Achieving Secure Continuous Delivery

Chris Rutter / Lucian Corlan
July 2016
Problem statement - Security

- Difficult access to (uncorrelated) vulnerability data
- No clear view on the security risk of a specific build or release
- No real agreed security gate (no trigger threshold)
- Product has a Roadmap and Security is (always) not (always) part of it

THERE IS NO RELEASE FOR A DAY
AND EVERYBODY LOOSES THEIR MINDS
Problem statement - Developers

- Security requirements appear when project is almost finished
- Security sign-off is a bottleneck
- When am I finally **secure enough**?
We’ve seen this before...

QA 5 years ago

- QA manual, at the end of a project
- JIRA tickets passed around for small bugs
- Long dev / test cycles
- Key dependencies for sign-off
- Lack of overview of quality or risk
Our Goals

• Security requirements identified early
• Viewed as true non-functional requirements
• Easy to fix issues detected and fixed within a sprint
• Security quality part of definition of done each sprint
• Security policy defined and automatically applied
• Ability to measure and track all of the above
• Pros: Security team have visibility and quality control of all testing

• Cons: Bottlenecks, Key dependencies, 1 monthly cycle, time cost, unclear sign-off criteria, manual reports / metrics
• Pros: Bottleneck reduced, High value threat modelling, shorter time to fix

• Cons: Reliance on static analysis, time consuming manual process, issues highlighted at end of sprint
#6 triggered by user Chris Rutter started 9 minutes ago

Total build time: 2 min 5 sec

- **Build**
  - Build Artefact
    - 9 minutes ago
    - 1 sec

- **Integration**
  - Deploy to Integration
    - 9 minutes ago
    - 1 sec
  - e2e Tests
    - 9 minutes ago
    - 1 sec

- **Security Scan**
  - Deploy to QA
    - 9 minutes ago
    - 1 sec
  - Fortify Scan
    - 9 minutes ago
    - 1 min 33 sec
  - Owasp Dependency Check
    - 7 minutes ago
    - 20 sec
• **Pros:** Issues highlighted quickly, multiple types of scan, defined policy under version control.

• **Cons:** Custom policy effort and maintenance, difficulty analysing risk from separate reports
Demo

Auto-Security-Pipeline

#14 triggered by user Chris Rutter started 5 minutes ago

Total build time: 2 min 6 sec

Build
- Build Artefact
  - 5 minutes ago
  - 1 sec

Integration
- Deploy to integration
  - 4 minutes ago
  - 1 sec
- e2e Tests
  - 4 minutes ago
  - 1 sec

Security Scan
- Deploy to QA
  - 4 minutes ago
  - 1 sec
- Fortify Scan
  - 4 minutes ago
  - 1 min 38 sec
- Owasp Dependency Check
  - 2 minutes ago
  - 29 sec

Sign-Off
- Security Policy Check
  - 2 minutes ago
  - 1 sec

OWASP
Open Web Application
Security Project
Pros: All scans & tests normalised in one place, mitigations and suppressions tracked, metrics available, devs / testers performing actives scans.

Cons: Dynamic scans manual or passive, lack of custom app attributes
YOU MAY BE COOL

BUT YOU’LL NEVER BE LEOPARD FUR COVERED MOTORCYCLE WITH MATCHING LEOPARD FUR ACCENTED LEATHER JACKET COOL
Automated dynamic scanning

- Donatello proxies e2e tests through ZAP for active scan mapping without crawling

Contextual risk policies – application passports

- Static & dynamic risk indicators based on Threat Modelling exercises and OWASP Top 10 and assign weight to risk indicators

- Integration with GRC tool
Contextual risk profiles

• Enhance Application criticality from ThreadFix
  
  • static attributes
    • PCI data involved
    • PII data involved
    • Exposure
    • New service?
    • User story review
    • Input filtering
    • Output encoding
    • 3rd party integration
    • Actively maintained
    • Transported data encryption
    • Non-repudiation or IP whitelisting
    • Security meter Defcon
    • Authentication
    • Randomness level

• Dynamic attributes
  • Number of user stories since last release
  • Number of user stories since last manual pentest
  • Number of Security User Stories (outcome of Threat Modeling)
Donatello / Threadfix

1 day commit / test cycle

- ZAP Scan
- Static Scan
- Dependency Check

Threadfix

Active Scans
Pen Tests

Threat Modelling

JIRA tickets for high-value threats

Donatello
Sources of inspiration

- Betfair Security solution & DevSecCon
- Proprietary API (python or node.js) hooking into all the tools, plus static attributes and interpretation of results per application in Gitlab
- Job in the continuous delivery tool to run the calculation (per build)
- Dashboard for metrics
