Agile Security Manifesto

Applying Agile Principles to Software Security

Nick Murison
February 9, 2017
About Me

• Consultant ~12 years
  – Penetration testing
  – Incident response
  – Training
  – Software security: code, design, risk
  – Company-level initiatives to improve software security

• BSc in Computer Science
• MSc in Information Security
The Challenge

- Idea → Prototype → Reality

When do we put security into the design?
In the old days… waterfall

Requirements Gathering

Architecture and Design

Code and Development

Testing

Deployment
The Agile Manifesto is a set of principles, not a rigid methodology. There are many agile methodologies, including scrum and Kanban.
“Agile Security Manifesto”
Cigital’s “Agile Security Manifesto”

- Four principles, inspired by the original Agile Manifesto

- **Goal:** guide the development of new activities *and make adjustments to existing activities* to make it natural and efficient to build security into an agile process

  - Rely on good developers and testers
    - Implement **secure** features
    - Continuously improve security
    - Focus on fixing software
  - Relying on security experts
    - Implementing security features
    - Completely changing processes
    - Finding bugs
Security and Agile: “Culture Clash”? 

• Security has traditionally been about controls 

• Agile is about creating a culture of collaboration and cultivation 

• Things that don’t often work: 
  – Imposing strict controls in an Agile culture 
  – Applying Agile principles in a strict control culture 

• We need to evolve security to better align with the culture, e.g. 
  – “Controls” can become “activities” 
  – Automation (e.g. CI/CD) can help introduce more security activities; seize on this! 
  – **Collaborate** and **enable** developers to achieve their objectives
Rely on good developers and testers over security specialists

• Security experts are valuable, and bring specific skills/experience

• But most agile teams don’t have an embedded security expert
  – Costly, and far between

• It’s easier to teach developers about security than teaching a security expert how to code

• **Empower** your agile teams to own security through:
  – Training
  – Automation (CI/CD)
  – Defining security user stories
  – Providing access to SMEs
Implement secure features over adding security features

• Focus on business value and achieving the software’s mission
  – Don’t force your developers to solve hard security problems when they’re trying to implement functionality

• Use SMEs to develop security features
  – E.g. authentication, authorization, data validation, crypto, etc.
  – Make these available and easy to use for development teams

• Enable your developers to write high quality secure components
  – Have them leverage secure-by-default frameworks
  – Make it easy for them to leverage the available security features
Continuously improve security over completely changing processes

• If an agile team is working well and producing good software, it probably needs very little change to produce secure software

• If the existing process is not producing good software, then adding security activities will not fix that

• Make the security experts use the same tools and paradigms as developers to allow information to flow
  – Security requirements should be communicated like user (security) stories
  – Security defects should go in the developer backlog tools
  – Security tests should integrate in the developers’ existing testing pipeline

• Focus on making incremental changes to how you approach software security, e.g.:
  – Add a small set of security rules to SonarQube, configured to not “break build” in your CI/CD pipeline
  – Wait a few sprints before adding more
  – Keep adding until you’ve covered your list of “Top N Bugs We Don’t Want To Have”
  – Start breaking builds for the "worst" type of bug
  – Add breaking builds for the “second worst” type of bug
  – Etc...
Focus on fixing software over finding bugs

• Penetration testing, secure code review etc. find issues, but they don’t magically fix them for you

• Automated security tools are great at findings issues…
  – … even if the issue doesn’t exist!
  – And they don’t fix the issues for you

• Apply a risk-based approach to focus development effort on the issues that matter and that cannot be handled through other means
  – E.g. business process, contracts, monitoring, etc.

• Use the development backlog to communicate and prioritise issues that need to be remediated
Modern Software Development

Planning → Implementation → Deployment → Testing → Evaluation → New Needs
Modern Software Development with Security

- Planning:
  - Security User Stories
  - Secure Design Review

- Implementation:
  - Secure Code Review
  - Security Testing

- Deployment:
  - Penetration Testing

- New Needs
- Evaluation
- Testing
Next Steps

• Measure what activities you have
  – E.g. BSIMM

• Identify what activities are reasonable for your organisation to add / update

• Implement activities incrementally

• Measure and re-measure
The best time to plant an oak tree was twenty years ago.

The next best time is now.

—Ancient Proverb

Nick Murison
nmurison@synopsys.com
@nickmurison
Thank You