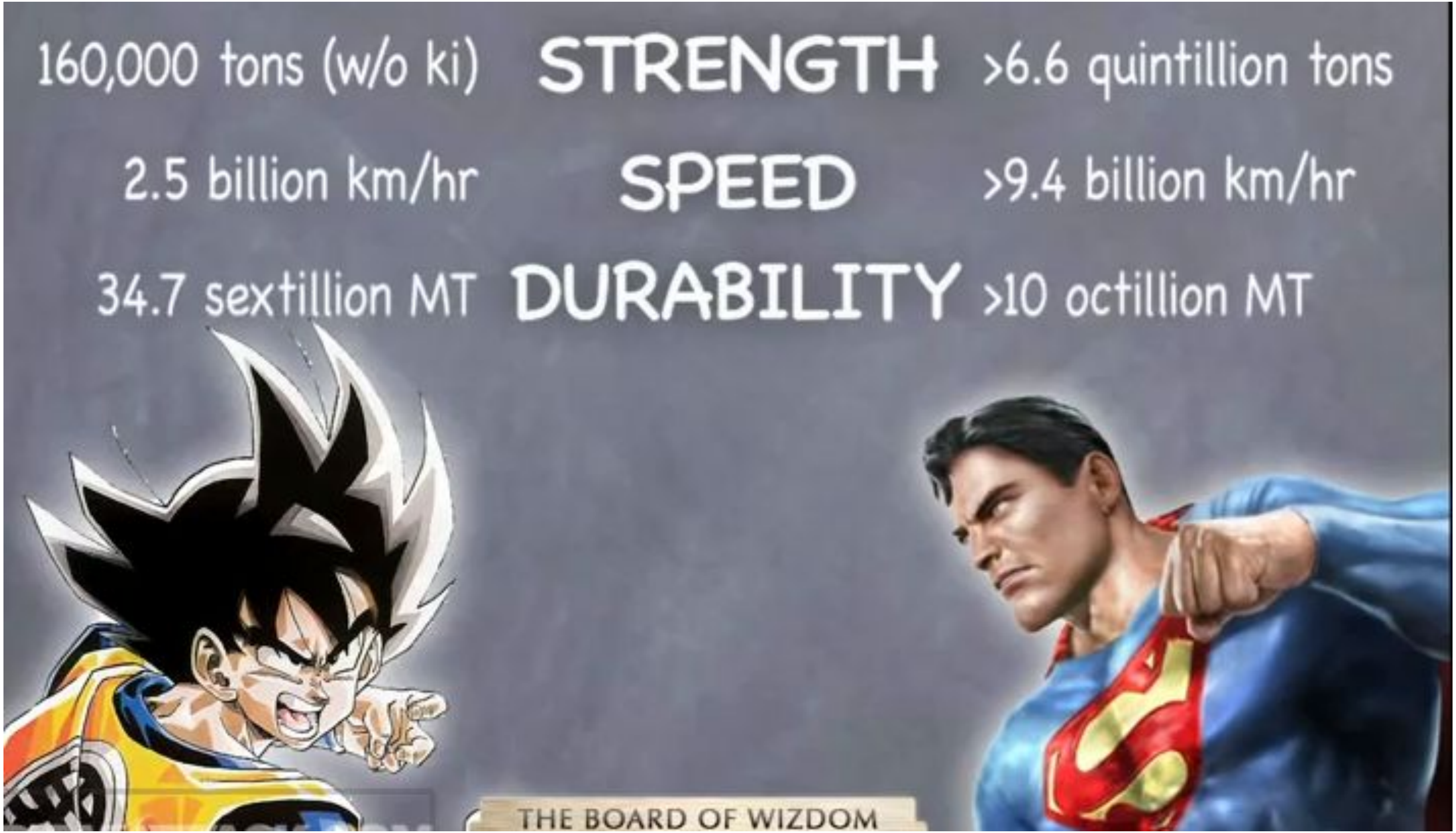


160,000 tons (w/o ki)	STRENGTH	>6.6 quintillion tons
2.5 billion km/hr	SPEED	>9.4 billion km/hr
34.7 sextillion MT	DURABILITY	>10 octillion MT

