OWASP
AppSec EU
Belfast
8-12 May, 2017

Securing the CI

Irene Michlin, Principal Security Consultant

People, process, and technology

Step 1: make sure your CI does not harm your security

Step 2: only then it can be used to improve your security

This is not "tools" talk

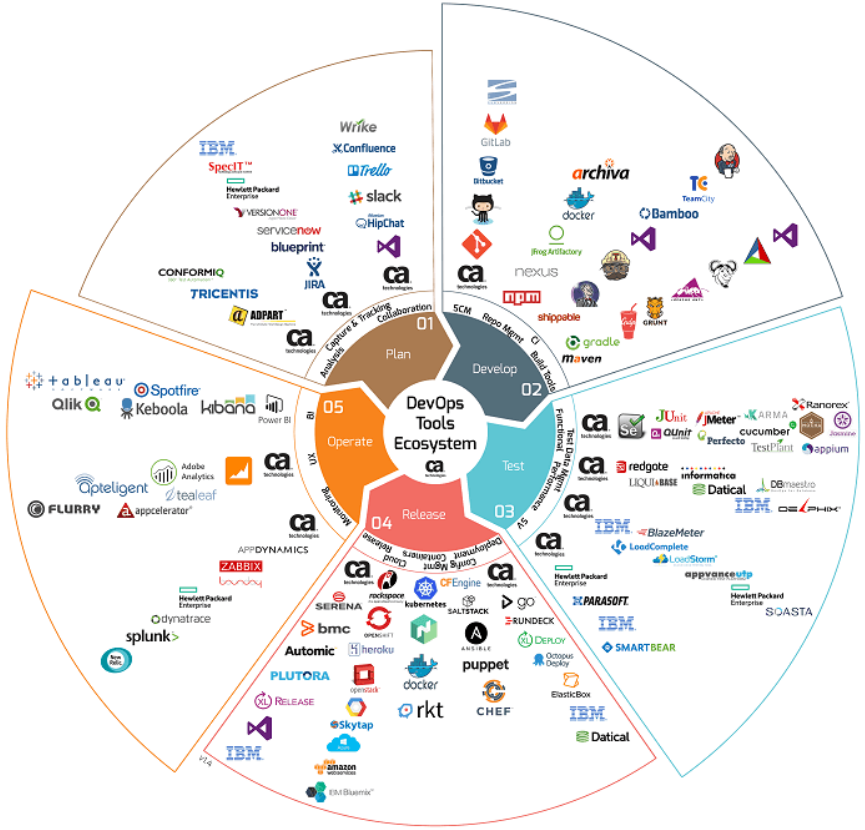
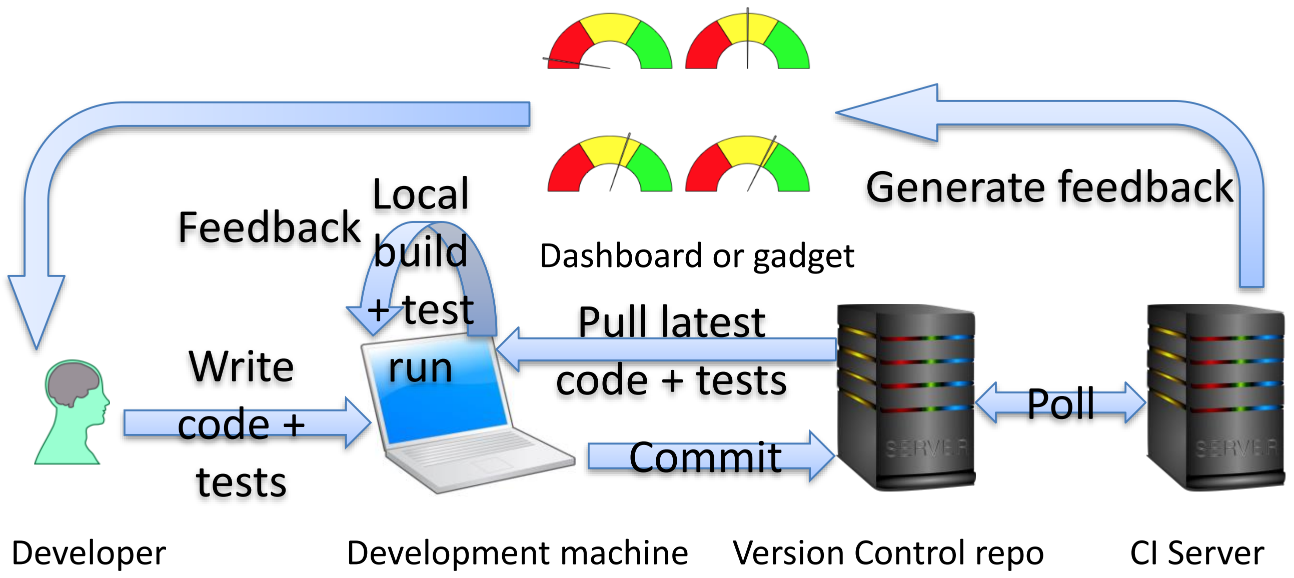


