Continuous Security in DevOps

Maciej Lasyk

4developers – Warsaw

2015-04-20

Join Fedora Infrastructure!

→ learn Ansible

→ join the security team!

→ use Fedora Security Lab (spin)

http://fedoraproject.org/en/join-fedora

Agenda?

→ DevOps indoctrination

→ technical infrastructure stuff

→ continuous delivery considerations

→ finally infosec tools & automation

→ working demo (hopefully) ;)

Agenda?

→ DevOps indoctrination

→ technical infrastructure stuff

→ continuous delivery considerations

→ finally infosec tools & automation

→ working demo (hopefully) ;)

Agenda?

→ DevOps indoctrination

→ technical infrastructure stuff

→ continuous delivery considerations

→ finally infosec tools & automation

→ working demo (hopefully) ;)

Agenda?

→ DevOps indoctrination

→ technical infrastructure stuff

→ continuous delivery considerations

→ finally infosec tools & automation

→ working demo (hopefully) ;)

Agenda?

→ DevOps indoctrination

→ technical infrastructure stuff

→ continuous delivery considerations

→ finally infosec tools & automation

→ working demo (hopefully) ;)

I'm not a security expert but an engineer

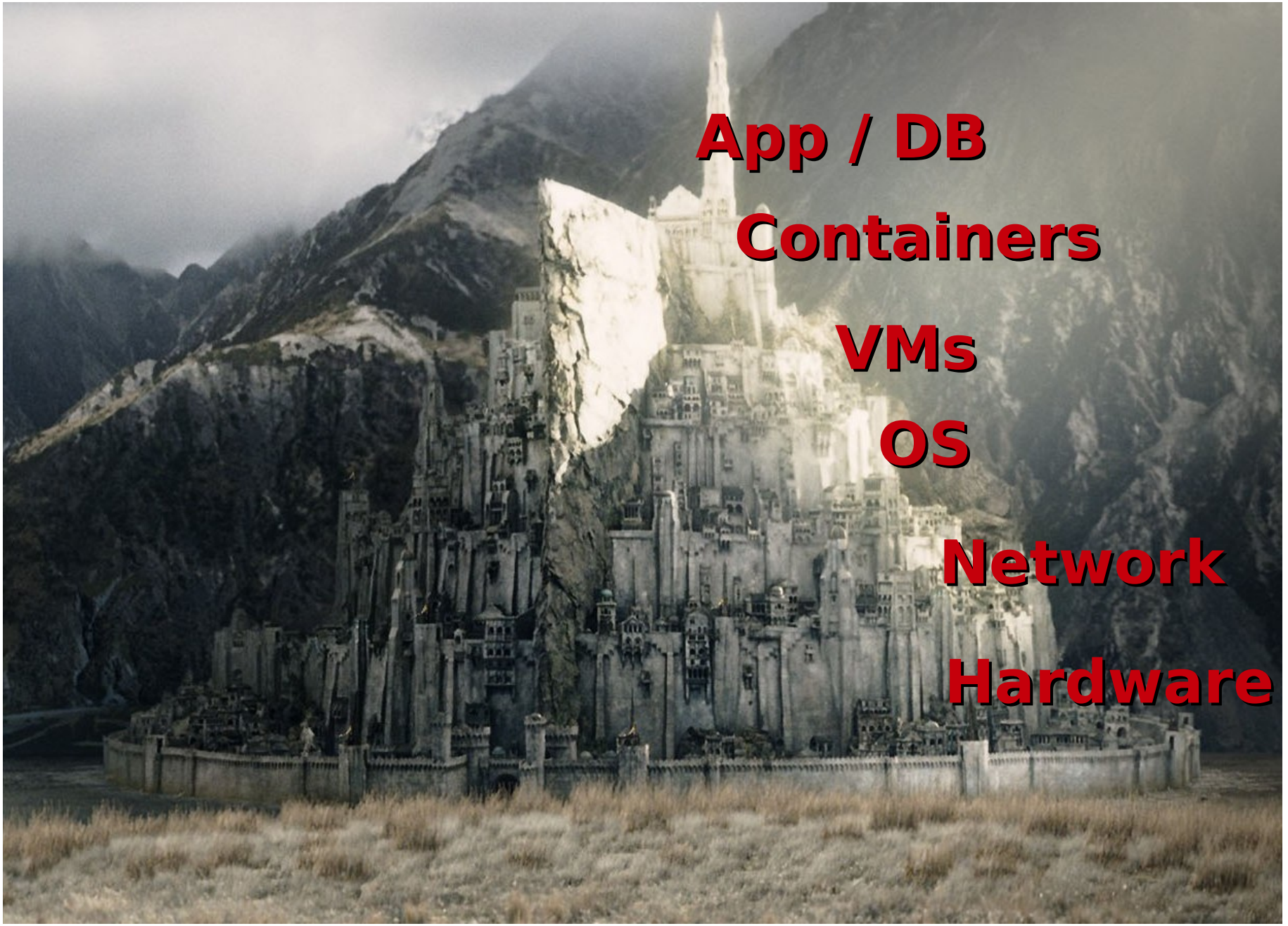
passionate about security & quality

“The only thing more dangerous than a developer is a developer conspiring with Security. The two working together gives means, motive and opportunity.”

“The Phoenix Project”

