PART 4: APPLICATION RUNTIMES

Infrastructure as Code

Kief Morris @ ThoughtWorks

WORKSHOP CONTENTS

FOUNDATIONS

STACKS

TESTING

RUNTIMES

DESIGN

WORKFLOW

PART 4

APPLICATION RUNTIMES

APPLICATION PACKAGING

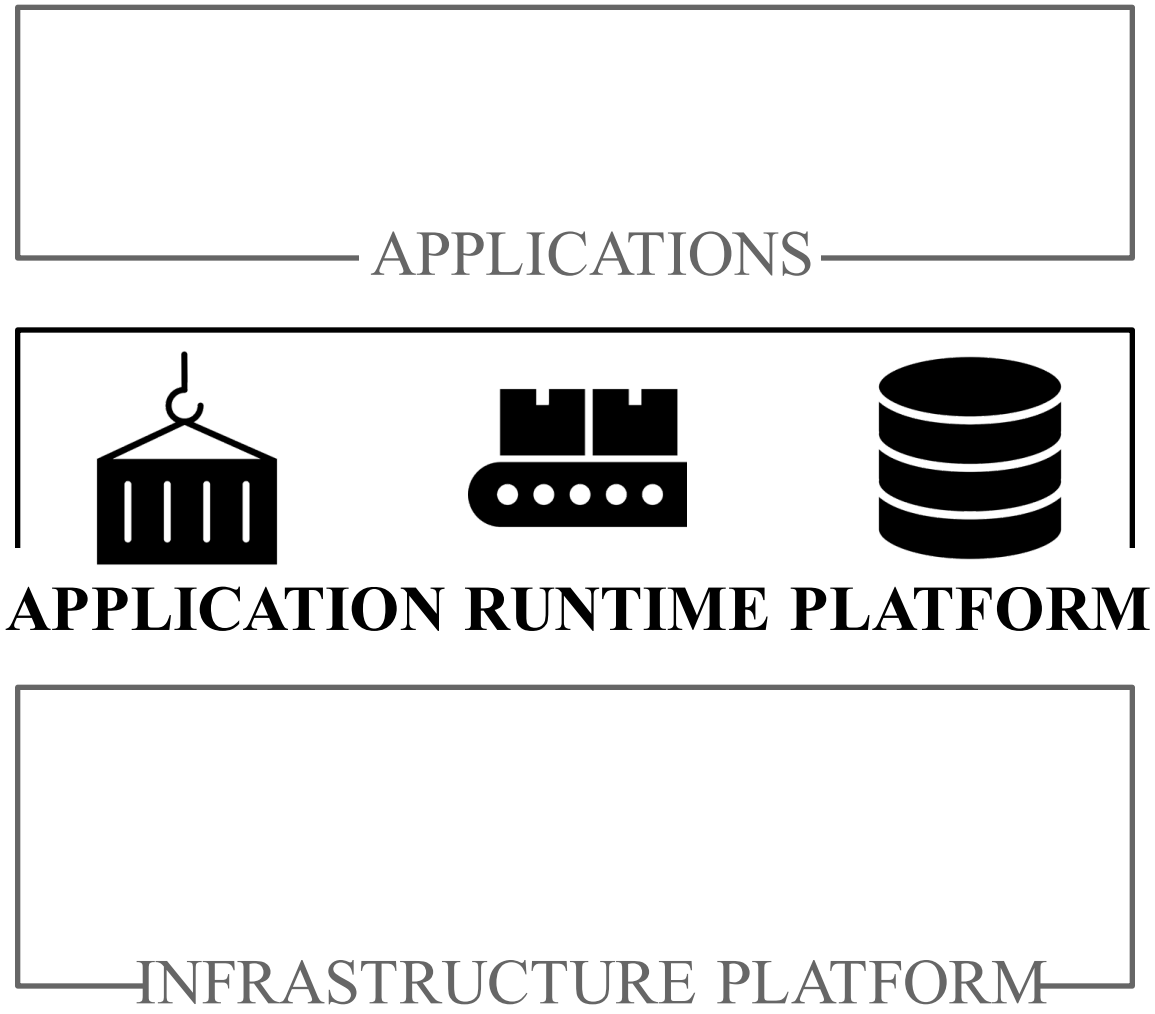
SERVERS AS CODE

SERVER LIFECYCLE

CLUSTERS AS CODE

WHAT:

- Process management
- Application orchestration
- Application-level connectivity
- Database services



EXAMPLES:

- Operating system
- Application server
- Container cluster
- PaaS
- FaaS runtime
- Service mesh
- Database server

