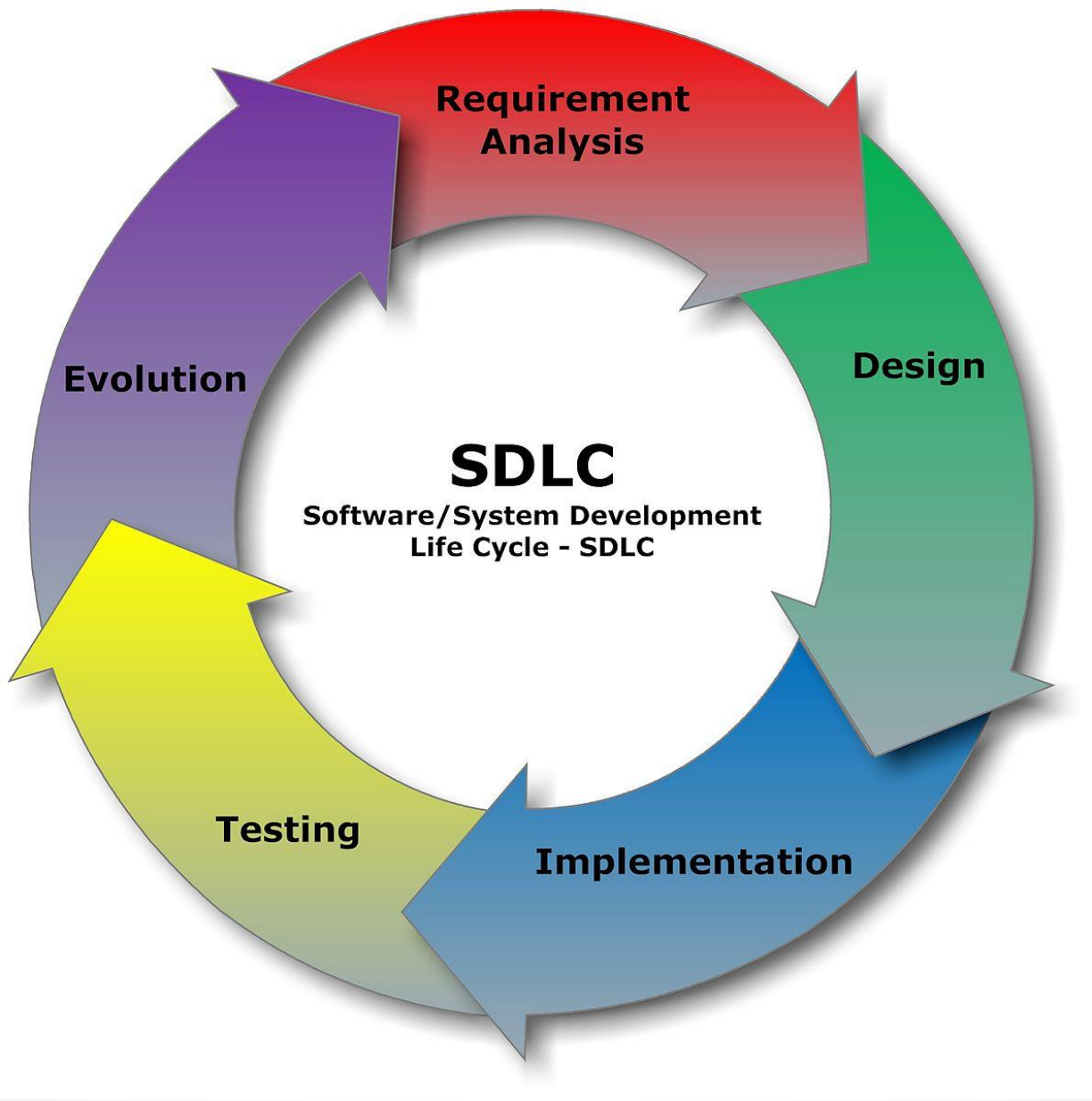
Secure Software Development

Prof. Zaheed Shaikh

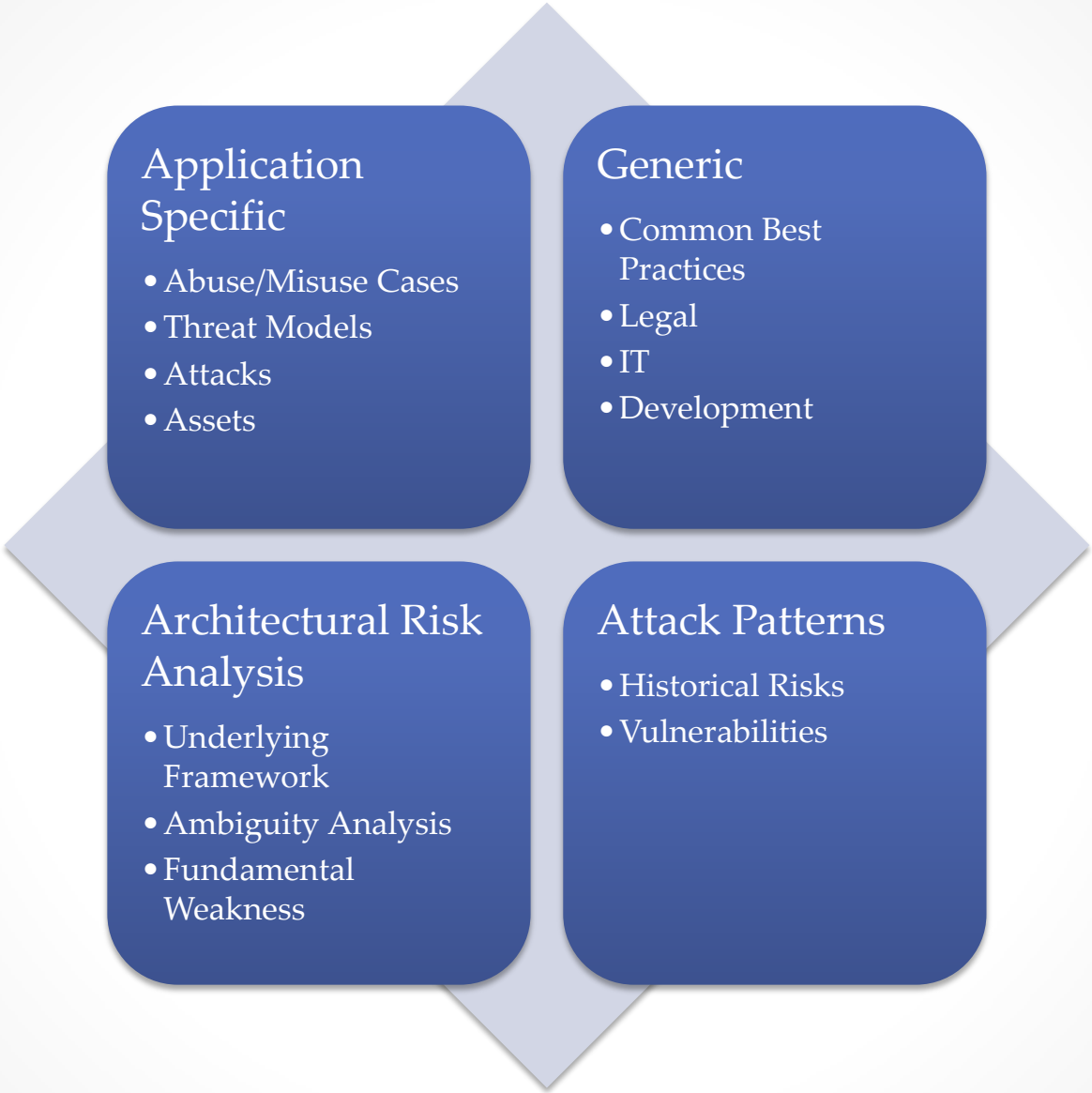

Requirement Analysis
Design
Implementation
Testing
Evolution
SDLC
Software/System Development Life Cycle - SDLC