by Gene Kim, Kevin Behr and George Spafford

General security rule in IT: security is based on layers



General security rule in IT: security is based on layers

DevOps Anti-Types & patterns

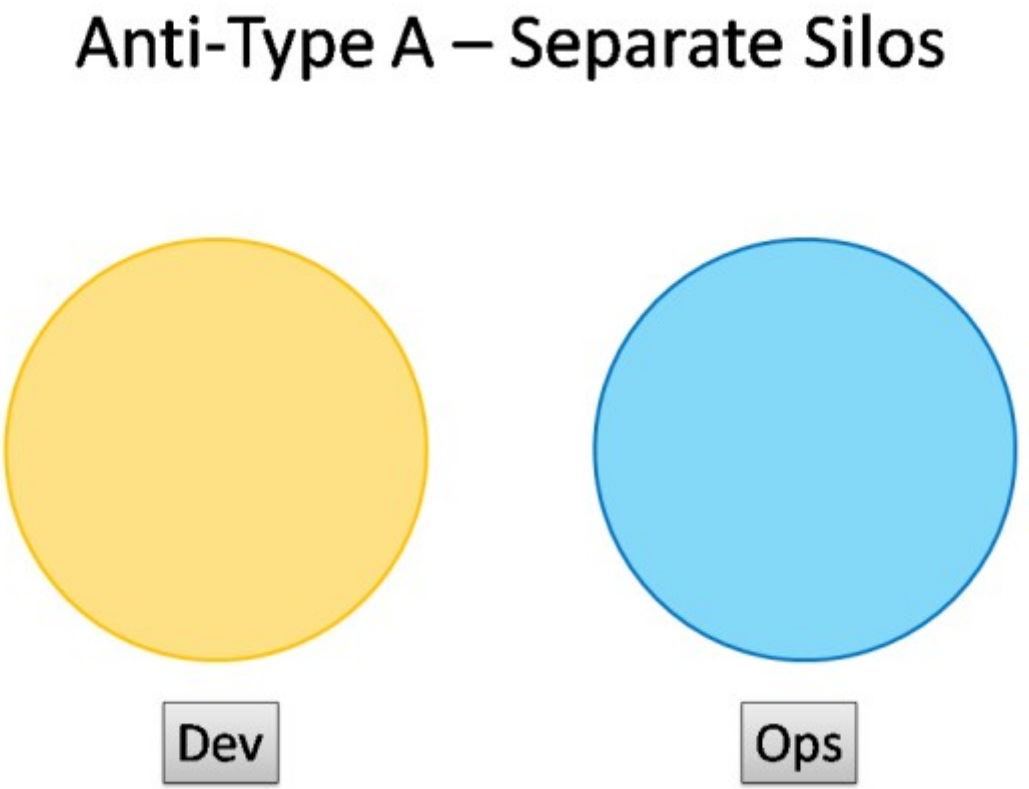
This is a copy/paste from

http://blog.matthewskelton.net/

w/my comments included and InfoSec layer added

Great job Matthew! Thanks!

