td>We use several processes, and periodically synch requirements, development &amp; release management cycles with I.T. to ...”</td>
<td>“Each phase of code production checks for code security, and we rely upon network and platform ...”</td>
</tr>
</tbody>
</table>
Chicken Scratch

- Sales team focused
- Culture of commitment counter to agile
- ‘Their’ customers don’t buy security
- Critical features become ‘deal enablers’
What to do?
• Modify Some Tasks to Fit Agile
• Threat Modeling
• Attacker stories
• Modify Agile to Fit Some Tasks
• Fuzzing
• Design
• Release Management
Prepare

- Risk Based Feature Reprioritization
- Train Developers – Enforce Testing
- Keep Metrics – not just post-it’s
- Determine ‘every-sprint’ requirements
- Identify Problem Code
- Identify Problem Developers
Modify the Process

• Use Management Software
• ‘Bucket List’ & Age should increase priority
• Only release what’s secure (i.e. you can’t ‘prioritize out’ security)
• Leverage Agile – pose security as design challenges and let developers make choices
• Hire Independent reviewer who doesn’t care about your internal politics.
Modify the Release

- Release Management need not be fully agile
- Adapt test cases
- Static/white box analysis
- Periodically wrap the release in complete regression sweep (e.g. pseudo-waterfall)
- Rapid prototype <-> rapid release: functionally complete software into security testers hands for review before release
Summary

- Agile process affects security
- Agile decision makers change security
- Agile security more reliant on programmer skill
- Need to modify prioritization
- Modify release cycle to accommodate testing
Security + Agile = FAIL

Adrian Lane
Securosis, L.L.C.
alane@securosis.com
http://securosis.com