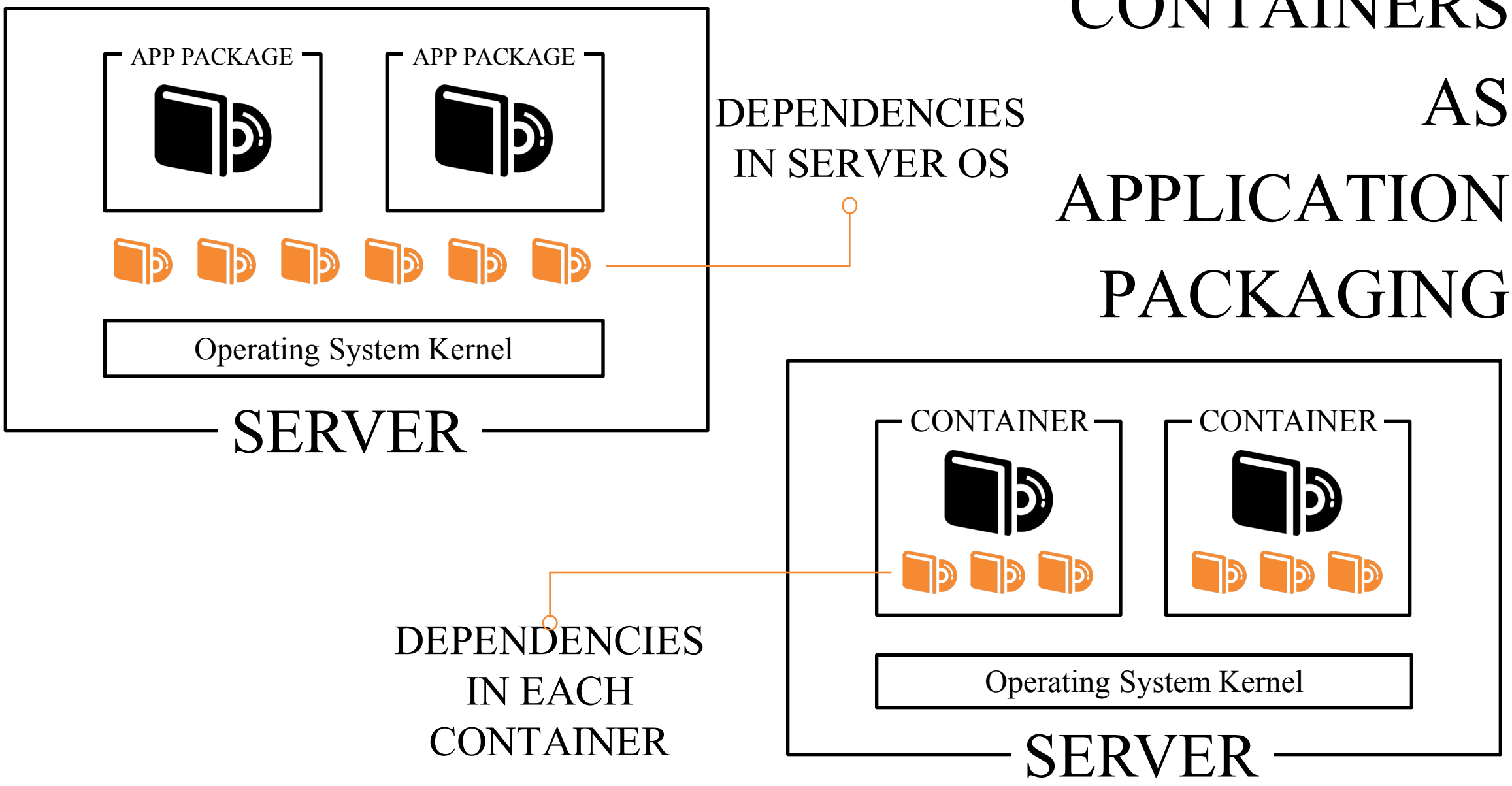
APPLICATION PACKAGING

DEPLOYABLE PARTS

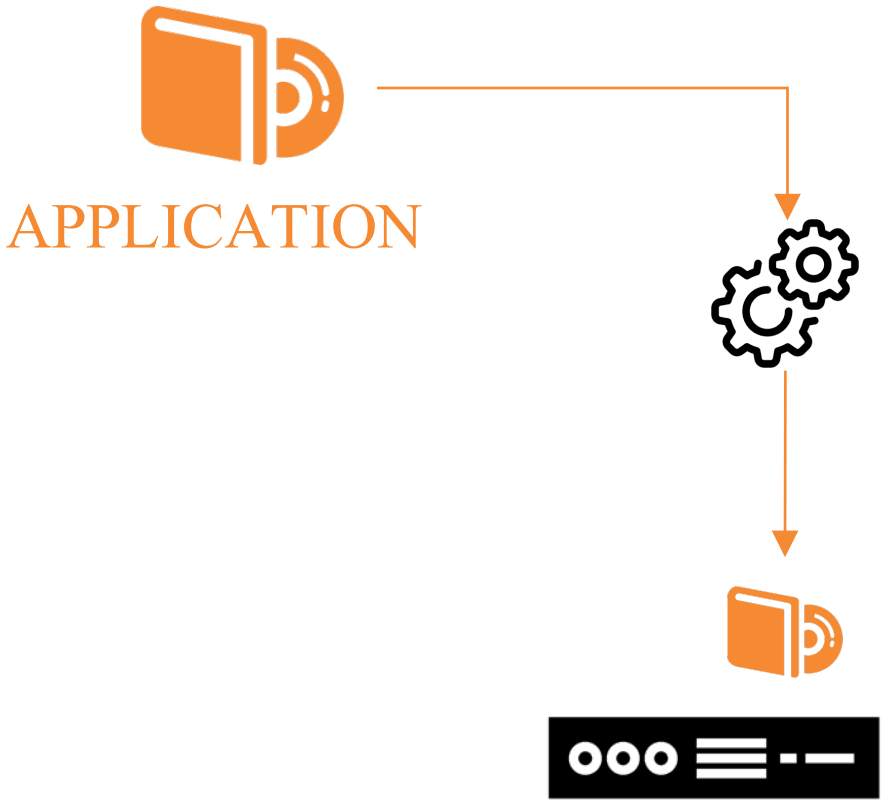