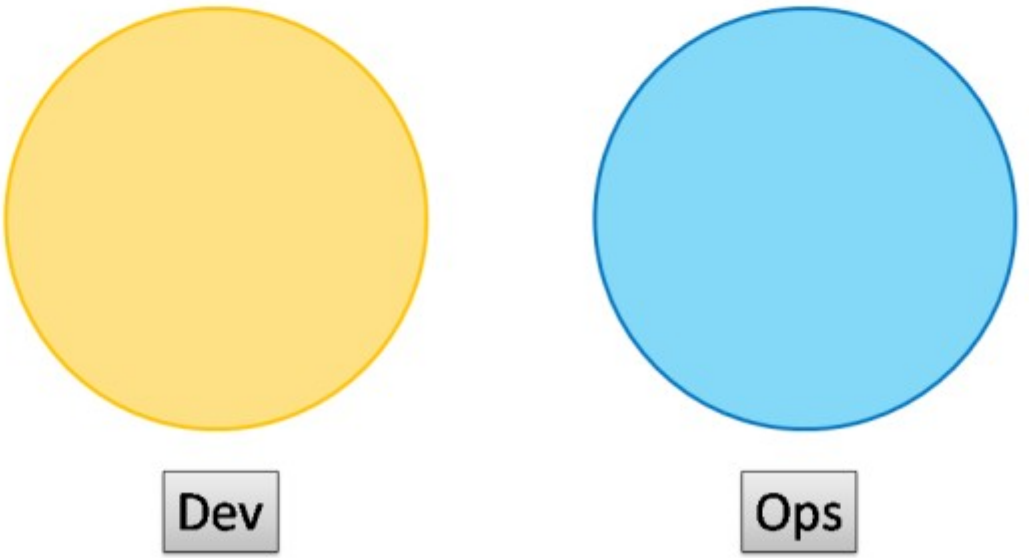
DevOps Anti-Types



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

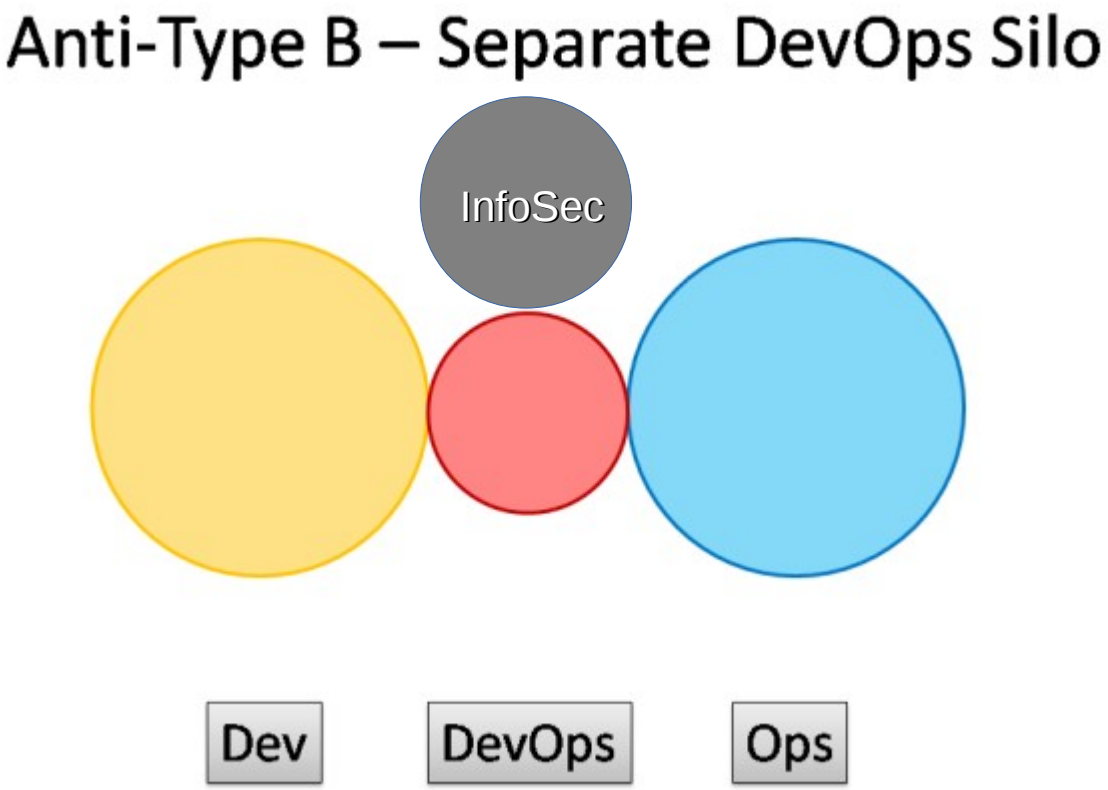
DevOps Anti-Types

Anti-Type A – Separate Silos



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

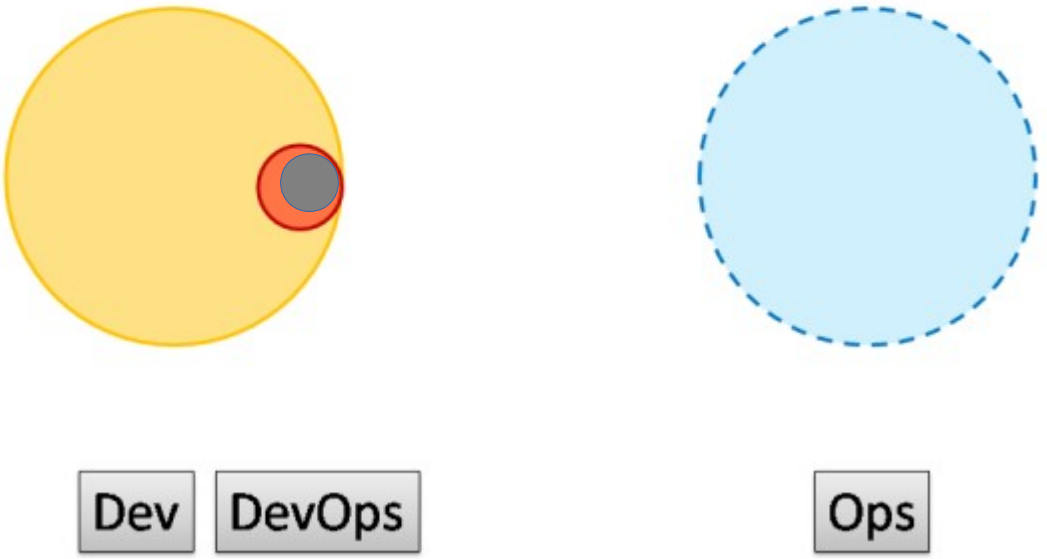
DevOps Anti-Types



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

DevOps Anti-Types

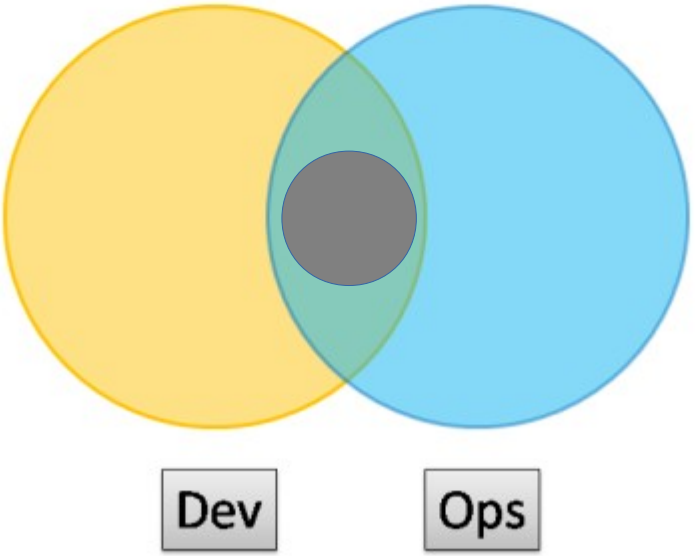
Anti-Type C – “We Don’t Need Ops”