DevOps

A History in Configuration Management

About me

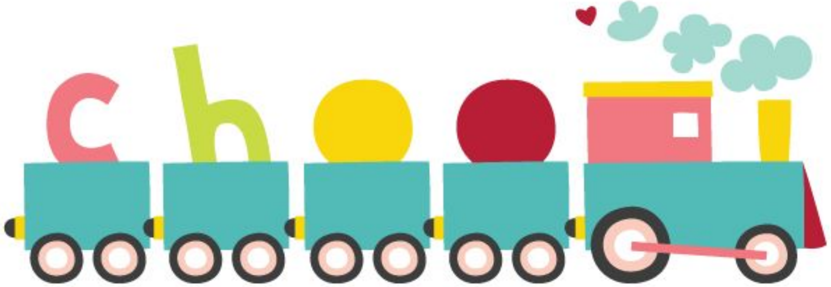
Senior Information Security Architect @ Epigen Technology

Security nerd & avid lock picker

Auditor, Analyst, Engineer

Organizer / Volunteer various conferences

Tech policy & tech literacy



Who we are...

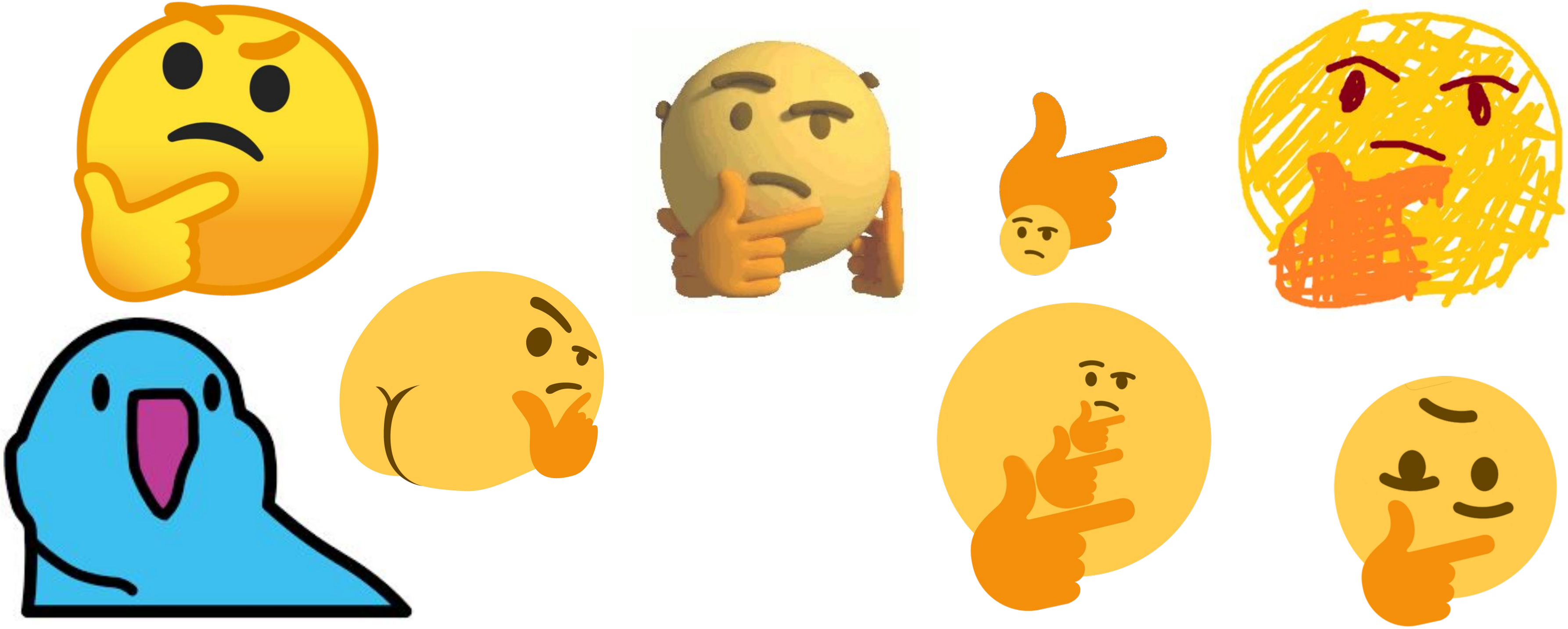
- Team of Senior Architects
- Trusted advisors to technology executives
- Chairing culture development within an organization
- Humans have to be involved in what we do
- Security minded DevOps
- Knowing when weaknesses are introduced to systems
- Understanding and education on scan results
- Identifying underlying issues to solve multiple problems
- It's ok to refactor

epigen technology

Agenda

Buncha stuff in maybe the adequate time

What is Configuration Management?