Of an application

- Executables
- Configuration
- Data

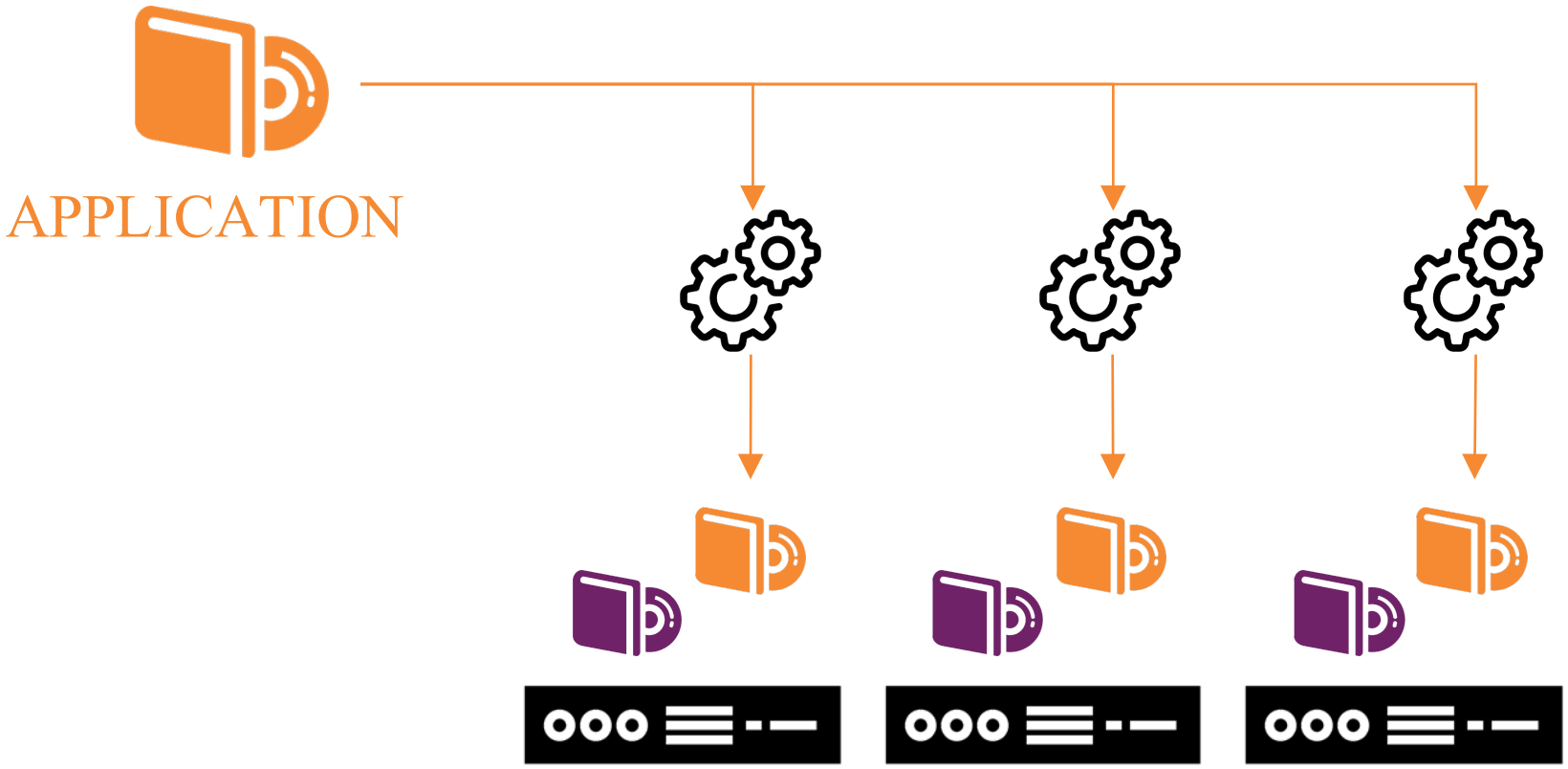