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

DevOps Patterns

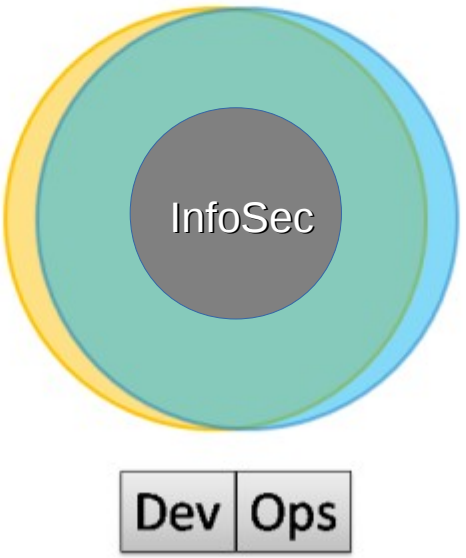
Type 1 – Smooth Collaboration



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

DevOps Patterns

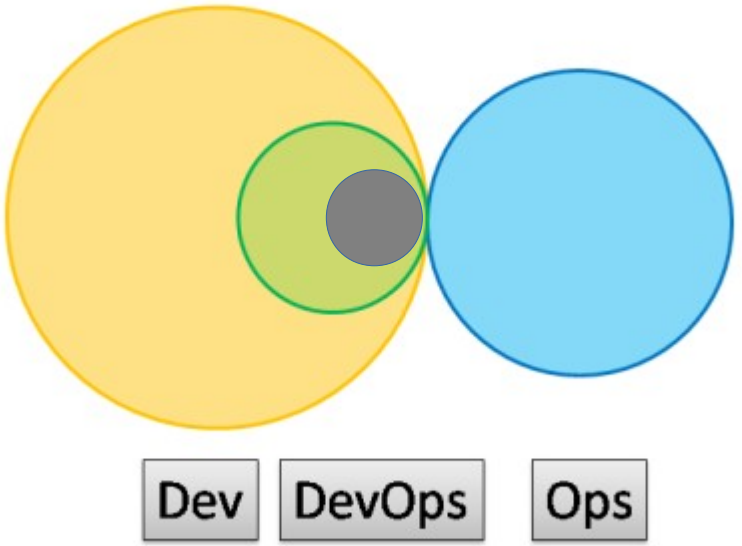
Type 2 – Fully Embedded



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

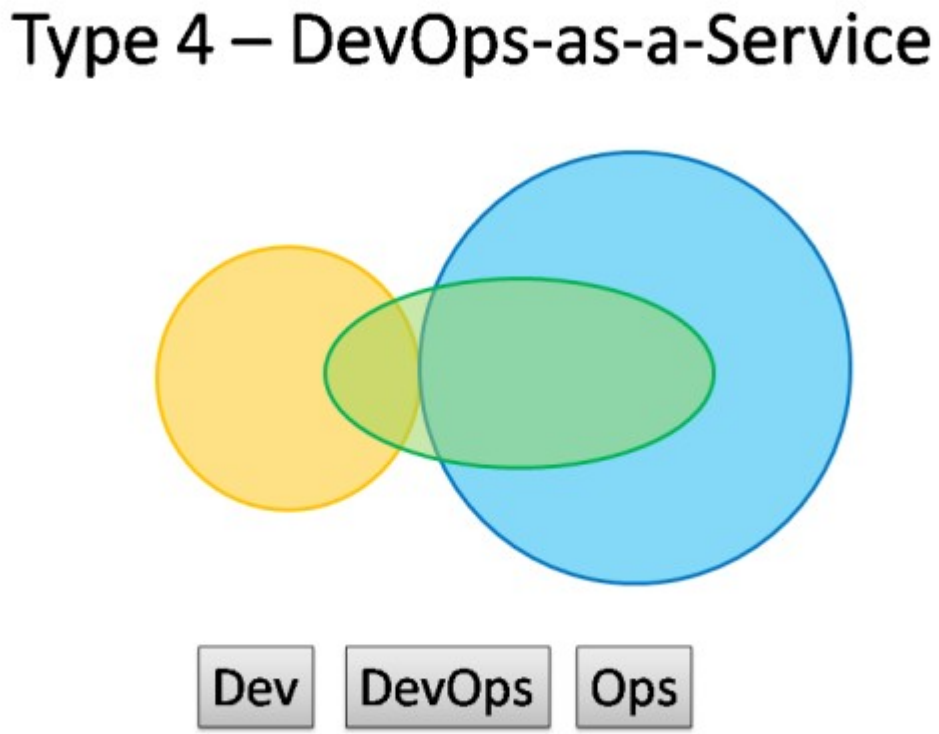
DevOps Patterns

Type 3 – Infrastructure-as-a-Service



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

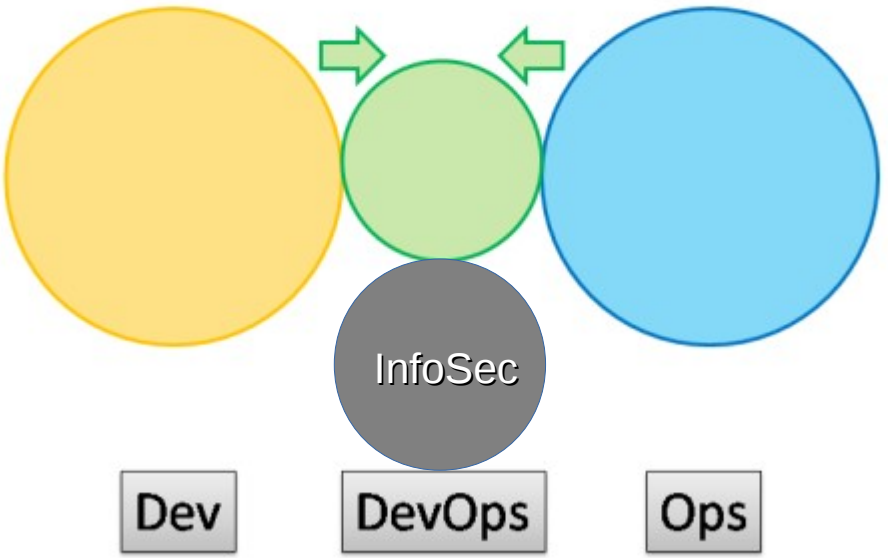
DevOps Patterns



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

DevOps Patterns

Type 5 – Temporary DevOps Team



http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

Deciding about InfoSec strategy w/devops remember:

→ security ninjas (just like admins) are expensive and rare

→ virtual teams might cut this problem

→ wandering experts

Deciding about InfoSec strategy w/devops remember:

→ security ninjas (just like admins) are expensive and rare

→ virtual teams might cut this problem

→ wandering experts

Deciding about InfoSec strategy w/devops remember:

→ security ninjas (just like admins) are expensive and rare

→ virtual teams might cut this problem

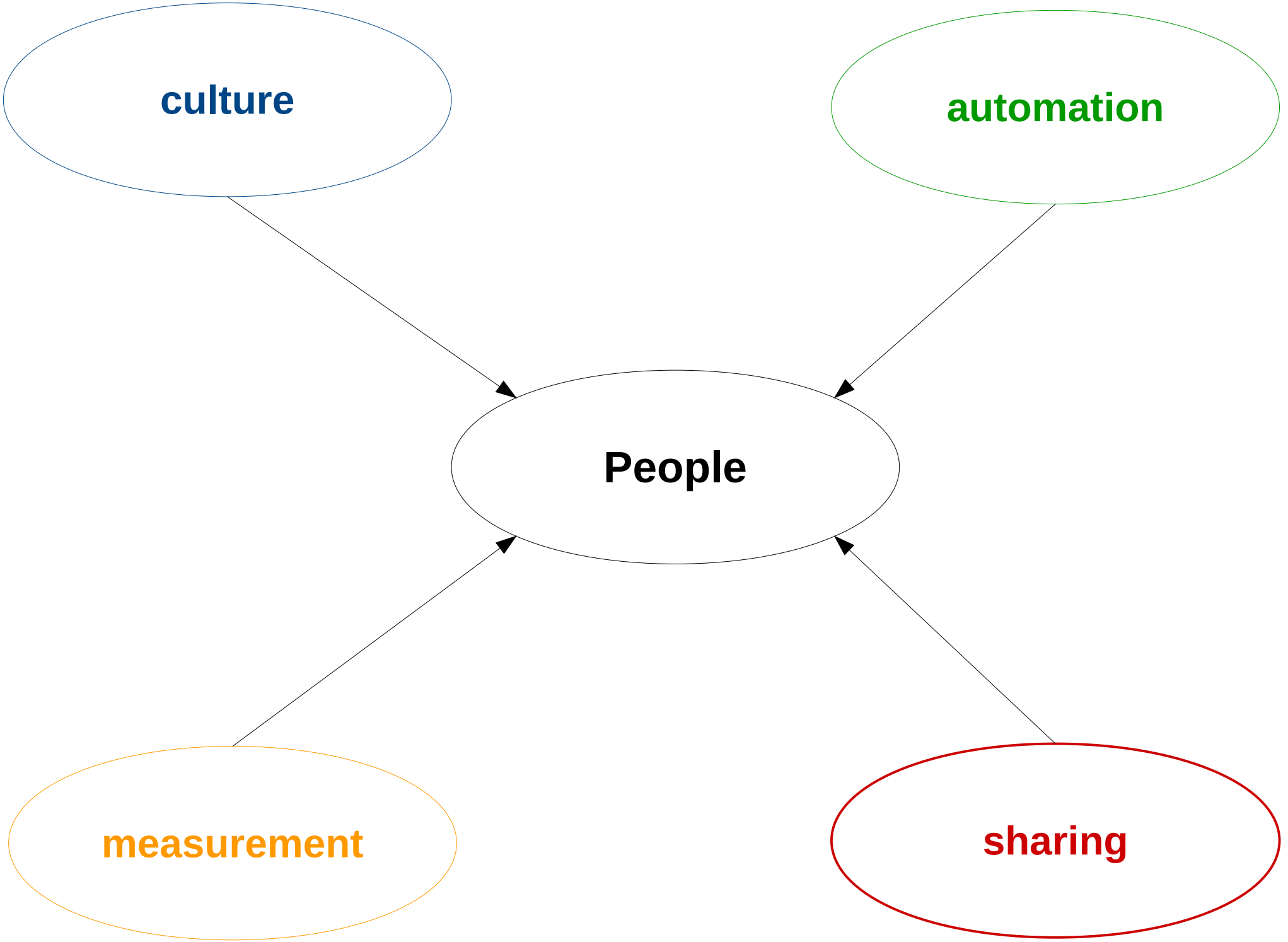
→ wandering experts

DevOPS ?== CAMS

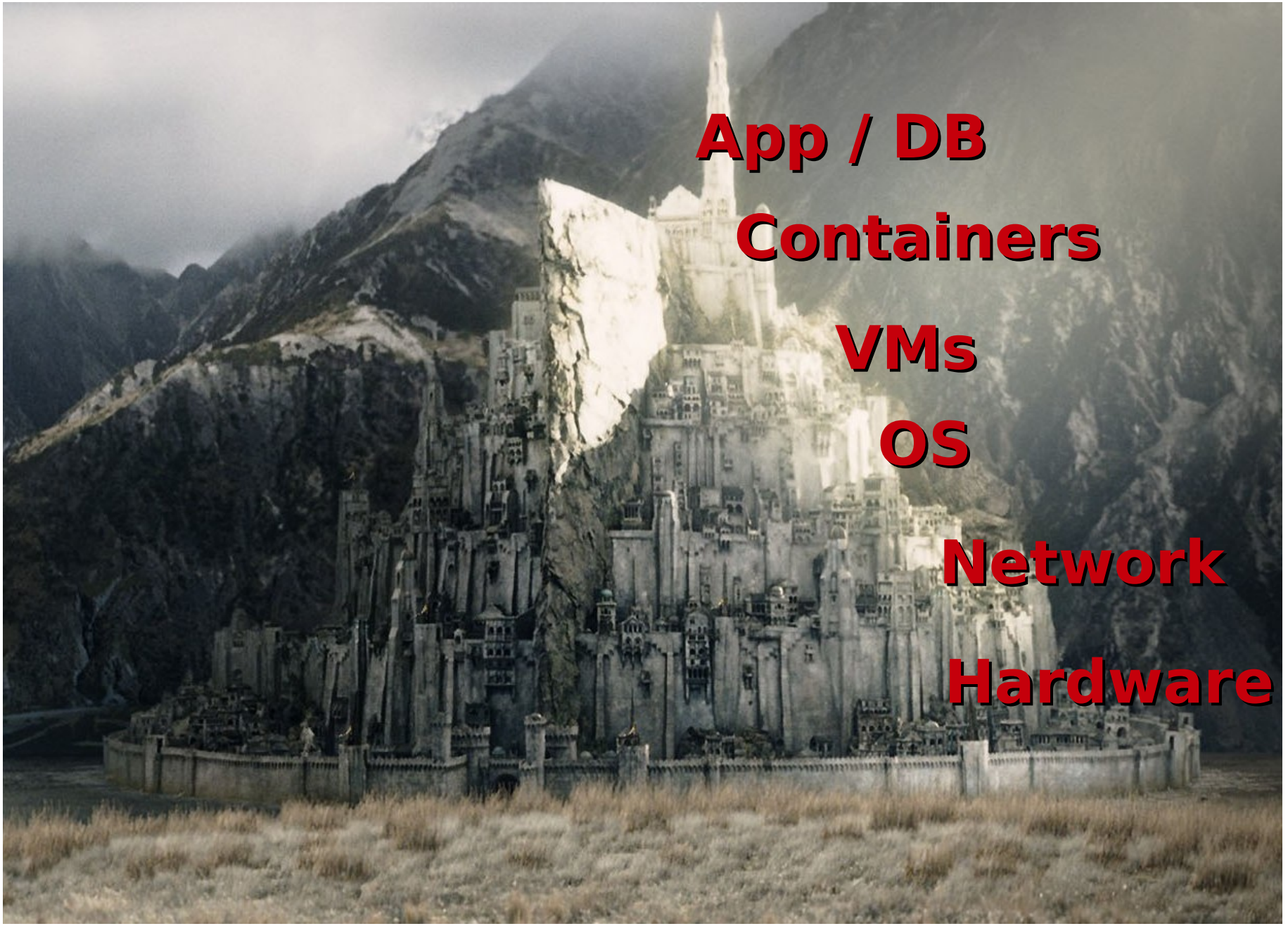
(culture, automation, measurement, sharing)

DevOPS !== CAMS

DevOPS === people!


culture
automation
People
measurement
sharing

General security rule in IT: security is based on layers



General security rule in IT: security is based on layers

C for Culture

A for Automation

M for Monitoring

S for Sharing

→ focus on delivery

→ close collaboration

→ lightweight environment and components

→ lightweight processes

→ focus on delivery

→ close collaboration

→ lightweight environment and components

→ lightweight processes

→ focus on delivery

→ close collaboration

→ lightweight environment and components

→ lightweight processes

→ focus on delivery

→ close collaboration

→ lightweight environment and components

→ lightweight processes

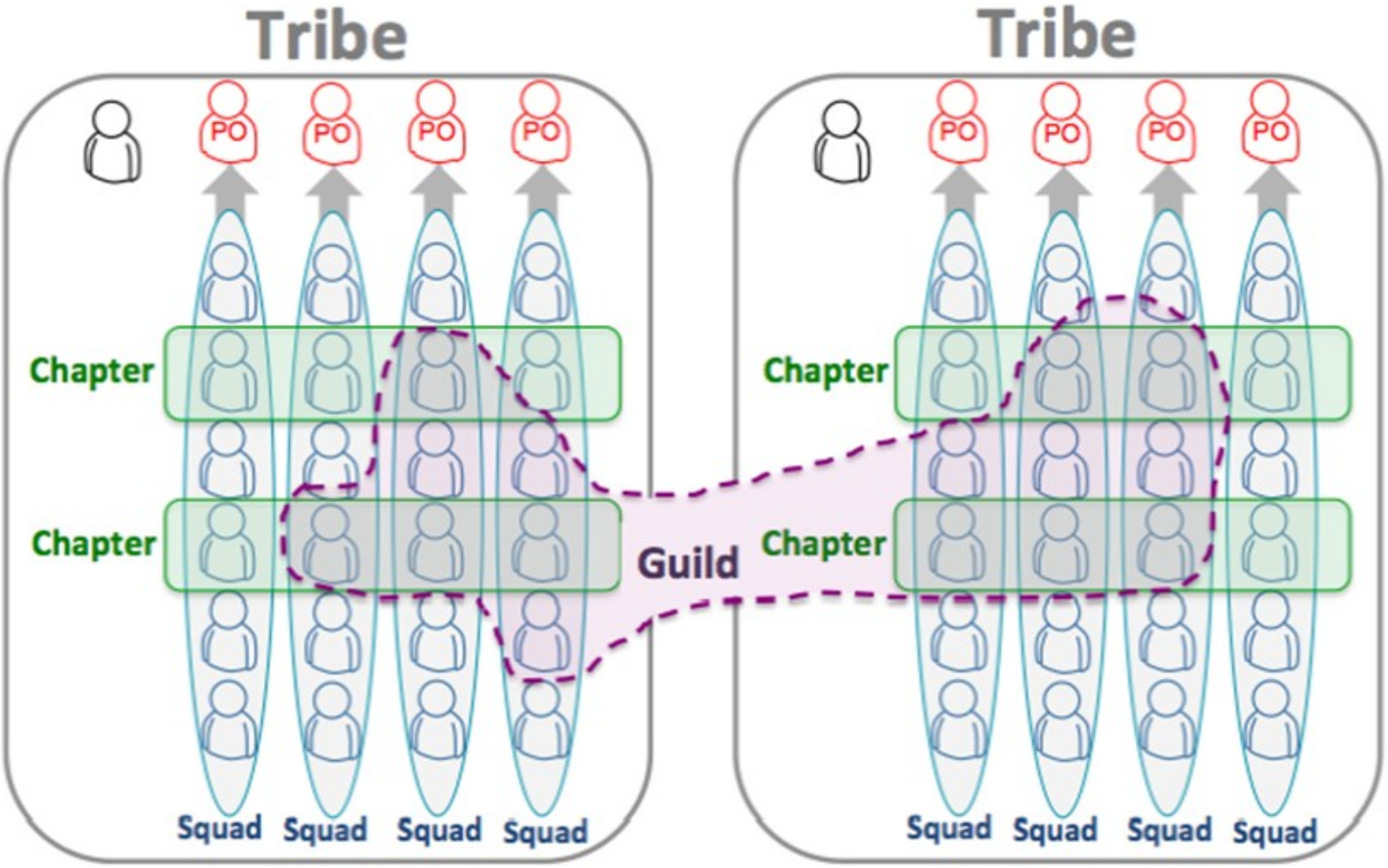
cultural change

modification of a society through innovation, invention, discovery, or contact with other societies

