ThoughtWorks®

SEGREGATION OF DUTIES AND CONTINUOUS DELIVERY

How to enable Continuous Delivery while continuing to protect the business and customers.

Sriram "Ram" Narayanan

www.sriramnarayanan.com

@sriramNRN

A friendly implementation of
Segregation of Duties
enables Continuous Delivery,
Security and Compliance to co-exist

What we'll cover today

- About Continuous Delivery
- The need for Segregation of Duties
- How typical enforcement of Segregation of Duties is a blocker to CD
- How to improve SoD enforcement and accelerate CD

Important Points

- People behave as they are measured (e.g. KPIs)
- Most issues are 10% technical and 90% cultural/behavioral
- CD-Friendly SoD and true Continuous Delivery are more process and people problems, and very less tool problems.
- You should move toward automation-friendly tools, though.

ThoughtWorks®

ABOUT CONTINUOUS DELIVERY

It's beyond Continuous Integration, and beyond "CI/CD"

What Continuous Delivery is NOT:

Topic	**Clarification**
“CI/CD”	You need more than just a “daemonic CI” and a “pipeline plugin”
Continuous Deployment	Deployment using Tools
Blanket permission to	Environment owners need to review, approve and trigger deployments at their convenience.
Permission to push “Containers” to Prod	What goes in those containers needs to be validated!

Continuous Delivery

Keep software in a reliable and deployable state so that you can deploy on demand.

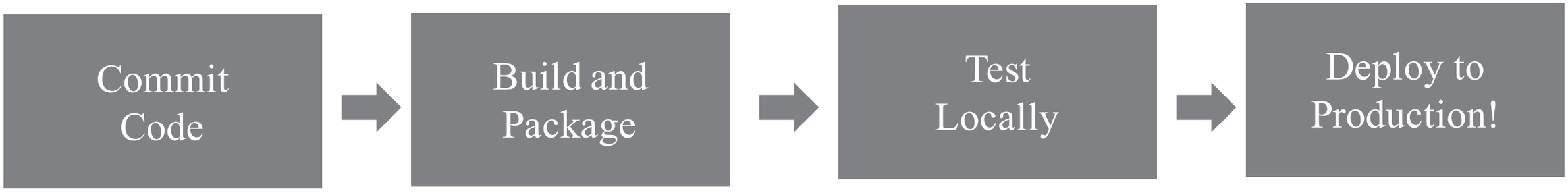
Continuous delivery is a software engineering approach in which teams produce software in short cycles, ensuring that the software can be reliably released at any time. It aims at building, testing, and releasing software faster and more frequently.

- Wikipedia

With fast I.T. turn-around times, business can:

- Stay competitive
- Respond to change faster
- Fix defects earlier
- Try new ideas boldly and revert confidently.

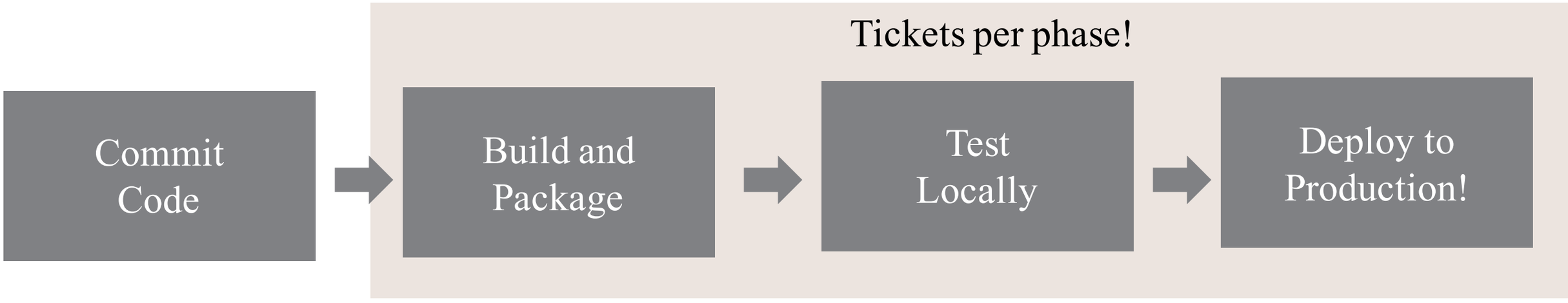
What we'd love to have!



Production Support

- Deploy when ever we want
- Debug processes on Production servers
- Query Production Databases
- Inspect traffic, review log files
- Apply hot fixes within minutes

Reality Check !