@apporima

What is Configuration Management?

...the practice of handling changes systematically so that a system maintains its integrity over time.

Configuration management embodies two concepts:

1. *the configuration management of items and their defining technical requirements and design documents, referred to herein as configuration documentation; and*
2. *the application of CM principles to digital data in general.*

MIL-HDBK-61 / MIL-HDBK-61A / MIL-HDBK-61B

What is Change Management?

1. *procedures are employed to systematically evaluate each proposed engineering change or*
2. *requested deviation to baselined documentation, to assess the total change impact (including costs) through*
3. *coordination with affected functional activities, to disposition the change or deviation and provide timely approval or*
4. *disapproval, and to assure timely implementation of approved changes by both parties.*

MIL-HDBK-61 / MIL-HDBK-61A / MIL-HDBK-61B

Where does CM come from?

Enter Clarence “Kelly” Johnson

Be Quick, Be Quiet, And Be On Time

1. The team leader must be an effective buffer
2. The team must be collocated in a small project office
3. Ruthlessly minimize the team size
4. Prototype quickly
5. The team must be trusted by company management and the customer
6. Restrict access to outsiders
7. Involve people in the big picture

Yoram Solomon

Summarized; 14 rules couldn’t fit

Undocumented 15th Rule 😱

Starve before doing business with the damned Navy.

They don't know what the hell they want and will drive you up a wall before they break either your heart or a more exposed part of your anatomy.

Ben Rich

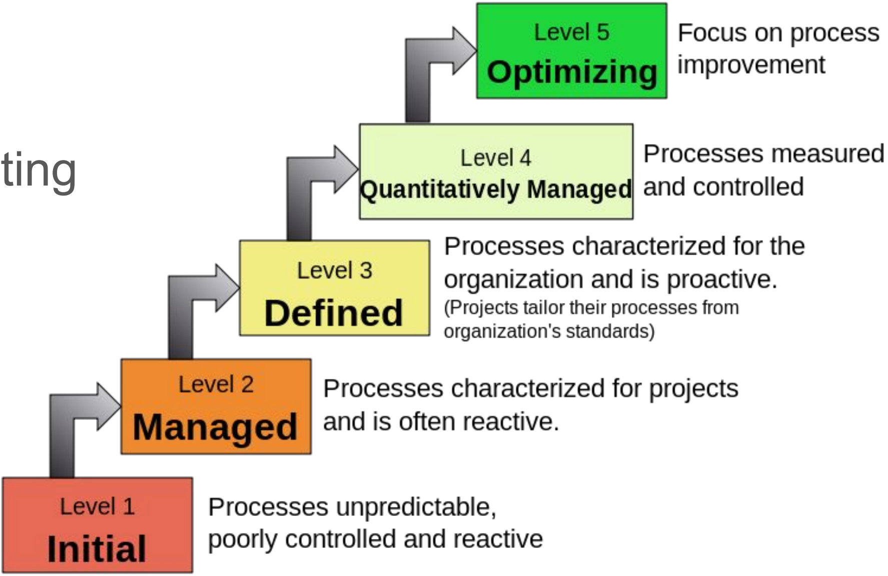
Skunk Works: A Personal Memoir of My Years of Lockheed.