Scaling Agile @ Spotify

with Tribes, Squads, Chapters & Guilds

Henrik Kniberg & Anders Ivarsson
Oct 2012



C for Culture

A for Automation

M for Monitoring

S for Sharing

→ repeatable tasks leads to automation

→ repeatable tasks leads to automation

→ automation leads to consistency

→ repeatable tasks leads to automation

→ automation leads to consistency

→ consistency reduces errors

→ repeatable tasks leads to automation

→ automation leads to consistency

→ consistency reduces errors

→ reducing errors leads to stable environment

→ repeatable tasks leads to automation

→ automation leads to consistency

→ consistency reduces errors

→ reducing errors leads to stable environment

→ stable environment leads to less unplanned work

→ repeatable tasks leads to automation

→ automation leads to consistency

→ consistency reduces errors

→ reducing errors leads to stable environment

→ stable environment leads to less unplanned work

→ less unplanned work leads to focus on delivery

- → flat learning curve
- → doesn't required additional resources
- → fit for maintenance jobs / procedures
- → great for any containers as non-daemon
- → might be easily adopted as universal language
- → ansible-galaxy

```
/
    inventory
       srv_group1
       srv_group2
    group_vars/
       srv_group1
       srv_group2
    host_vars/
       server1
       server2
    roles/
        webserver/
        monitoring/
        app1/
        app2/
        security/
            portscan/
    master.yml
```

```
/
    inventory
      srv_group1
      srv_group2
    group_vars/
      srv_group1
      srv_group2
    host_vars/
      server1
      server2
    roles/
       webserver/
       monitoring/
       app1/
       app2/
       security/
            portscan/
    master.yml
```

```
/
    inventory
      srv_group1
      srv_group2
    group_vars/
      srv_group1
      srv_group2
    host_vars/
      server1
      server2
    roles/
       webserver/
       monitoring/
       app1/
       app2/
       security/
            portscan/
    master.yml
```

```
/
    inventory
      srv_group1
      srv_group2
    group_vars/
      srv_group1
      srv_group2
    host_vars/
      server1
      server2
    roles/
       webserver/
       monitoring/
       app1/
       app2/
       security/
            portscan/
    master.yml
```

```
/
    inventory
      srv_group1
      srv_group2
    group_vars/
      srv_group1
      srv_group2
    host_vars/
      server1
      server2
    roles/
        webserver/
        monitoring/
        app1/
        app2/
        security/
            portscan/
    master.yml
```

```
/
    inventory
      srv_group1
      srv_group2
    group_vars/
      srv_group1
      srv_group2
    host_vars/
      server1
      server2
    roles/
       webserver/
       monitoring/
       app1/
       app2/
       security/
           portscan/
    master.yml
```

```
/
    inventory
      srv_group1
      srv_group2
    group_vars/
      srv_group1
      srv_group2
    host_vars/
      server1
      server2
    roles/
       webserver/
       monitoring/
       app1/
       app2/
       security/
           portscan/
    master.yml
```

```
ansible-playbook master.yml  \
    –tags app2,portscan
```

```
- name: run portscan
  shell: /usr/bin/nmap -sS -p- > wide_scan_results

# vars in e.g. group_vars
ports:
      tcp:
            - 80
            - 443

--exclude-ports="{{ ports.tcp | join(",") }}"

async, pool, fire & forget

- name: Parse results
  shell: python parse.py {{ ports.tcp }}
  register: parse_results

- name: Notify
- shell: echo "{{ parse_results.stdout }}" | mail -s "results" a@b.com
- when: '"error_placeholder' in parse_results.stdout"
```

```
- name: run portscan
  shell: /usr/bin/nmap -sS -p- > wide_scan_results

# vars in e.g. group_vars
ports:
      tcp:
            - 80
            - 443

--exclude-ports="{{ ports.tcp | join(",") }}"

async, pool, fire & forget

- name: Parse results
  shell: python parse.py {{ ports.tcp }}
  register: parse_results

- name: Notify
- shell: echo "{{ parse_results.stdout }}" | mail -s "results" a@b.com
- when: '"error_placeholder' in parse_results.stdout"
```

```
- name: run portscan
  shell: /usr/bin/nmap -sS -p- > wide_scan_results

# vars in e.g. group_vars
ports:
      tcp:
            - 80
            - 443

--exclude-ports="{{ ports.tcp | join(",") }}"
```

async, pool, fire & forget

```
- name: Parse results
  shell: python parse.py {{ ports.tcp }}
  register: parse_results

- name: Notify
- shell: echo "{{ parse_results.stdout }}" | mail -s "results" a@b.com
- when: '"error_placeholder' in parse_results.stdout"
```

```
- name: run portscan
  shell: /usr/bin/nmap -sS -p- > wide_scan_results

# vars in e.g. group_vars
ports:
      tcp:
            - 80
            - 443

--exclude-ports="{{ ports.tcp | join(",") }}"
```

async, pool, fire & forget

```
- name: Parse results
  shell: python parse.py {{ ports.tcp }}
   register: parse_results

- name: Notify
- shell: echo "{{ parse_results.stdout }}" | mail -s "results" a@b.com
- when: "'error_placeholder' in parse_results.stdout"
```

```
- name: run portscan
  shell: /usr/bin/nmap -sS -p- > wide_scan_results

# vars in e.g. group_vars
ports:
      tcp:
            - 80
            - 443

--exclude-ports="{{ ports.tcp | join(",") }}"

async, pool, fire & forget

- name: Parse results
  shell: python parse.py {{ ports.tcp }}
   register: parse_results

 - name: Notify
 - shell: echo "{{ parse_results.stdout }}" | mail -s "results" a@b.com
 - when: "'error_placeholder' in parse_results.stdout"
```

C for Culture

A for Automation

M for Monitoring

S for Sharing

→ Visualization – graph everything (or make it possible)

