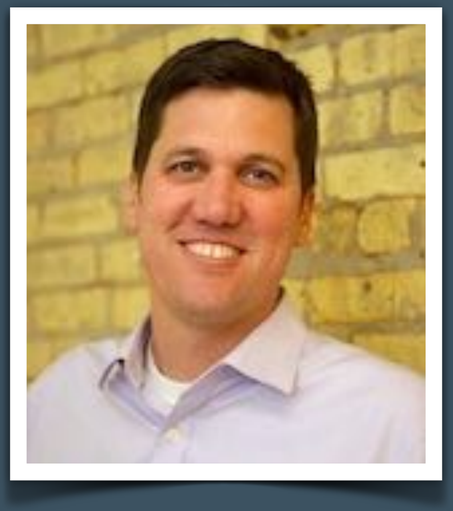
Security as Code:
A SecDevOps Use Case

Ed Bellis, Kenna Cofounder & CTO

DOES15 October 21,2015

About Me



Ed Bellis: Kenna Cofounder & CTO | @ebellis

★Cofounder & CTO of Kenna

★Former CISO of Orbitz, VP InfoSec at Bank of America

★Recognized Security Expert & Evangelist: Black Hat, OWASP, Gartner, IANS, SaaScon, InfoSec World & SecTor

★Contributing Author: Beautiful Security by O'Reilly Media; Writer/ Blogger: CSO Magazine, CSO Online, InfoSec Island

kenna

Are You In The Right Room?

- InfoSec Value of Automation
- Automation Top Concerns
- Applying SecDevOps Principles at Kenna via Automation
- Security Automation Use Case
- Q & A

Security Value
or
“why are we doing this?”

To Those Who Came Before Us

We Salute You

Justin Collins, Neil Matatall & Alex Smolen from Twitter

kenna

Deploy Smaller Changes, More Frequently

- Decouple feature releases from code deployments
- Deploy features in a disabled state, using feature flags
- Require all developers to check code into trunk daily
- Integrate security testing at each code check
- The result is code always being in a secure, deployable state
- Practice deploying smaller changes, which dramatically reduces risk and improves MTTR

kenna

Source: http://www.facebook.com/note.php?note_id=14218138919

Inject Failures Often

The Netflix Tech Blog

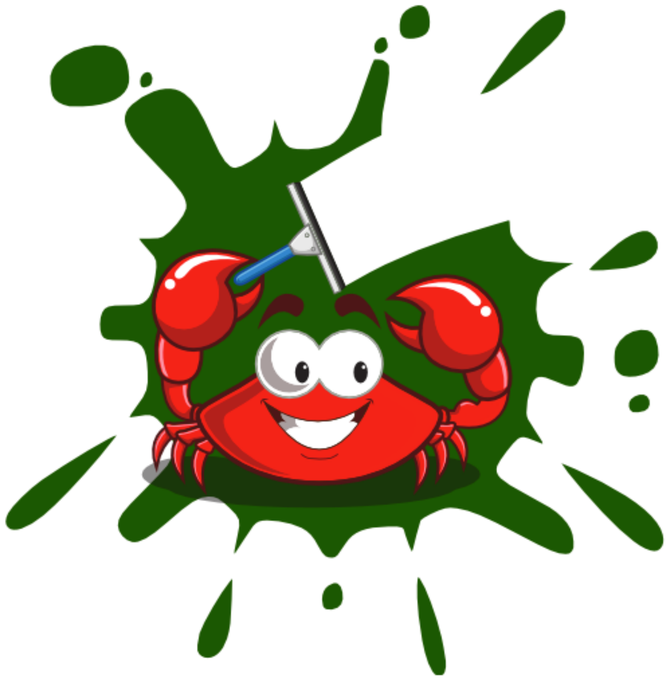
Five Lessons We've Learned Using AWS

We've sometimes referred to the Netflix software architecture in AWS as our Rambo Architecture. Each system has to be able to succeed, no matter what, even all on its own. We're designing each distributed system to expect and tolerate failure from other systems on which it depends.