Carnegie Mellon: Capability Maturity Model

DOD began contracting

in the 1980s

Characteristics of the Maturity levels



Level 5 **Optimizing**: Focus on process improvement

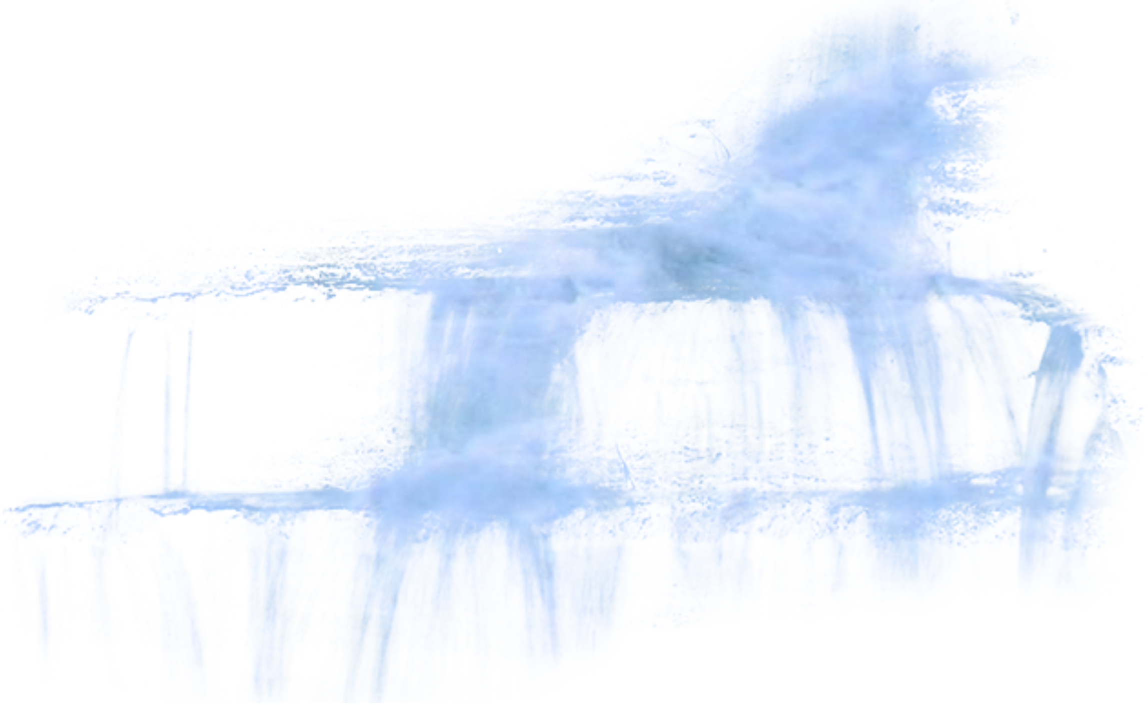
Level 4 **Quantitatively Managed**: Processes measured and controlled

Level 3 **Defined**: Processes characterized for the organization and is proactive. (Projects tailor their processes from organization's standards)

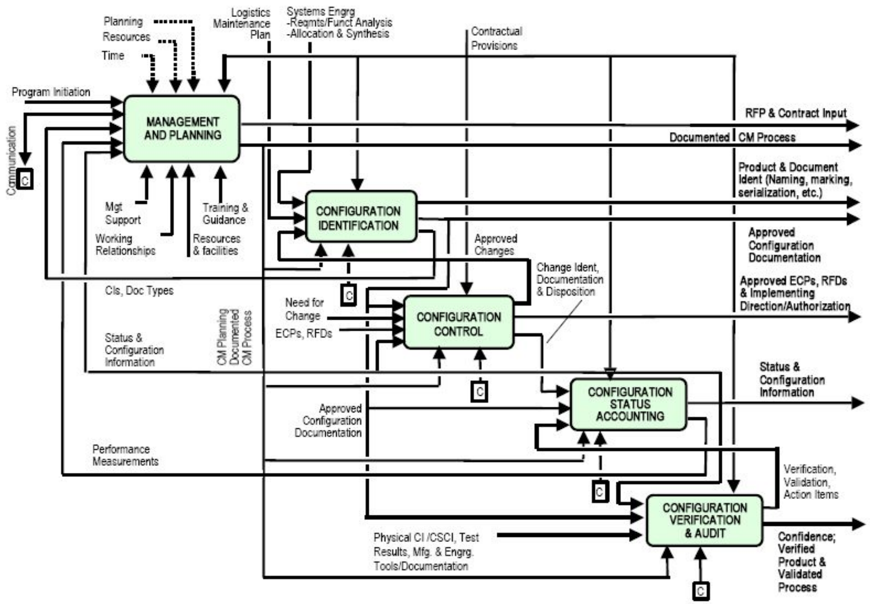
Level 2 **Managed**: Processes characterized for projects and is often reactive.

