Planning For A Responsive AppSec Program

@DarrenPMeyer

Problems

- Difficult to anticipate what will come next
  - Waterfall, Agile, DevOps… what next?
- Adaptation to new methodologies is slow in AppSec compared Development
  - Your developers are using DevOps *now*; DevSecOps is still building
- Development trends toward speed of delivery
  - Speed is the enemy of safety

Root causes

There are a lot, and they vary by org, but commonly include:

- Lack of Agility
  - Security orgs generally do not pursue Agility as a goal

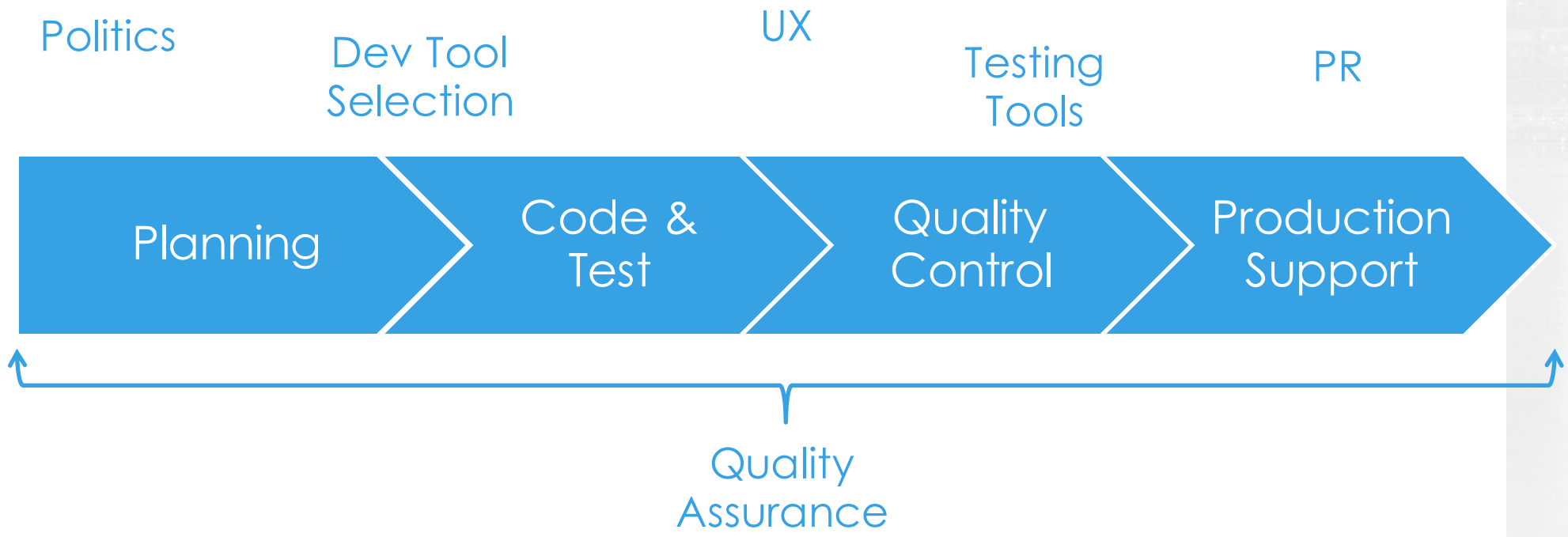
- "We are special" syndrome
  - Belief that security is too different from other Quality activities to borrow from their toolkits

Actions

- Get security out of its isolated box
  - "Move left" mantra in DevSecOps is a good start!
- Move away from a pure-defense model of security
  - Security is *Quality*
  - Security is *Safety* – or better yet, *Resiliency*
- Consciously design for greater agility

Get out of the Security box

- "Move left", yes. Also move *out*.



Security is *Quality*

- Security is a Quality Assurance activity. Stop treating it like audit and Quality Control

- Quality Control is a *failsafe* – Quality Assurance is constant
  - Which sounds more like what security should be?

Security is *Resiliency*

- Safety models work really well for security

- The resiliency safety model as applied to security:
  - Treat security as a socio-technical system problem
  - Operation is Normal, Abnormal, or Emergency
  - Goals are:
    - Spend as much time as possible in Normal
    - Limit the damage caused by Abnormal and Emergency situations

Resiliency: Security is Socio-Technical

- Remember "People, Process, Technology"?
  - "People" doesn't just mean hiring meat to fill seats
  - People interact with process and technology, and you *must* plan for how they do so and what risks that poses

- Are your developers, QA folks, etc. *part of* your security process?
  - Do they know? Do they *believe you*?
  - Do they have a stake in how it happens?

Resiliency: Operation Modes

- Normal
  - Things are working as you expect. All to plan.
- Abnormal
  - Things aren't going to plan, but in well-understood ways.
  - You have a plan for what to do and how to recover
- Emergency
  - We don't have a (specific) plan for this!!

Resiliency: Goals

- Spend as much time as possible in Normal
  - This is where we defend: try to stop bad things from happening
    - But the cup is already broken
  - Also make plans to get *back to* normal as quickly as possible

- Limit the damage caused by Abnormal and Emergency situations
  - In public health and safety circles, this is "Harm Reduction"
  - "Bad things will happen, lets make them less bad"

Consciously design for Agility

- Objectives > Controls
  - Controls are only evaluated on "do they meet objectives"
  - Controls are in a constant state of change, Objectives less so

- Perfect is the enemy of good
  - Accepting risk – with eyes open – is OK
  - Chip away at risk, optimize for "what can we do *now*"