One of the first systems our engineers built in AWS is called the Chaos Monkey. The Chaos Monkey's job is to randomly kill instances & services within our architecture. If we aren't constantly testing our ability to succeed despite failure, then it isn't likely to work when it matters most - in the event of an unexpected outage.

kenna

With a Little Help from our Friends (at Netflix)



kenna

Break Things Before Production

- Enforce consistency in code, environments and configurations across the environments
- Add your ASSERTs to find misconfigurations, enforce https, etc.
- Add static code analysis to automated continuous integration and testing process

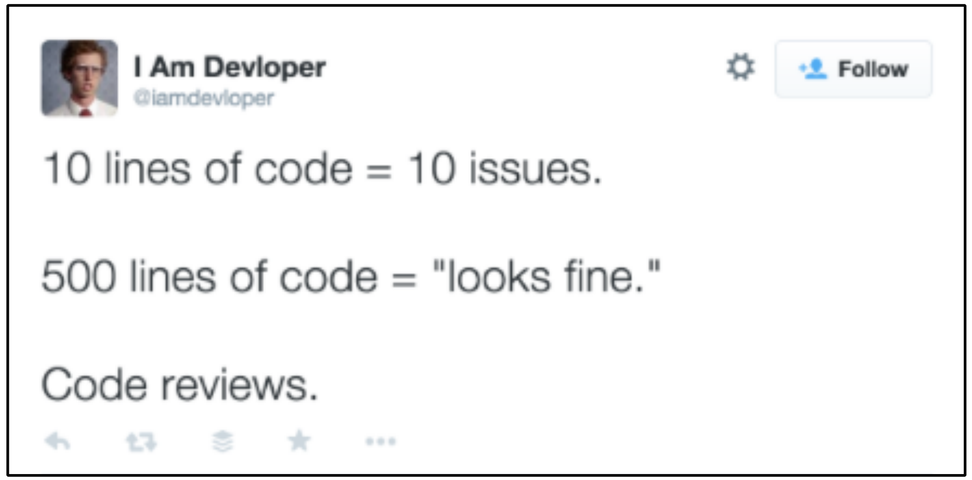
SecDevOps At Kenna
. . .or. . .
“what every CISO should know”

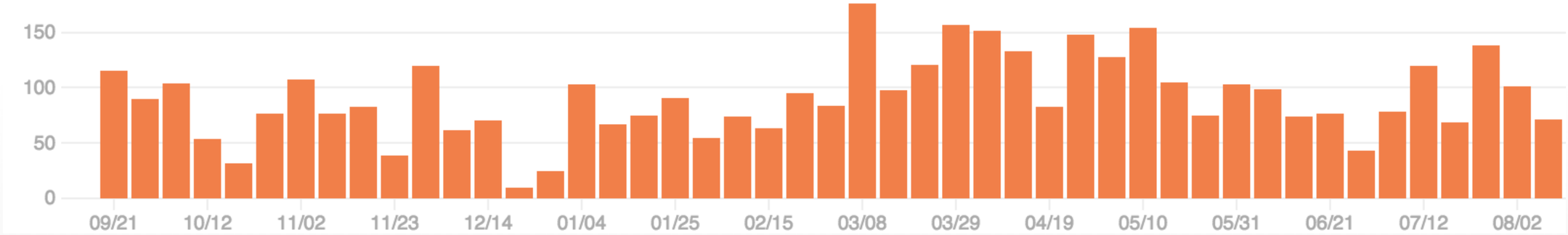
Small, Frequent Changes ARE your security friends.

By the Numbers

Small & Frequent Commits

- Average between 50 & 150 commits commits to Master/week
- Simplicity is your friend