Production Support

- ~~Deploy when ever we want~~ - “Raise a ticket to deploy”
- ~~Debug processes on Production servers~~ - “No way !!”
- ~~Query Production Databases~~ - A ticket for individual query results
- ~~Inspect traffic, review log files~~ - A ticket for log extracts
- ~~Apply hot fixes within minutes~~ - Ticket please!

What puzzles (frustrates!) Dev Teams and Business

- Why are Ops, Audit and Security Teams throwing roadblocks at us?
- Are they raising roadblocks just to assert their importance?
- Why are Ops given access that they cannot make use of to solve issues?
- Why do we have such ridiculous policies!?
- Why does everyone make us raise so many tickets?
- Why are we trusted to write the software but not to troubleshoot it!!!??
- Are Ops, Security and Compliance on our side, or our competitors side?

What Ops, Security, Compliance have to say:

"We are merely following industry norms to protect business and customers. We are not the enemy! Please don't blame us for doing our job!!"

So, who is right?

Development teams – who develop software that meets business goals?

Or

Ops and Security – who ensure uptimes and protect customers?

ThoughtWorks®

SEGREGATION OF DUTIES

Why Ops, Security and Compliance do what they do

Expectations from an organization

- Make money (if a business)
- Conform to the laws (e.g. those that protect the customers' interests)
- Run in a stable manner

How orgs are managed - GRC

Topic	Explanation
Governance	The executives are responsible for the org's operations
Risk Management	Identify, analyze and respond to risks
Compliance	Conform to stated requirements (Regulations, Org policies, Business guarantees to customers)
Applicable to	IT, Finance, Legal

Source: Wikipedia

Some examples of fraud and error

- Untimely and/or non-uniform deployment
- Deploying with the wrong permissions
- Handling production environments with zero exposure and skills
- Accessing confidential data in violation of privacy policies
- Changing production configurations ad-hoc with poor review, and poor documentation of changes
- Bypassing domain logic and enforcement in the application, and changing production data directly
- Logging confidential data and accessing these via logs

Separation of duties *(SoD) (also known as "Segregation of duties") is the concept of having more than one person required to complete a task. … an internal control intended to prevent fraud and error*

- Wikipedia

Segregation of Duties

- A well-understood concept in Finance, Law, Governance, Military, etc.
- No single person should have end to end access to complete an entire workflow
- At least one other person should be able to
  - Regulate the activity, if need be.
  - Review the activity

Segregation of Duties in IT

No single person or team should have end to end access from code to production

Typical SoD procedures for Deployments

Intent	**Action**	**Typical Implementation**	**Impact**
Devs should not author and deploy code	Deployment by Ops	Dependent upon Ops availability	Business cannot deploy on-demand
Demonstrate deployment in an auditable manner	Deployment using Tools	Special tools, typically not available in Dev	Dev and Prod deployments are different
Control over when prod is changed	Deployment at specific times	Strict calendar schedules	Cannot deploy frequently. Exceptions can be expensive.

Typical SoD procedures for Troubleshooting

Intent	**Action**	**Typical Implementation**	**Impact**
Devs should not access confidential data in logs	Regulate access to log systems	Access to prod logs governed by SLAs. Extracts only.	Lack of direct access to logs prevents fast troubleshooting
Prevent adhoc/harmful changes, and data sniffing	Regulate access to prod servers	Special tools, typically not available in Dev	Dev and Prod deployments are different

Typical SoD procedures for Databases

Intent	**Action**	**Typical Implementation**	**Impact**
Ensure database schema and data integrity by skilled DBAs	Regulate changes to databases	Changes reviewed and denied before prod deployment. Documentation.	Waste of precious time. Wasteful documentation.
Prevent adhoc/harmful changes, and data sniffing	Regulate access to prod data	A query per ticket, reviewed, approved, applied	Waste of precious time. Penalties for delays

Typical SoD procedures for Configuration

Intent	**Action**	**Typical Implementation**	**Impact**
Ensure that all (app,OS) changes to prod are valid and documented	Regulate changes to production	Changes reviewed and denied before prod deployment. Documentation	Waste of precious time. Wasteful documentation.
Prevent attacks based on known weaknesses	Apply patches regularly at scheduled intervals	Configuration (settings, patches) not shared with devs	Software not tested with Prod configuration

Defensive SoD and insecurely architected software can prevent Continuous Delivery