Level 1 **Initial**: Processes unpredictable, poorly controlled and reactive

Waterfall model



Configuration Management & ITIL



Planning
Resources
Time
Logistics Maintenance Plan
Systems Engrg
-Reqmts/Funct Analysis
-Allocation & Synthesis
Contractual Provisions
Program Initiation
MANAGEMENT AND PLANNING
RFP & Contract Input
Documented CM Process
Communication
C
Mgt Support
Training & Guidance
Working Relationships
Resources & facilities
CONFIGURATION IDENTIFICATION
Product & Document Ident (Naming, marking, serialization, etc.)
Approved Changes
Approved Configuration Documentation
Change Ident, Documentation & Disposition
CIs, Doc Types
C
Need for Change
ECPs, RFDs
CONFIGURATION CONTROL
Approved ECPs, RFDs & Implementing Direction/Authorization
Status & Configuration Information
CM Planning
Documented CM Process
C
CONFIGURATION STATUS ACCOUNTING
Status & Configuration Information
Approved Configuration Documentation
Performance Measurements
C
Verification, Validation, Action Items
CONFIGURATION VERIFICATION & AUDIT
Physical CI /CSCI, Test Results, Mfg. & Engrg. Tools/Documentation
Confidence; Verified Product & Validated Process
C

- Planning: Configuration Management Plan
- Identification: label artifacts for change
- Control: assurance of authorized artifacts
- Monitoring: tracking configuration items
- Verification: reviews and audits

MIL-HDBK-61 / MIL-HDBK-61A / MIL-HDBK-61B

ITIL: Configuration Management

@apporima

Agile: 16 Disciplines

- Adaptive software development (ASD)
- Agile modeling
- Agile unified process (AUP)
- Disciplined agile delivery
- Dynamic systems development method (DSDM)
- Extreme programming (XP)
- Feature-driven development (FDD)
- Lean software development
- Kanban
- Rapid application development (RAD)
- Scrum
- Scrumban

Rescue as a Service

@apporima

Agile: failed implementations

Have we lost sight of the mission and its business objectives?

Focused on how to avoid falling behind

Constant changing priorities ensuring everything is a critical issue

Creating new processes that bypass old processes creating process fatigue

People, Process, Tools

Enter DevOps: The Industry Response

@apporima

Configuration Management Evolved

1. SkunkWorks model
2. Carnegie Mellon Capability Maturity Model (CMM; CMM(I)ntegration)
3. Information Technology Infrastructure Library (ITIL)
4. Agile: 12 methods
5. Rugged DevOps
6. DevOps
7. DevSecOps
8. Rugged Enterprise DevLegalHRFinSecNetQAGovCustOps! (lol @nathenharvey)

Secure DevOps Toolchain

Pre-Commit

Security activities before code is checked in to version control

Threat Modeling/Attack Mapping:
- Attacker personas
- Evil user stories
- Raindance
- Mozilla Rapid Risk Assessment
- OWASP ThreatDragon

Security and Privacy Stories:
- OWASP ASVS
- SAFECode Security Stories

IDE Security Plugins:
- DevSkim
- FindSecurityBugs
- Puma Scan
- SonarLint

Pre-Commit Security Hooks:
- git-hound
- git-secrets
- Repo-supervisor
- ThoughtWorks Talisman

Secure Coding Standards:
- CERT Secure Coding Standards
- OWASP Proactive Controls

Manual and Peer Reviews:
- Gerrit
- GitHub pull request
- GitLab merge request
- Review Board

Commit (Continuous Integration)

Fast, automated security checks during the build and Continuous Integration steps

Static Code Analysis (SCA):
- FindSecurityBugs
- Brakeman
- ESLint
- Phan

Security Unit Tests:
- JUnit
- Mocha
- xUnit

Infrastructure as Code Analysis:
- ansible-lint
- Foodcritic
- puppet-lint
- cfn_nag

Dependency Management:
- OWASP Dependency Check
- Bundler-Audit
- Gemnasium
- PHP Security Checker
- Retire.JS
- Node Security Platform

Container Security:
- Actuary
- Anchore
- Clair
- Dagda
- Docker Bench
- Falco

Container Hardening:
- Bane
- CIS Benchmarks
- grsecurity

Acceptance (Continuous Delivery)

Automated security acceptance, functional testing, and deep out-of-band scanning during Continuous Delivery

Infrastructure as Code:
- Ansible
- Chef
- Puppet
- SaltStack
- Terraform
- Vagrant

Immutable Infrastructure:
- Docker
- rkt

Security Scanning:
- Arachni
- nmap
- sqlmap
- sslyze
- ZAP
- ssh_scan

Cloud Configuration Management:
- AWS CloudFormation
- Azure Resource Manager
- Google Cloud Deployment Manager