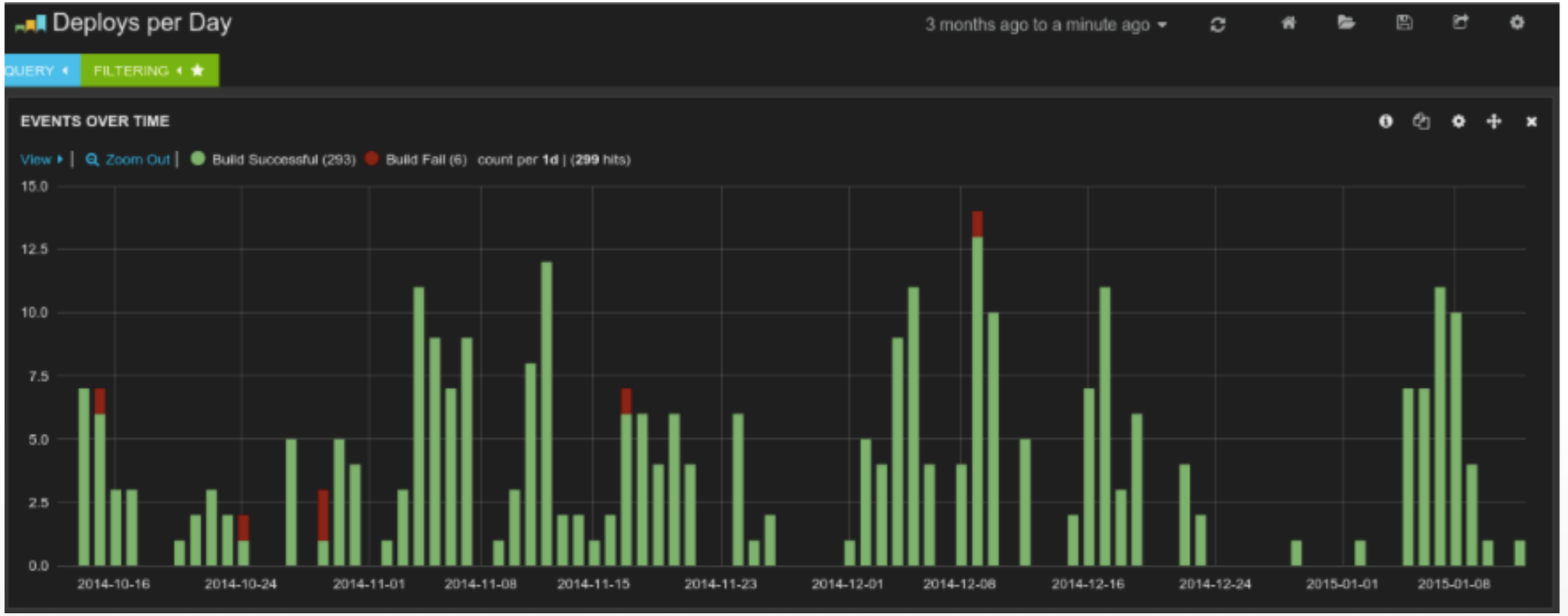




kenna

By the Numbers

Always Be Deploying (EVERY day, MANY times per day)



kenna

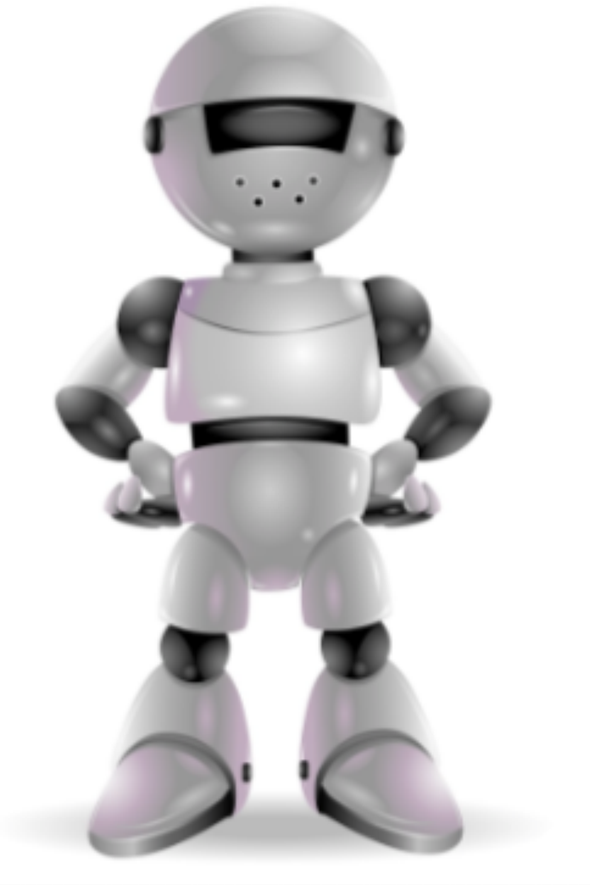
Static Security Requirements Are Stuck In the Stone Ages

Security Automation

Chef All the Things!