ThoughtWorks®

CD-FRIENDLY SOD

Ensure Segregation of Duties while also enabling fast response times

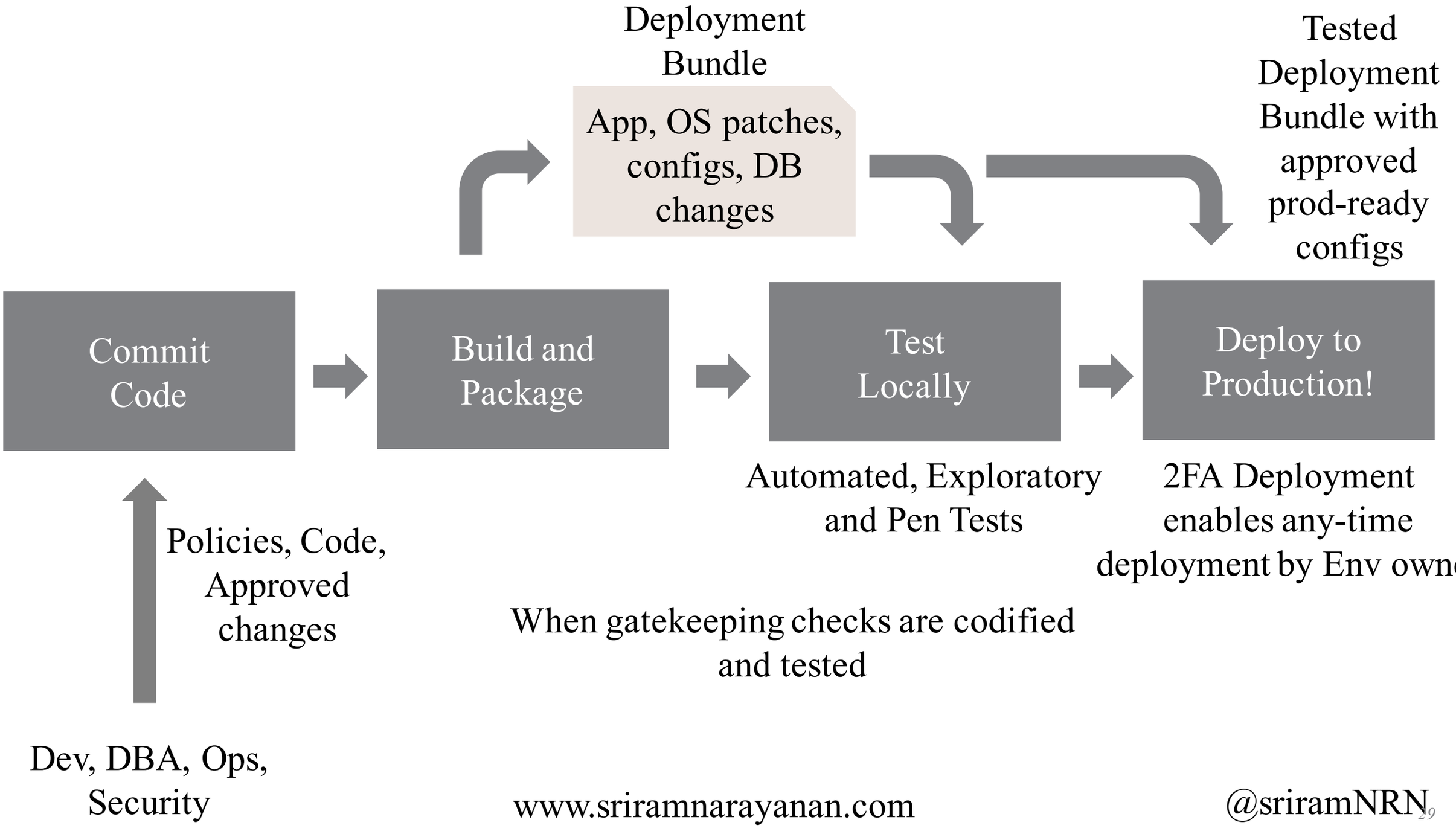
www.sriramnarayanan.com

@sriramNRN

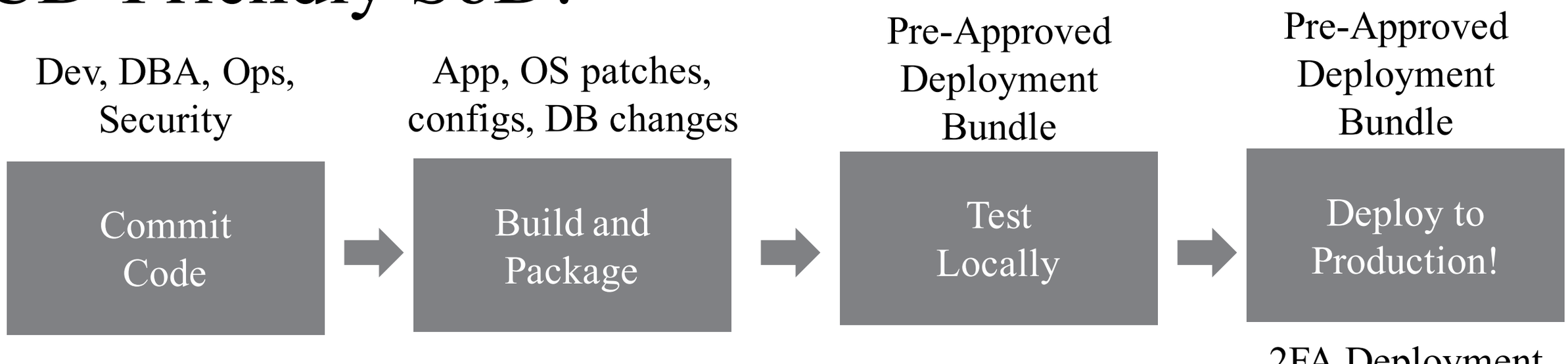
CD-friendly SoD – General principles

- Involve Ops and Security right from Design phase
- Policies in executable form via CD-Friendly config mgmt tools.
- Separate confidential data and logs from regular data and logs
- Single Deployment bundle – app, config, policy, DB schema.
- Bundle Once, Deploy anywhere
- Restrict access to confidential data/logs, permit easy access to regular data/logs.
- Enforce via config than via tickets (e.g. resource throttling vs tickets).
- Use multi-factor (vs tickets) where possible to regulate actions.

CD-Friendly deployment and configuration



CD-Friendly SoD!



Production Support

- Deploy when ever we want – Environment owners decide, use 2FA
- Debug processes on Production servers – Yes, configs elsewhere.
- Query Production Databases – Easier access to regular data.
- Inspect traffic, review log files – Easier access to regular data.
- Apply hot fixes within minutes – Test in 1-click dev envs first

CD-Friendly SoD procedures for Deployments

Intent	**Action**	**Recommended Implementation**	**Impact**
Devs should not author and deploy code	Deployment by Environment Owners	Review and deploy changes in small batches	Small batches makes changes easier to review.
Demonstrate deployment in an auditable manner	Configuration management tools	Build once, deploy anywhere	Dev-Prod are the auditably the same
Control over when prod is changed	Deployment by Environment Owners	Frequent Deploys in small batches. Multi-factor controls	Deploy only when the Env owner wants to.

