Painless Agile Security

Why is this all so hard?

Julian Simpson, Safe Advisory

Thank You to Our Sponsors and Hosts!

AppSec NZ appsec.org.nz

OWASP NEW ZEALAND owasp.org.nz

AUT UNIVERSITY TE WĀNANGA ARONUI O TAMAKI MAKAU RAU

BASTION SECURITY GROUP

DATACOM

aws

84.

PentesterLab

plexure

VERACODE

Without them, this Conference couldn't happen.

The next 30 minutes

- How did we get here?
- How does Agile go wrong?
- How do you make it better?

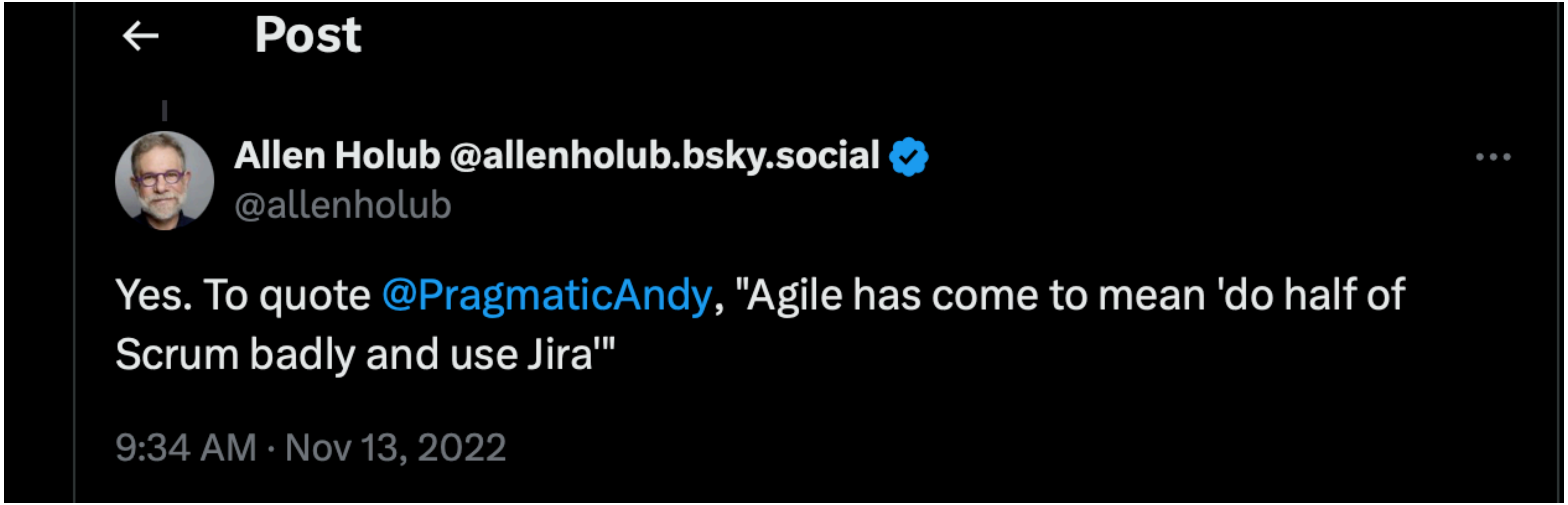
1 minute of Agile hate

- Scrum Masters (awkward name!)
- Sprint packing
- Sprint planning meetings
- Micromanagement
- Daily standoffs

Getting Agile wrong

Getting Agile wrong

Also, we don’t understand Rugby metaphors



Post

Allen Holub @allenholub.bsky.social
@allenholub

Yes. To quote @PragmaticAndy, "Agile has come to mean 'do half of Scrum badly and use Jira'"

9:34 AM · Nov 13, 2022

Getting Agile wrong

What do we mean when we say Agile?