Open-Sourced Cookbooks

ModSecurity	Nessus	Nmap	SSH
iptables	encrypted volumes	Duo 2FA	openVPN



Test All the Things! (including security)

Static + Dynamic Throughout

Continuous Integration via CircleCI

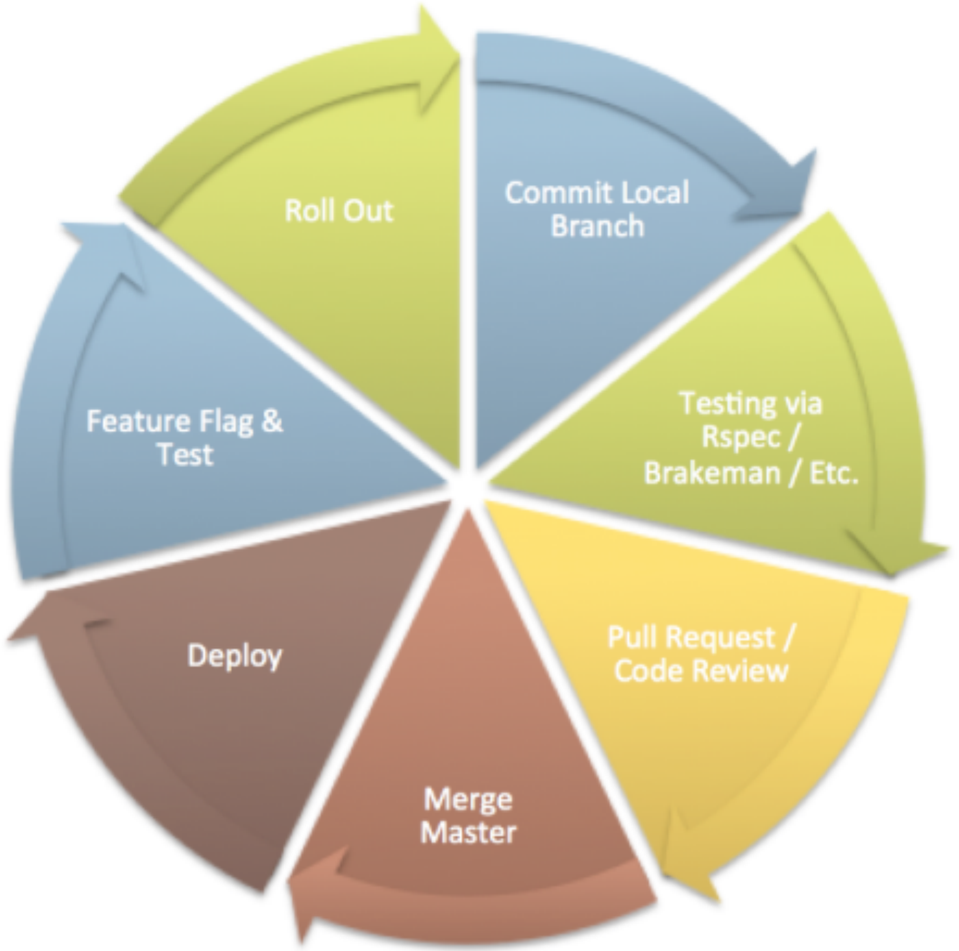
ChatOps = Slack + graphite + logstash + sensu + pagerduty

kenna

A SecDevOps Use Case

Continuous Security Testing

- Brakeman Static Analysis
- Dynamic Application Scanning
- Paid Penetration Testing Service
- Open Bug Bounty Program
- Continuous Infrastructure Scanning



All results Loaded into Kenna via API

kenna

DevOps as a Compliance Enabler

Automation as Evidence & Doc

Cookbooks

Leveraging the ELK Stack

Elasticsearch

Logstash

Kibana

Github + Code Climate + Dogfood

Compliance Automation Extra Credit: https://telekomlabs.github.io/

kenna

API’s For Your Protection

The Kenna API

EndPoints:

Vulnerabilities

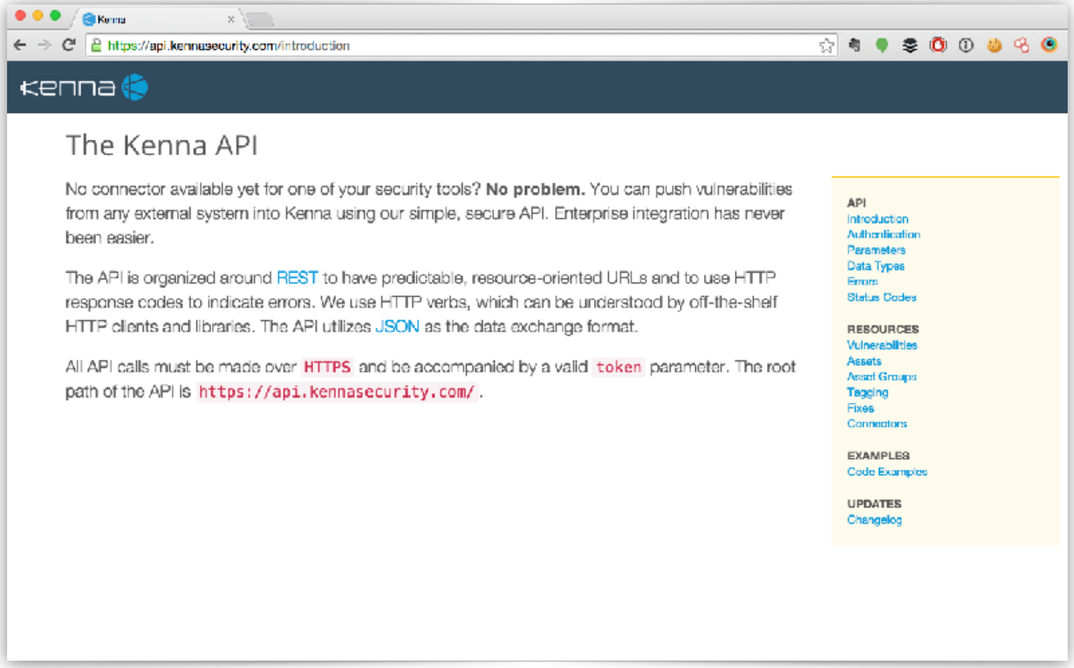
Assets

Tagging (metadata)

Patches

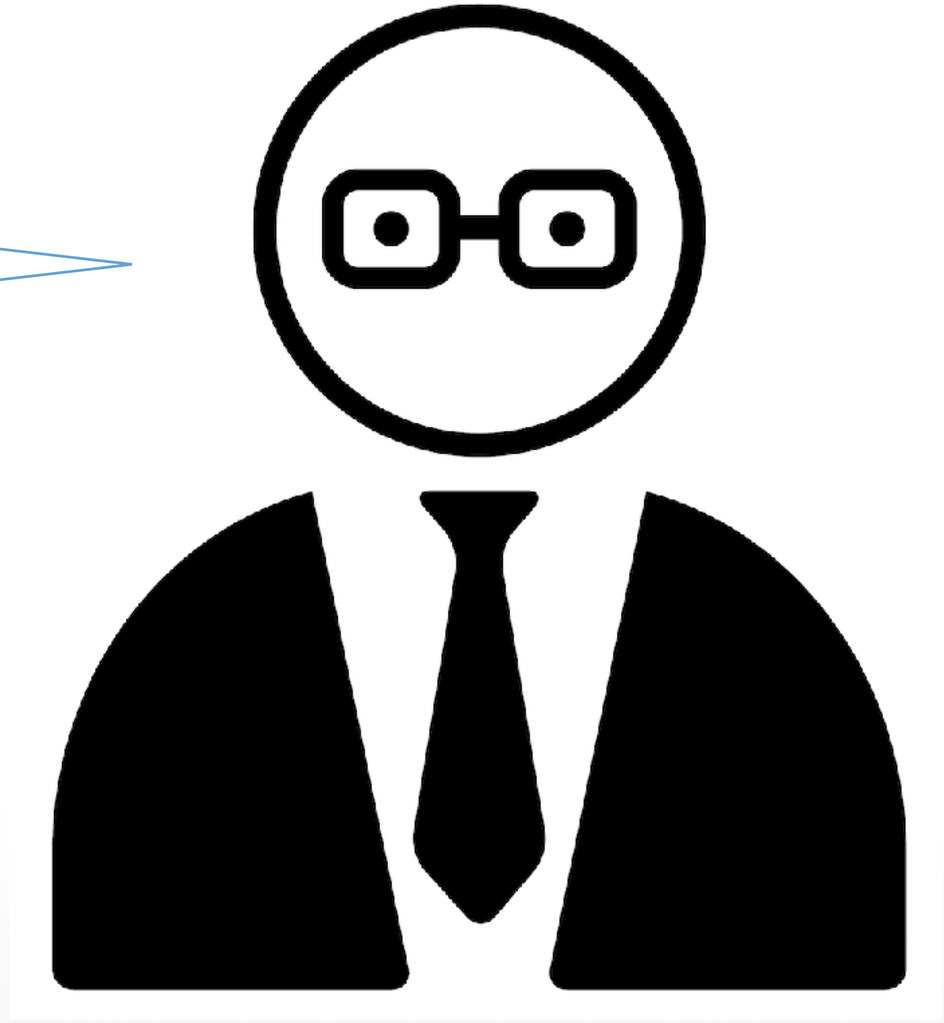
Connectors (integration & scanner control)