→ Same monitoring interfaces for all

→ Logfiles lines number (e.g. audit.log) as a metric

→ False negs / pos number as a metric

C for Culture

A for Automation

M for Monitoring

S for Sharing

It's simple as: stop hiding security incidents reports in the

locked drawer

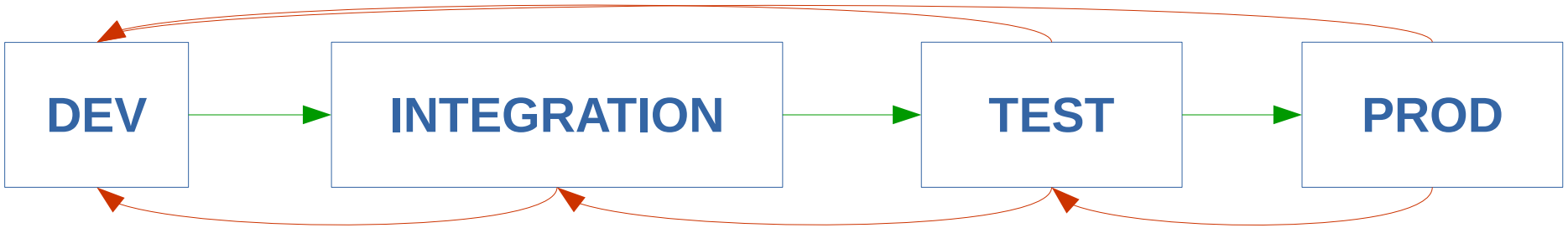
Let other learn: think continuous improvement!

Share the knowledge about mistakes


DEV
INTEGRATION
TEST
PROD

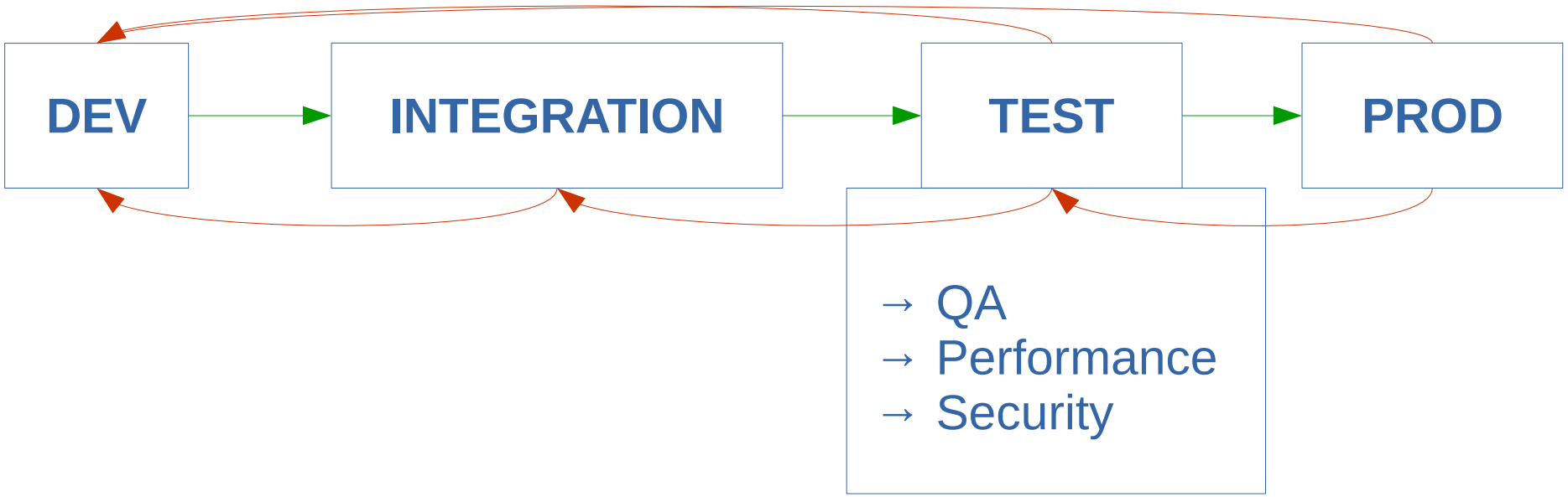
Delivery Pipeline!

Delivery Pipeline!



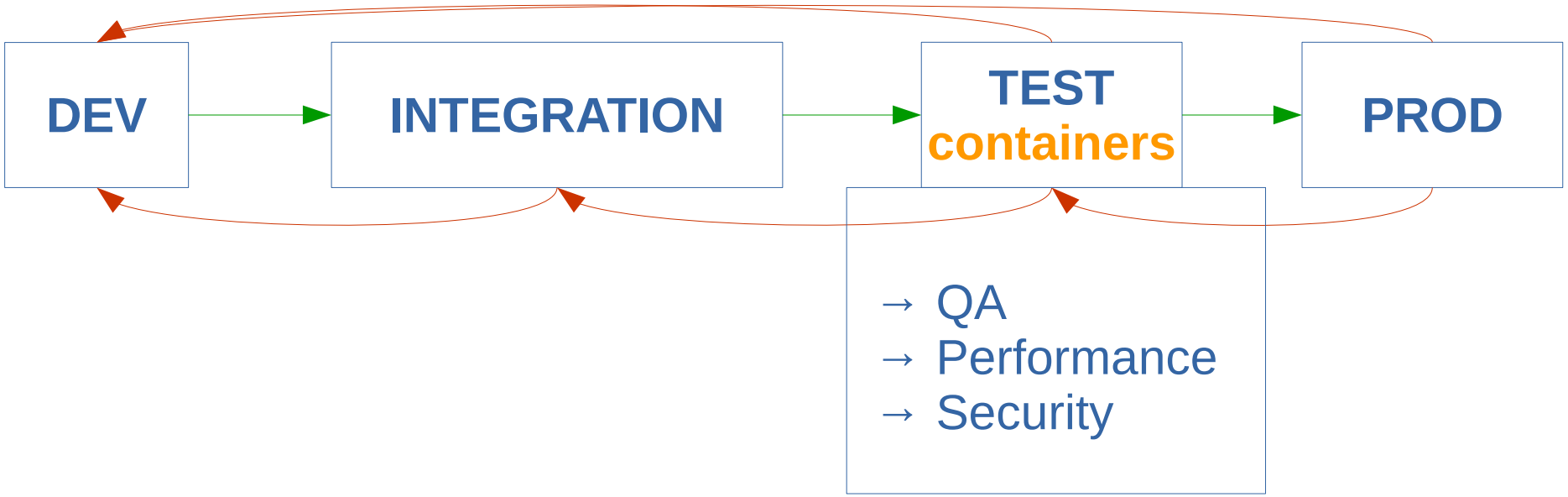
Feedback loop!

Delivery Pipeline!



Feedback loop!

Delivery Pipeline!



Feedback loop!

Experimentation gives you improvements!

Continuous security scanning

Let's wrap this up

→ security is about providing quality – it must be the part of delivery

→ including security in CD is a business decission; involve business in devops!

→ security doesn't have to slow the CD pipeline

Let's wrap this up

→ security is about providing quality – it must be the part of delivery

→ including security in CD is a business decision; involve business in devops!