- There are many Agile methodologies (with a big “A”)
- There are lots of agile practices (with a small “a”)
- Process alone will not help teams deliver working software to production
- That takes a team, with skills and autonomy

Getting Agile wrong

The 90s

- There was no Git, SVN, or TFS
- You were probably doing well to have any source control
- Developers branched code for months
- JIRA didn’t exist
- C# didn’t exist
- People still used Visual Basic and Perl

Getting Agile Wrong

2001-ish

- Kent Beck publishes the eXtreme Programming Explained book
- Based on real world success of Smalltalk nerds at Chrysler in late 90s
- Project was a real success and they wrote up the lessons as XP

Getting Agile Wrong

12 XP practices

- Pair programming
- Planning game
- Test-driven development
- Whole team
- Continuous Integration
- Refactoring
- Small releases
- Coding standards
- Collective Code Ownership
- Simple Design
- System metaphor
- Sustainable pace

Getting Agile Wrong

As XP takes off, Scrum appears from 1993 to ruin the match

- Scrum team
- Sprints
- Scrum Master
- Product Owner
- Product backlog
- Sprint backlog
- Velocity

Getting Agile Wrong

How the scrum collapses

- Teams can stick to the scrum “rules”
- There are no rules about software development practice
- That works, until delivery stops because of technical debt

Ways to do Agile, poorly

- Prioritising delivery over code quality
- “Refactoring sprints”
- Features without testing, security, or operational requirements
- Estimates as a commitment
- Sprint stuffing
- Senior management in standup and retrospective
- Not shipping to production when code is “done”

Doing Agile better

Everything fits in the timebox

Doing Agile better

Everything fits in the timebox

- An Agile sprint or iteration is meant to contain the full lifecycle of a project
- That means TDD (before the code)
- Acceptance tests (after the code, in the same repo if you can)
- Security
- Deployment

Doing Agile better

Everything fits in the iteration / sprint

- If that means you do fewer features, but better, that is OK
- Retrospectives, velocity, yesterday’s weather etc. are designed to help the team do better in the next iteration or sprint

Doing Agile better

Estimates are for the devs

- The team doing the work needs to estimate the **complexity** of the work (not the duration)
- If things are more complicated than expected, the team can review their estimation
- Estimated complexity shouldn’t be a delivery commitment

Doing Agile better

Iterative, not Incremental

- Incremental approaches are for building houses
- We iterate on software every day
- We also iterate on features as we understand more

Doing Agile better

How do you know when it works?

- TDD is a superpower for iterative work
- You get to make changes in the small in code with lower risk
- Acceptance tests prove that your app can be “wired up” properly

Doing Agile better

You can measure quality

- Unit Test Coverage %
- Acceptance Test Coverage %
- Number of Code Smells
- Percentage of Duplication
- Cyclomatic Complexity
- Number of SAST vulnerabilities

Doing Agile better

Beware of Conway’s Law

- Teams need to deliver working code
- In a single branch
- Abandoned branches (including PRs not merged) don’t help us deliver code
- Continuous Integration and Trunk Based Development fixed that in the 90s

Sneaking security into the sprint

Sneaking security into the sprint

Start Small

- Agile is all about breaking work up into small chunks
- Any security tool is better than nothing
- Adding new analysis is an iterative and incremental improvement - and it doesn’t matter if you throw it away later
- That could be SCA, SAST, DAST.
- Just do something. Inside this sprint.

Sneaking security into the sprint

Work with Developers and QA

- Find the team members who care about the whole thing
- You're a team member, not a cop - gain everyone's trust by helping them with security
- Use the team's tools
    - e.g. if they use Atlassian products, hold your nose and use them too
    - or if they like diagrams in code, learn those

Sneaking security into the sprint

Fit into the team’s workflows

- You want to find security issues in new code
- Ideally within the sprint, so it’s still relevant to the devs
- And work with them through standups and retrospectives to help them get better

Summary

- Processes or Methodologies aren’t enough
- But it is possible to do Agile well
- You can do security in Agile, but you need to work with it

Thanks!

- If there’s time, feel free to ask questions
- Or come have a chat if you see me
- julian@safeadvisory.co.nz
- https://www.linkedin.com/in/juliansimpson