Threat Processing & Risk Meters



kenna

But what about segregation of duties?



kenna

There's a DevOps for that

Dev checks in code

Ops team deploys to QA/Staging

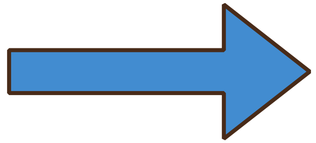
QA Tests

Source & Results are documented

Management Sign Off

Deploy to Production

Fix or Rollback



Dev checks in code

Automated Tests (including security)

Pull Request / Code Review

Continuous Integration / Deployment

"Pull requests are the new segregation of duties."

kenna

Experiment: 626

Because Auditors

kenna

Our Auditor Told Us “you must scan”

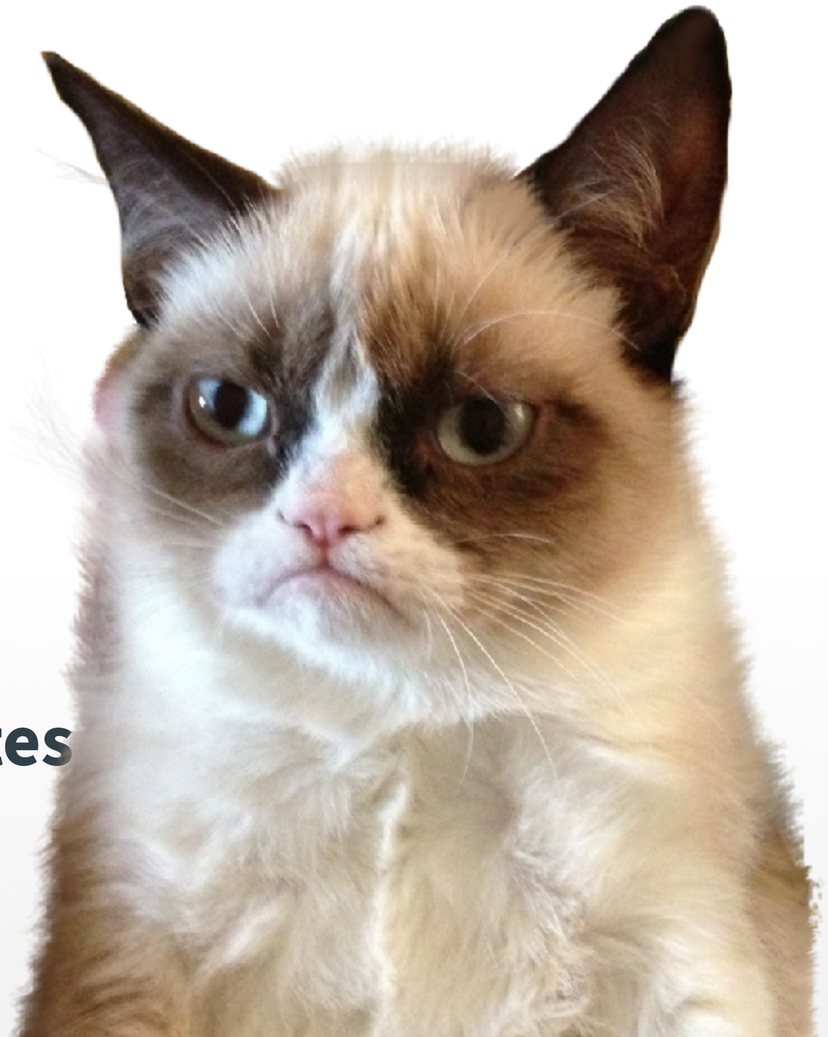
Can Be Slow

Can Be Intrusive

Requiring Scanning during Maintenance Windows :(

A Lot Can Happen In A Week

Depends On Timely Vulnerability Definition Updates



kenna

What About Ours?