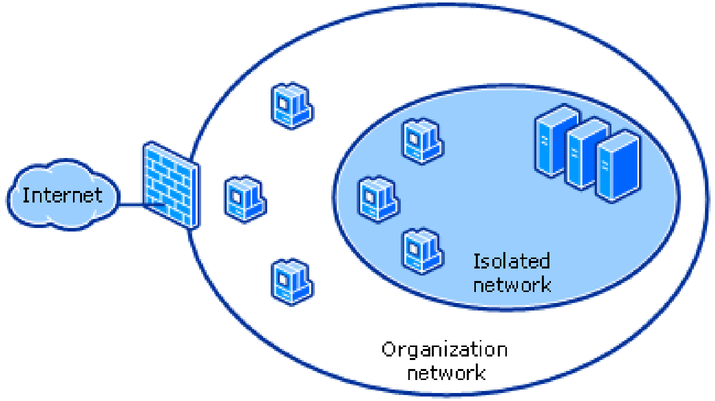
Image from https://www.blazemeter.com/blog/ultimate-devops-tools-ecosystem-tutorial-part-1

Basic CI cycle



Isolate your environment

- Phishing link in email => keylogger installed => source code gone (or backdoor deployed)
- Experimenting with development network => accounting department affected before EOY
- Extra challenges: remote work or BYOD



Version control server

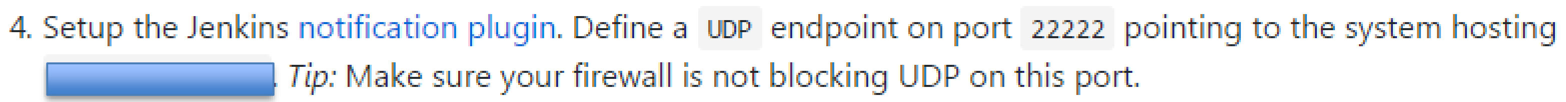
- It has one job only – remove or disable everything else
- No shared or generic accounts
- Matching business process to close accounts

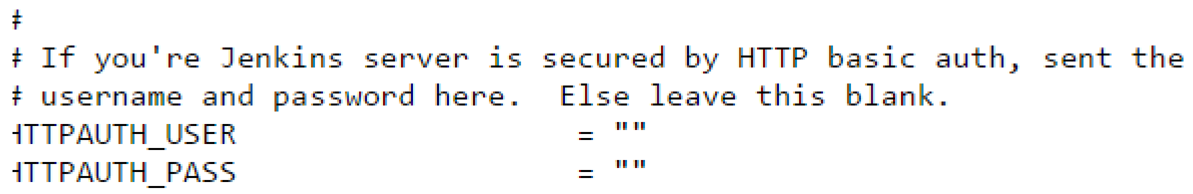
Integration build server

- Who is responsible for keeping it up to date?
- Where do external components come from?
- Check vendor advice on compiler and linker options

Feedback mechanism

- IoT electronic toys are notoriously insecure
- Custom integration scripts - are you cutting corners?


4. Setup the Jenkins notification plugin. Define a `UDP` endpoint on port `22222` pointing to the system hosting [redacted]. *Tip:* Make sure your firewall is not blocking UDP on this port.


```
#
# If you're Jenkins server is secured by HTTP basic auth, sent the
# username and password here.  Else leave this blank.
ITTPAUTH_USER                    = ""
ITTPAUTH_PASS                    = ""
```

Do no harm

- Do not acquire CI components « by accident »
- Not everything is secure out of the box
- Dormant account today is an attacker-controlled account tomorrow