CD-Friendly SoD procedures for Troubleshooting

Intent	**Action**	**Recommended Implementation**	**Impact**
Devs should not access confidential data in logs and config files	Separate confidential and regular logs. Externalised configuration	Log UUIDs. Prod Support teams access regular logs, and can SSH to prod.	Confidential data remains restricted. Prod support is fast.
Prevent adhoc/harmful changes, and data sniffing	Standard environments.	1-click environment creation and 1-click deployment	Prod errors can be caught earlier in Dev. Reduces prod errors.

CD-Friendly SoD procedures for Databases

Intent	**Action**	**Recommended Implementation**	**Impact**
Ensure database schema and data integrity by skilled DBAs	Regulate changes to databases using CD-friendly DB config tools	DBAs review and recommend changes at Dev using CD-friendly tools.	Identical schema from Dev through prod as approved by the DBA.
Prevent adhoc/harmful changes, and data sniffing	Delink confidential and regular data.	Restrict access to confidential data. Provide access to regular data.	Most troubleshooting needs just regular data, and is fast.

CD-Friendly SoD procedures for Configuration

Intent	Action	Recommended Implementation	Impact
Ensure that all (app,OS) changes to prod are valid and documented	Ops and Security config settings in CD-Friendly config management tool	Test pre-approved configs from dev through prod.	Pre-approved and tested configs enable frequent deploys.
Prevent attacks based on known weaknesses	Test OS patches in Dev	Apply and test OS patches via automation in 1-click dev env.	Rapidly test OS patches and Software in non-Prod first.

THANK YOU

Sriram "Ram" Narayanan

@sriramNRN

ram@thoughtworks.com

www.sriramnarayanan.com