1. **Built On Automation**
2. **Made of Composable Ingredients**
3. **It Should be Non-Intrusive**
4. **It Should be Close to Real-Time**
5. **It Should Support Multiple Sources for Definitions**
6. **It Should Only Take ~3.14 Days to Create**

(have I mentioned it's NOT a scanner)

Our Base Ingredients

Automation

(Chef, Ansible, Docker + stuff, cron + SSH)

Reliable Software + Version Info

Vulnerability Definitions

NVD/CVE feed with CPE included

“cpe:/a:google:chrome:8.0.552.215" is vulnerable to something

Eggs, Milk, Flour and Salt

These are not security related

kenna

Gather host: software+version

Automation Framework + Scripting

we used chef + tattle

XYZ Monitoring Services

opsmatic, logstash, splunk…

System Packager

rpm, dkpkg, pacman…

Dockerfiles

kenna

What is tattle?

Ridiculously simple way to store and regurgitate data

tattle update software myapp --version 1.2.3 --installer ubuntu

tattle report software

> {“myapp”:{“version":"1.2.3","installer":"ubuntu"}}

 # on github soon

Vulnerability Definitions

NVD CVE Feed with CPE data

https://nvd.nist.gov/download.cfm

<entry id="CVE-2011-0001">

...<vuln:cve-id>CVE-2011-0001</vuln:cve-id>

...<vuln:vulnerable-software-list>

<vuln:product>cpe:/a:zaal:tgt:1.0.1</vuln:product>

<vuln:product>cpe:/a:zaal:tgt:1.0.0</vuln:product>

...<vuln:summary>Double free vulnerability in ...

kenna

Mix It Together

Parse Some XML

$ cat nvdcve-2.0-2014.xml | ruby -e 'require "ox"; p Ox.parse(ARGF.read).nvd.locate("entry").select {|x| x.locate("/vuln:product/*").member? "cpe:/a:oracle:mysql:5.5.26" }.map(&:id)'*

["CVE-2014-0384", "CVE-2014-0386", "CVE-AWW-YEAH",...]

What was that?

- *We have mysql 5.5.26 installed*
- *Searching for it's most specific CPE locator*
- *We found that there are a bunch of Vulnerabilities for it from 2014*

kenna

What We Did

Extend Our Kenna API to Allow Software+Version (we already had the data anyway…)

```
curl -H "Content-type: application/json" \
https://api.kennasecurity.com/assets/63/software -X PUT -d "{
  ‘asset’: {
    ‘software’: $(tattle report software)
  }
}"
```

On Upload of Software Data

- *Update Assets & Create Associated Vulnerabilities*
- *Close Vulnerabilities for Software that’s been Updated*

kenna

Cake!

kenna

Icing on the Cake

Also Known As Next Steps…

- *Integrate Existing 0 Day Feeds*
- *Extend upgrade/uninstall tracking to build a timeline*
- *Alert when new Vulnerabilities match Assets*

(The Real-Time Thing)

The Cake Is A Lie

Reliability of CPEs for all data sources

needs further study

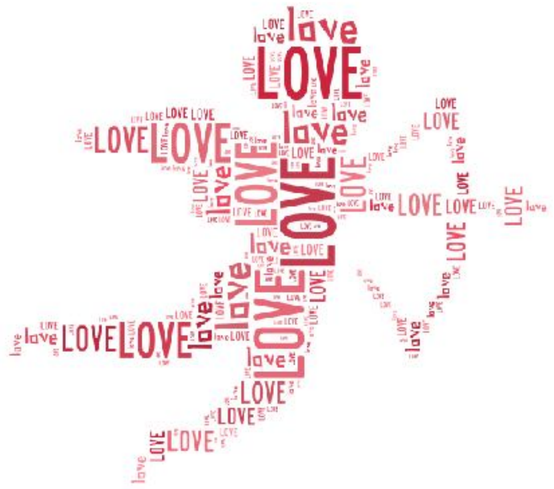
Vendor/Product/Version mismatches

dpkg says “zlib1g”, you say “zlib”

Packager whitelists / Packager specific CPEs

“Ubuntu fixed this with USN-1337-1”

In Summary



DevOps

❤️

InfoSec

kenna

Resources

Special Thanks to Gene Kim

- https://puppetlabs.com/2015-devops-report
- https://api.kennasecurity.com
- https://telekomlabs.github.io/
- https://github.com/jro/automated_security
- https://github.com/joeyschoblaska/brakeman-risk-io
- tattle on github - coming soon

kenna

@ebellis

We Are Hiring!
kennasecurity.com/jobs