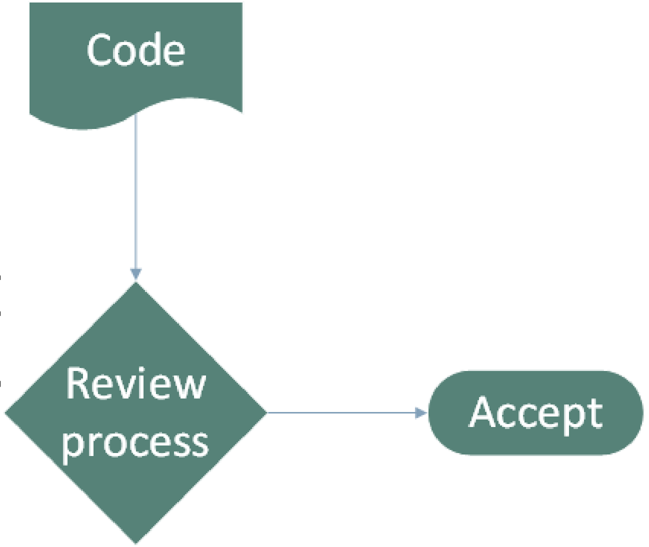
From dormant to active

CI Maturity model

	Novice	Beginner	Intermediary	Advanced	Expert
Build	Verification before commit Nightly build	CI build on commit Artefacts are managed	No build scripts – only configurations Dependencies are managed	Distributed builds Staged build sequence	Builds from VM CI server orchestrates VMs
Dev & QA	Unit Test Code Coverage	Metrics on tech debt & compliance Mock-ups & proxies	Code reviews Automated functional tests	Test Data Test in target	Automated acceptance test
SCM	Early branching Feature branches Rare merges	Late branching Release branches Merges are common	Pre-tested commits Integration branch is pristine	Branch by abstraction Feature toggle	Release notes & traceability analysis auto-generated
Visibility	Build status is notified to committer	Latest build status is available to all stakeholders	Trend reports Build status can be subscribed to	Real-time dashboards in work areas	Build reports & stats shared with customer and public
Security	RCA after bugs in the field reported Tracking security issues in bug DB	Coding standards CI build standard options Banned functions/ APIs Security features tested	Dynamic analysis on commit Security-driven test cases Signed binaries	Incremental threat modelling and attack surface review Fuzzing Static analysis	3rd party code inventory Customised tools Risk-based testing Bug bounty program

Code reviews

- No change too small
- Leave trivial checks to tools
- Not a separate task, but in DoI
- Reject & rework is part of "nor



Dev & QA	Unit Test Code Coverage	Metrics on tech debt & compliance Mock-ups & proxies	Code reviews Automated functional tests	Test Data Test in target	Automated acceptance test

Root-cause Analysis

- What happens to externally reported issues?
- The first security feedback to introduce
- What was missing in our CI process? => Improve

Security

RCA after bugs in the field reported
Tracking security issues in bug DB

Coding standards
CI build standard options
Banned functions/ APIs
Security features tested

Dynamic analysis on commit
Security-driven test cases
Signed binaries

Incremental threat modelling and attack surface review
Fuzzing
Static analysis

3rd party code inventory
Customised tools
Risk-based testing
Bug bounty program

Chain of custody

- Can you trust your release notes?
- Has every "unit of work" in the release gone through all the checks?
- Was it modified since "time of check"?

SCM

Early branching
Feature branches
Rare merges

Late branching
Release branches
Merges are common

Pre-tested commits
Integration branch is pristine

Branch by abstraction
Feature toggle

Release notes & traceability analysis auto-generated

“On commit” is great

- Automated coding standards checks
- Code complexity / code duplication
- Banned functions / APIs
- Dynamic analysis
- Static analysis
- Fuzzing
- https://www.nccgroup.trust/uk/our-research/securing-the-continuous-integration-process/

Points of contact

Irene Michlin
Principal Security Consultant

M: +44 (0) 7972 333 148
E: irene.michlin@nccgroup.trust
T: @IreneMichlin

Thank You to Our Sponsors

OWASP AppSec EU Belfast

CME Group

SYNOPSYS®

Hewlett Packard Enterprise

ARXAN

BLACKDUCK

CHECKMARX

CITRIX®

FLEXERA SOFTWARE

kiuwan

netsparker®

POSITIVE TECHNOLOGIES

RAPID7

Sonatype

Twistlock

Invest Northern Ireland

INTUITION TechChannels

jscrambler

promon

WhiteHat SECURITY

MCSGROUP INTELLIGENT RECRUITMENT SOLUTIONS

edgescan

Invest Northern Ireland

Allstate. You're in good hands.