Architecture, Deployment and Operations

A baseline application development and deployment model based on lessons learned from the last two years of building, extending and deploying apps in CI/CD in “enterprise” GIS.

Background

Everywhere I go, I see a recurring pattern:

- Myriad approaches to solving the problems of:
  - Provisioning local dev environments
  - Management (or absence of) test/stage/prod infrastructure
  - Effectively synchronizing configurations between all of the environments
- Some approaches seem to work but most aren't scalable
- Customers start demanding cargo-cult activities to "fix" problems introduced by ill-fitting solutions
- True automated continuous deployment becomes difficult or impossible as the band-aid solutions pile up

What’s the scope of the problem?

Some of the stickier issues this model aims to solve:

- Difficulty standing up a local dev environment
- Inconsistency between build agents and runtime environments cause builds to pass but deployments to fail
- One-size-fits-all build infrastructure leading to snowflake system configs
- Installing “sciency” dependencies can be hard to get right and harder to repeat

Proposed Model

Prerequisites

- Containers to bridge the gap between developer laptops, Continuous Integration (CI) servers and production servers
- At minimum, partial adherence to 12-Factor Application Development Principles
- Ability to create, modify and delete various AWS resources*

Local Development Environment

Developers run Docker* on their local machines

- Applies to:
  - In-house developed software that relies on specific non-default system libraries
  - Backing services (e.g., database, map server, cache, proxy, etc)
- Must be able to start on a "fresh" dev box with **only** git and Docker installed
- `Dockerfile` is a "recipe" you hand to Ops to create backing services if disparate teams

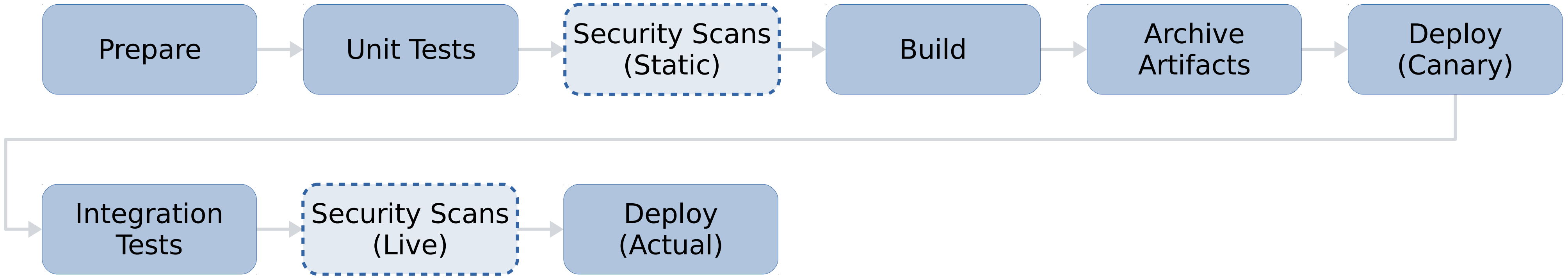
PROPOSED MODEL

CI/CD Pipeline

Parameters

Name	Default	Description
`version`	`HEAD`	Git tag or commit SHA to be built and deployed
`target`	`stage`	Enumeration: `{dev|stage|prod}`
`skip_slow_scans`	`false`	

Stages



PROPOSED MODEL

Deployment Infrastructure

Naming and Tagging is critical

Build Artifacts (S3, Nexus, etc)

```
myproj-build-artifacts/<component>/myproj-<component_name>-<version>-
<commit_sha>.tar.gz
```

Domains/Routing (DNS, nginx, Apache, etc)

Canonical Route (used for Canary testing):

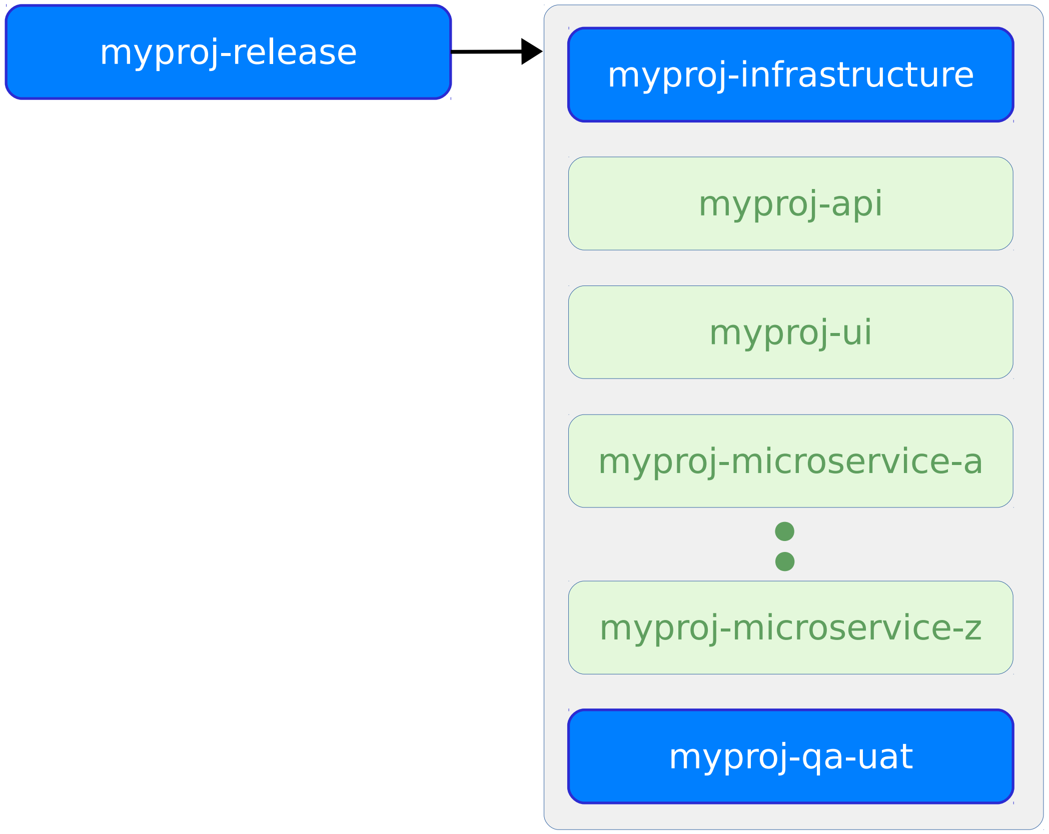
```
<component_name>-<commit_sha>.<target>.myproj.somecorp.com
```

Pointer Route (used by product owner, QA, etc):

```
<component_name>.<target>.myproj.somecorp.com
```

Releases

Releases Orchestration via CI/CD Pipelines



Dev/Stage

`<target>.myproj.somecorp.com`

Production

`myproj.somecorp.com`

Questions?

???