Secure Software Development

- Consider security throughout the software development lifecycle
  - Requirements
  - Design
  - Implementation
  - Testing
  - Deployment

Requirements

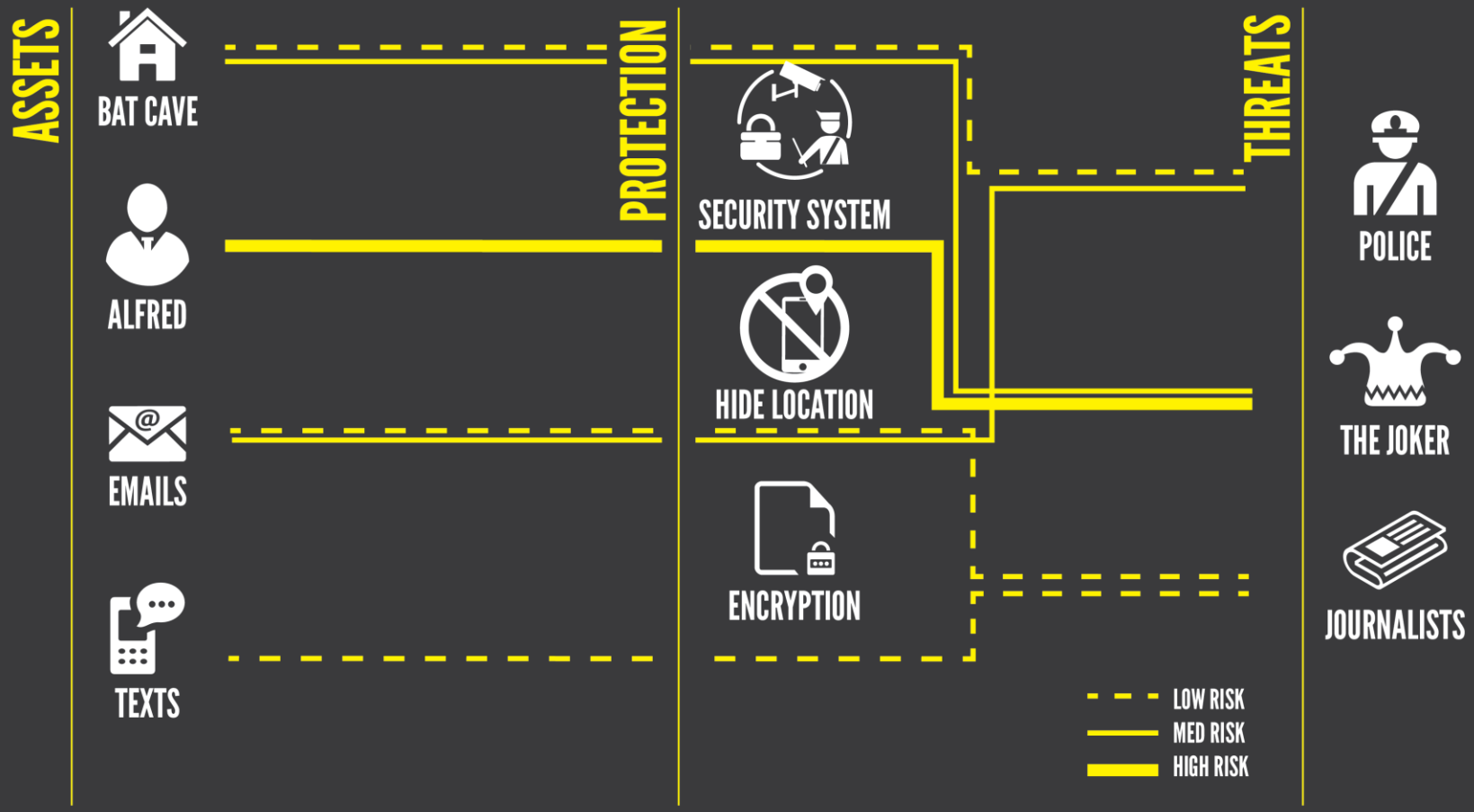
- Identify sensitive data and resources
- Define security requirements for them
  - Confidentiality
  - Integrity
  - Availability
- Consider threats and abuse cases that violate these requirements


