Dev Sec Ops
more of the same (*)

(*) back to the roots

@patrickdebois

my own view
devops mantras
devops buzzword bingo

The initial conflict

DEV **OPS**

Optimise the **whole**
systems thinking

What is *your* biggest
bottleneck ?

FRONTEND OPS
NETWORK OPS
WIN OPS
BIZ DEVSEC OPS
NO OPS
AI OPS
CHAT OPS
GIT OPS

HR Marketing Finance Sales

BUSINESS

DEV OPS

Dev Sec Ops

yet another bottleneck

dev(.*)ops is a **label**
to get the stories out

Agile manifesto
ITIL books
Devops chaos

Culture (Lean) Automation
Measurement Sharing

@botchagalupe

CULTURE
(the beliefs)

We can't be an expert in everything
- we **NEED** to collaborate
security is feeling the pressure

Full stack *developer*

Developer First

+

T-shaped people

empathy

#hugops

You are (co) - responsible
shared responsibility

We can only be responsible
for **our own *promises***
(promise theory)

Mark Burgess

Assume/Embrace failure 💥

and learn from it

engineering mindset 

Bring the pain **forward**
(seek feedback)

You build it, you run it
... you secure it

Tools (alone) don't matter

Toolishness

Not about the technology
it's about the culture (*)

(*) actually ... it's about the business !

LEAN
(business)

Not different from any other change process

start small / celebrate success

management buy-in

There is no such thing as a dev(sec)ops TEAM (*)

(*) should be a transitional phase

Excuse:
This will never work here
(read: we are not willing to change ... yet)

Goal Alignment
individual > team > organisation
> business > industry

Backlog alignment

business value
vs operational cost
vs insurance

Do the right thing
vs the thing right

As a strategic weapon

Turn Offense into Defense

Iterative *development*

Iterative *deploy*

Iterative *defence*

Understand the system
value chain
vs kill chain

Less costly to fix early
in the chain

Shift Right
&& Shift Left

What if your bottleneck moves **outside your control?**

become friends with your suppliers
(cloud, OSS , ...)

AUTOMATION
(delivery)

If it's hard

do it more often

continuous

integration

deployment

delivery

verification

everything

repeatable **b u i l d s**

version control +
artifacts ~ inventory
(discovery phase)

Supply chain - Provenance
code + signing
+ environment
+ actions

You need tests
You need brakes
You need controls
to go faster

Agile ~ TDD

Devops ~ Config Mgmt

DevSecops ~ Threat Modeling

Technology as a conversation starter

it is not about testing
or performance or security
but about architecture
this creates resilience and agility

Repeatable process

Process Standardisation

(Exceptional)🦄
Change advisory board

Turn frequent exceptions
in standard changes

automated resolutions
for known states

treat servers as cattle not pets

treat policies on roles and not people

(make it scale)

it will not take your job away
but allow you to spend more time
on more important tasks

MEASUREMENT
(feedback)

Monitoring ~~sucks~~
Monitoring ❤️

Inventory
Prioritise
Implement
Validate

stories per day
of deploys per day
vulnerability fixes per day
+ $ per day

Story Points
vs
Error Budget
vs
Vulnerability Budget

Event Cognitive Overload

The Dunbar # for metrics

Signal noise ratio

take all issues seriously

triaging story, bug , cve first

Compliance / auditing
spreadsheets vs automated
fictional vs realistic

Learn from mistakes

blameless post mortems

(no single causality)

escape rates 🔑
gotta catch em all

Protect yourself from
biases/unknowns

Observability 👀

Resilience *engineering* 

Vulnerability bounty programs
~ Chaos 🙈

game days ~
capture the 🇹🇩

Feedback channel
for the business to **learn** from

SHARING
(openness)

Shortage of Talent🤘
vs Lack of Resources

1:10:100

Competence level up
builds **trust**

Self Organizing team
don't come over night 🐢

Educate practices
dojos - sharing

Mob programming

 remove barriers

Our work is to protect others from hurting themselves 🧨

vs..

ALL changes
MUST be peer-reviewed (*)

(*) probably not if you trust others

What the **others needs** to do
vs what can we do ourselves

Think service not server

serverless vs servicefull

DEVOPS (managing servers)
is DEAD - Simon Wardley

Self Servicing

Bring information to
where it's **needed**
(self feedback)

Incentivise people
to do the right thing
(paved road)

Product mindset

platform team

Are you building
confidence vs trust?
the system takes over

devsecops
as a devops *evolution*

devops as a
pre-requisite to implement
succesful devsecops

BEWARE:
Your bottleneck moves

🏁 Are we there yet? (*)

(*) NO - but let's keep sharing our stories

features , containers & vulnerabilities

come and go 

but developers , security and operations are forever☀️

THANK YOU!

@patrickdebois