Security Acceptance Testing:
- BDD-Security
- Gauntlt
- Mittn

Infrastructure Tests:
- Serverspec
- Test Kitchen

Infrastructure Compliance Checks:
- HubbleStack
- InSpec

Production (Continuous Deployment)

Security checks before, during, and after code is deployed to production

Security Smoke Tests:
- ZAP Baseline Scan
- nmap
- ssllabs-scan

Configuration Safety Checks:
- AWS Config
- AWS Trusted Advisor
- Microsoft Azure Advisor
- Security Monkey
- OSQuery

Secrets Management:
- Ansible Vault
- Blackbox
- Chef Vault
- Docker Secrets
- Hashicorp Vault
- Pinterest Knox

Cloud Secrets Management:
- AWS KMS
- Azure Key Vault
- Google Cloud KMS

Cloud Security Testing:
- CloudSploit
- Nimbostratus

Server Hardening:
- dev-sec.io
- SIMP

Host Intrusion Detection System (HIDS):
- fail2ban
- OSSEC
- Samhain

Operations

Continuous security monitoring, testing, audit, and compliance checks

Fault Injection:
- Chaos Kong
- Chaos Monkey

Cyber Simulations:
- Game day exercises
- Tabletop scenarios

Penetration Testing:
- Attack-driven defense
- Bug Bounties
- Red team exercises

Threat Intelligence:
- Diamond Model
- Kill Chain
- STIX
- TAXII

Continuous Scanning:
- OpenSCAP
- OpenVAS
- Prowler
- Scout2
- vuls

Blameless Postmortems:
- Etsy Morgue

Continuous Monitoring:
- grafana
- graphite
- statsd
- seyren
- sof-elk
- ElastAlert
- 411

Cloud Monitoring:
- CloudWatch
- CloudTrail
- Reddalert

Cloud Compliance:
- Cloud Custodian
- Compliance Monkey
- Forseti Security

Building a DevSecOps Program (CALMS)

Culture

Break down barriers between Development, Security, and Operations through education and outreach

Automation

Embed self-service automated security scanning and testing in continuous delivery

Lean

Value stream analysis on security and compliance processes to optimize flow

Measurement

Use metrics to shape design and drive decisions

Sharing

Share threats, risks, and vulnerabilities by adding them to engineering backlogs

Start Your DevOps Metrics Program

- Number of high-severity vulnerabilities and how long they are open
- Build and deployment cycle time
- Automated test frequency and coverage
- Scanning frequency and coverage
- Number of attacks (and attackers) hitting your application

First Steps in Automation

- Build a security smoke test (e.g., ZAP Baseline Scan)
- Conduct negative unit testing to get off of the happy path
- Attack your system before somebody else does (e.g., Gauntlt)
- Add hardening steps into configuration recipes (e.g., dev-sec.io)
- Harden and test your CI/CD pipelines and do not rely on developer-friendly defaults

Learn to build, deliver, and deploy modern applications using secure DevOps and cloud principles, practices, and tools.

DEV540: Secure DevOps and Cloud Application Security

www.sans.org/DEV540

SANS AppSec
APPLICATION & SOFTWARE SECURITY

SANS APPSEC CURRICULUM

PLATFORM SECURITY

- DEV531 Defending Mobile Applications Security Essentials
- DEV541 Secure Coding in Java/JEE *GSSP-JAVA*
- DEV544 Secure Coding in .NET *GSSP-NET*

CORE

- STH.DEVELOPER Application Security Awareness Modules
- DEV522 Defending Web Applications Security Essentials *GWEB*
- DEV534 Secure DevOps: A Practical Introduction
- DEV540 Secure DevOps and Cloud Application Security

SPECIALIZATION

- SEC542 Web App Penetration Testing and Ethical Hacking *GWAPT*
- SEC642 Advanced Web App Penetration Testing, Ethical Hacking, and Exploitation Techniques

ASSESSMENT

- AppSec CyberTalent Assessment sans.org/appsec-assessment

Poster contributors:
- Ben Allen
- Jim Bird
- David Deatherage
- Mark Geeslin
- Eric Johnson
- Frank Kim
- Jason Lam
- Gregory Leonard
- Dr. Johannes Ullrich

SANS

@apporima

Takeaways

- Having sight of the objectives
- Understanding where the things come from
- Linear Frameworks
- Identifying organizational trauma
- Ensure organizational integrity
- Ensure organizational security
    - Sustained team communication
    - Information management

Successful executions are key to implementation

@apporima

Question?