Application Specific
• Abuse/Misuse Cases
• Threat Models
• Attacks
• Assets
Generic
• Common Best Practices
• Legal
• IT
• Development
Architectural Risk Analysis
• Underlying Framework
• Ambiguity Analysis
• Fundamental Weakness
Attack Patterns
• Historical Risks
• Vulnerabilities

Design

- Apply principles for secure software design
  - Prevent, mitigate and detect possible attacks
- Security principles
  - Favor Simplicity
  - Trust with Reluctance
  - Defend in Depth

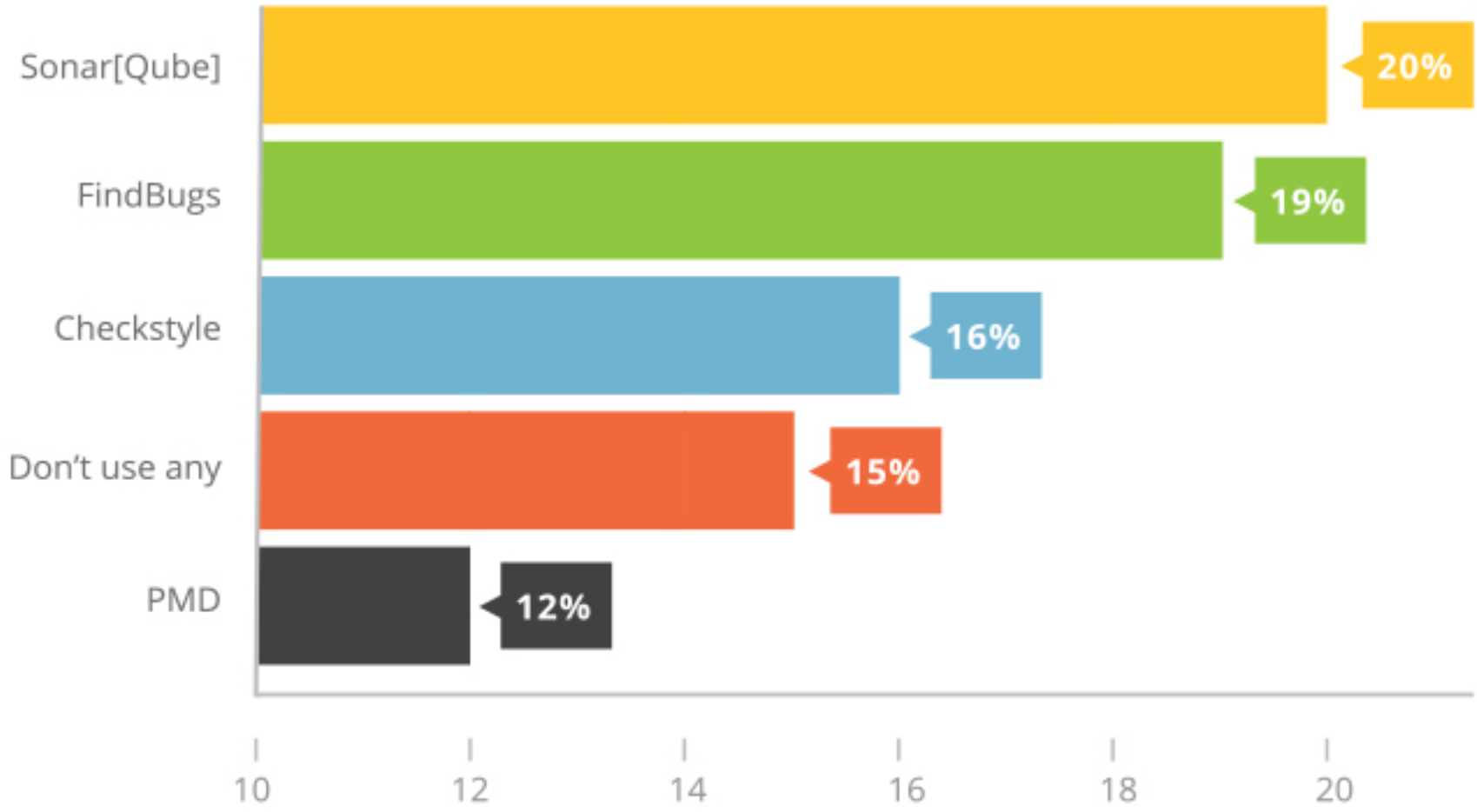
BRUCE WAYNE/BATMAN'S THREAT MODEL



Implementation

- Apply coding rules that implement secure design
- Use automated code review techniques to find potential vulnerabilities components
  - Static Analysis
  - Symbolic execution

STATIC CODE ANALYSIS TOOL USAGE BY DEVELOPERS
(SAMPLE SIZE: 2119)



REBELLABS

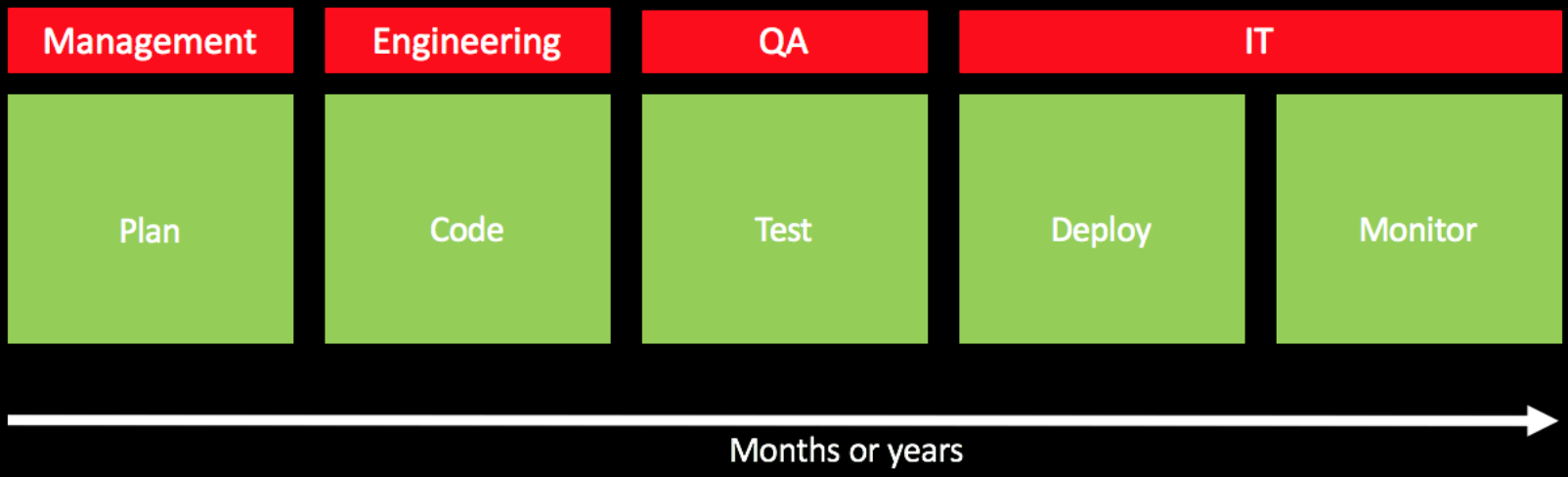
Testing

- Penetration Testing to find potential flaws in the real system
  - Fuzz testing
    - Employ attack patterns