→ security doesn't have to slow the CD pipeline

Let's wrap this up

→ security is about providing quality – it must be the part of delivery

→ including security in CD is a business decission; involve business in devops!

→ security doesn't have to slow the CD pipeline

Deep dive into technical infra

(briefly, more in my arch presentation today)

Linux Containers

why InfoSec should bother about infra?

→ because infra is a code

→ because infra might be a tool

control groups (cgroups)

- grouping processes
- allocating resources to particular groups
  - memory
  - network
  - CPU
  - storage bandwidth (I/O throttling)
  - device whitelisting

Kernel Namespaces

Providing a unique views of the system for processes.

→ PID – PIDs isolation

→ NET – network isolation (via virt-ifaces; demo @arch)

→ IPC – won't use this

→ MNT – chroot like; deals w/mountpoints

→ UTS – deals w/hostname

Layered filesystems

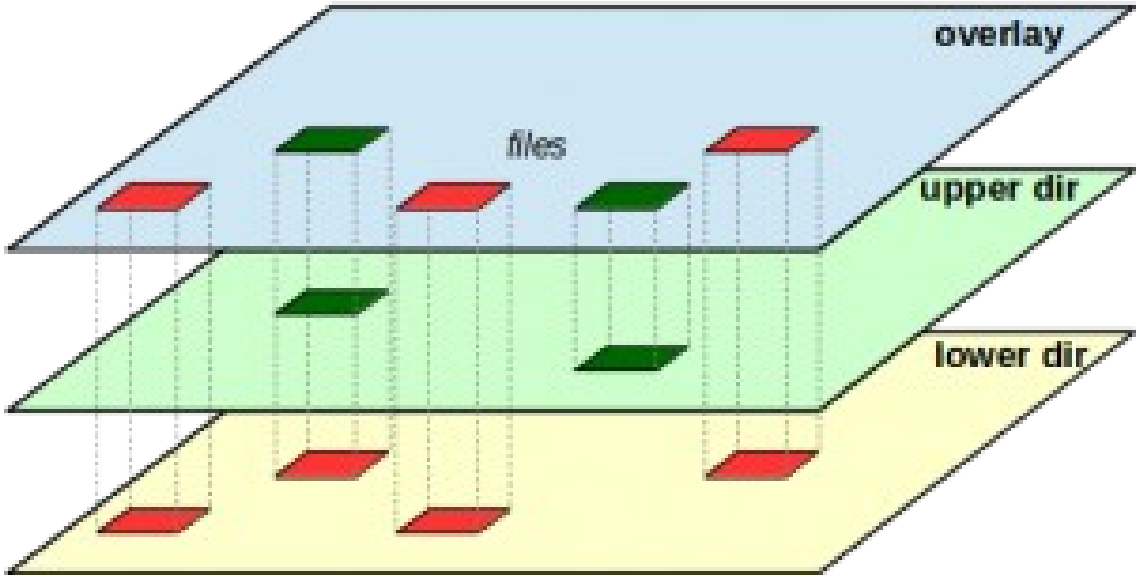
→ OS installation

→ libraries

→ application

→ apps updates

We ship this as one package – container
It has to be lightweight!



http://www.blaess.fr/christophe/2014/12/14/le-systeme-overlayfs-de-linux-3-18/

Docker in a nutshell – installing WP in seconds demo

Docker in a nutshell – installing WP in seconds demo

remember #DockerKrk & infosec & devops meetups

http://www.meetup.com/Docker-Krakow-Poland/

http://www.meetup.com/Krakow-DevOps/

http://www.meetup.com/Infosec-Krakow/

It doesn't have to be docker

LXC, LXD, systemd-nspawn etc

Just make sure it does its job

Summing this up – learn how to use containers

so you can focus on InfoSec work not on infrastructure mojo

You'll see how this repays :)

Tools overview

GAUNTLT - http://gauntlt.org/

→ Hooks for sectools (nmap, sslyze, sqlmap)

→ Output formatting (json and others)

→ see yourself (demo)

nikto - https://www.cirt.net/Nikto2

→ webapp sec scanner

→ customizable reports (templates)

→ logging to metasploit

→ save full requests for positive tests

→ ...

→ see yourself (demo)

nikto - https://www.cirt.net/Nikto2

And docker maybe? (demo)

https://registry.hub.docker.com/u/activeshadow/nikto/dockerfile/

Remember to verify those images..

nikto - https://www.cirt.net/Nikto2

```
FROM debian:jessie

RUN apt-get update && apt-get install -y libtimedate-perl libnet-ssleay-perl \
	&& rm -rf /var/lib/apt/lists/*

ADD https://cirt.net/nikto/nikto-2.1.5.tar.gz /root/
WORKDIR /opt

RUN tar xzf /root/nikto-2.1.5.tar.gz && rm /root/nikto-2.1.5.tar.gz \
	&& echo "EXECDIR=/opt/nikto-2.1.5" >> nikto-2.1.5/nikto.conf \
	&& ln -s /opt/nikto-2.1.5/nikto.conf /etc/nikto.conf \
	&& chmod +x nikto-2.1.5/nikto.pl && ln -s /opt/nikto-2.1.5/nikto.pl /usr/local/bin/nikto \
	&& nikto -update

WORKDIR /root
CMD ["nikto"]
```

wapiti - http://wapiti.sourceforge.net/

→ webapp sec scanner

→ rich vulns detection (see docs)

→ JSON reports (and some other formats)

→ suspend / resume attack

→ modular

→ ...

→ see yourself (demo)

skipfish - https://code.google.com/p/skipfish

→ webapp sec scanner

→ high performance

→ easy to use

→ rich vulns detection (see docs)

→ ...

→ see yourself (demo)

mittn - https://github.com/F-Secure/mittn

- → high level testing suite
- → alternative for Gauntlt
- → no required low-level knowledge about tools
- → Python / Behave (BDD)
- → automated web scanning w/Burp (BSPAS)
- → tls w/sslyze
- → HTTP api fuzzing w/Radamsa

OWASP + DevOps (by Mateusz Olejarka)

https://www.owasp.org/images/d/df/Owasp_plus_devops.pptx

→ OWASP dependency check

→ OWASP dependency track

→ OWASP ESAPI

→ OWASP AppSensor

→ OWASP Zed Attack Proxy

→ O-Saft

How to deal with false negs / pos?

→ actually human analysis is always required

→ before “feedback loop” check yourself if it's red

→ mark, hide, automate

Demo

→ install docker

→ install jenkins

→ install owasp-zap container

→ install wordpress container

→ configure scan job

→ run it

→ try w/docker inside docker:

http://www.jayway.com/2015/03/14/docker-in-docker-with-jenkins-and-supervisord/

Looking for a job?

Information Security Manager

Catch me: maciek@lasyk.info

Continuous Security in DevOps

Maciej Lasyk

@docent-net

http://maciej.lasyk.info

4developers – Warsaw

2015-04-20