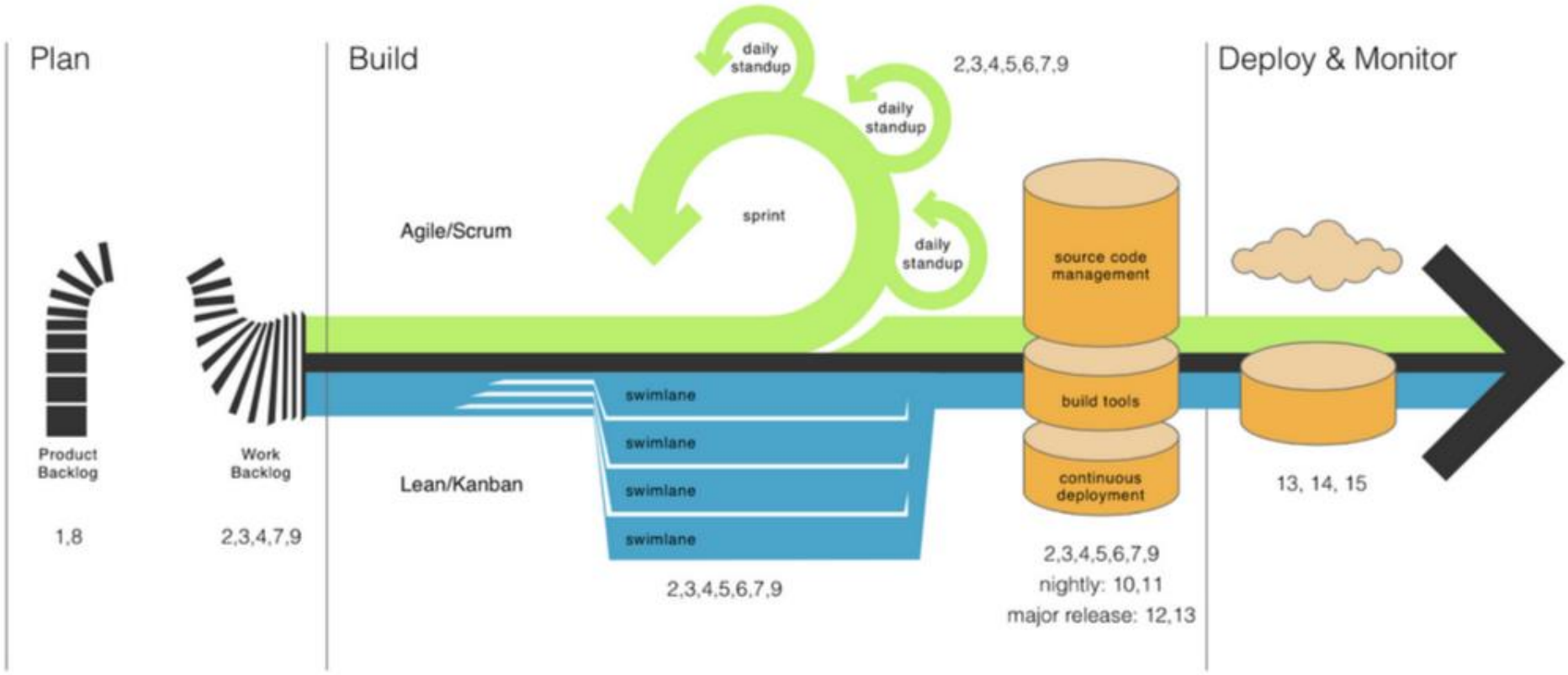
Different methodologies

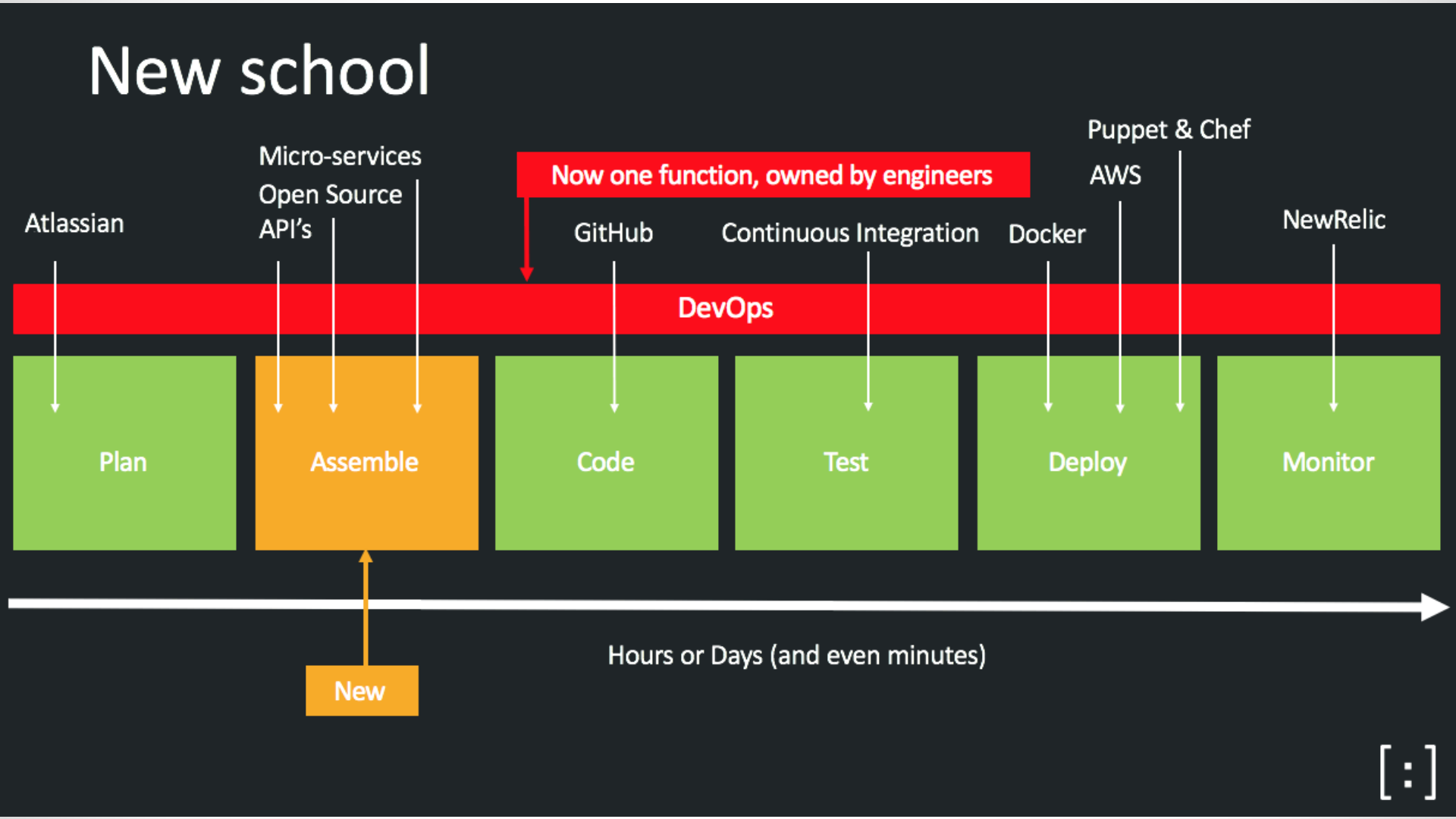
- BSIMM (Building Security In – Maturity Model)
  - http://bsimm.com
- Microsoft Security Development Lifecycle
  - https://www.microsoft.com/en-us/sdl/
- OpenSAMM Software Assurance Maturity Model
  - http://opensamm.org

Old School


Management
Engineering
QA
IT
Plan
Code
Test
Deploy
Monitor
Months or years

Continuous Delivery of Software




New school
Micro-services
Open Source
API's
Atlassian
Now one function, owned by engineers
GitHub
Continuous Integration
Docker
Puppet & Chef
AWS
NewRelic
DevOps
Plan
Assemble
Code
Test
Deploy
Monitor
Hours or Days (and even minutes)
New
[:]

Continuous Security

- Requires security automation
- Integrate into CD environment and tools
  - Source code management systems
    - GitHub, Bitbucket etc.
  - Build systems
    - Travis CI, Jenkins etc.
- Audit third party component and open-source library usage

Takeaways

- Security practices should be built in during the software development process
- Continuous delivery needs continuous security