CONTAINERS AS APPLICATION PACKAGING



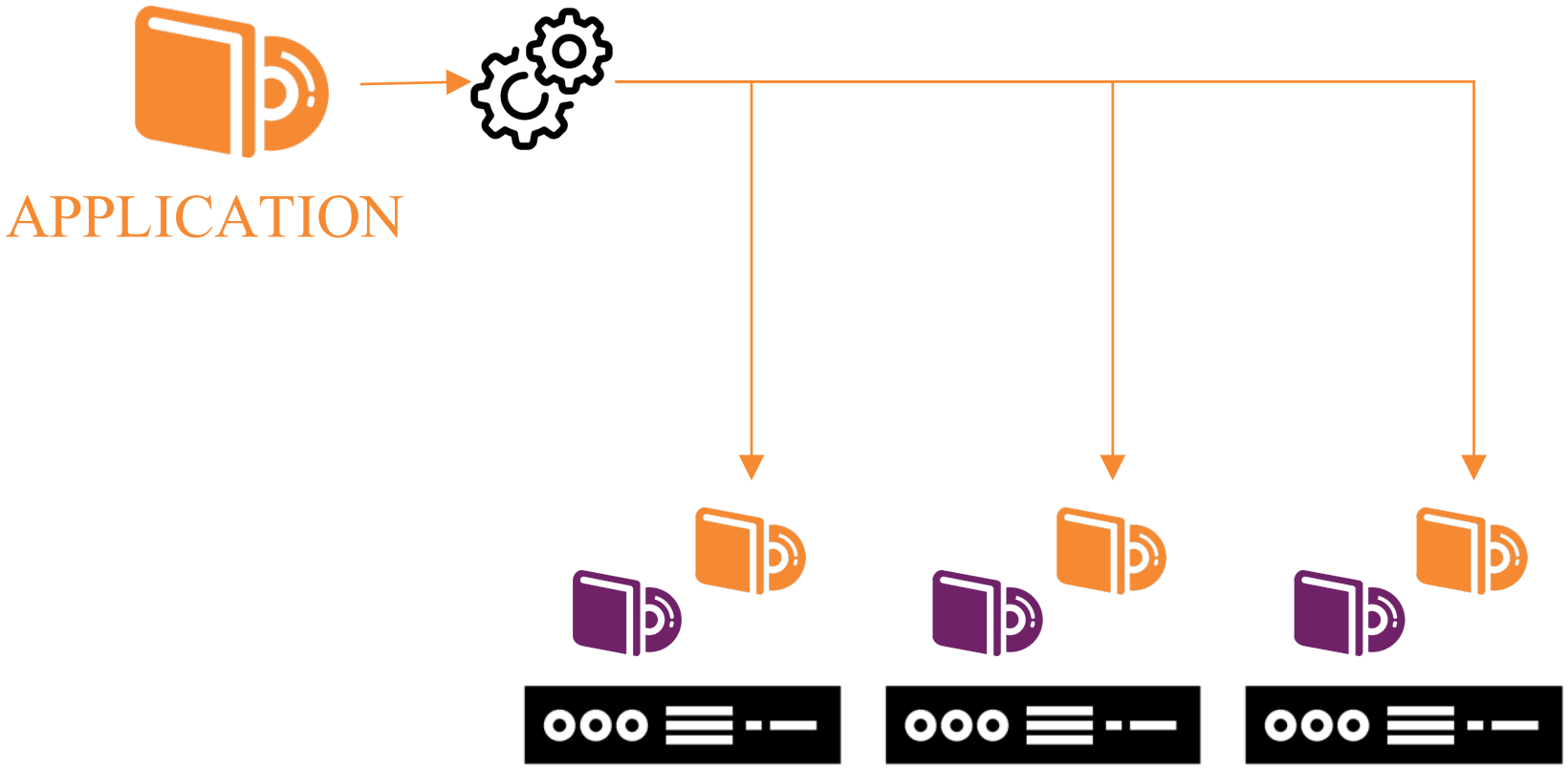
DEPLOYING TO A SERVER



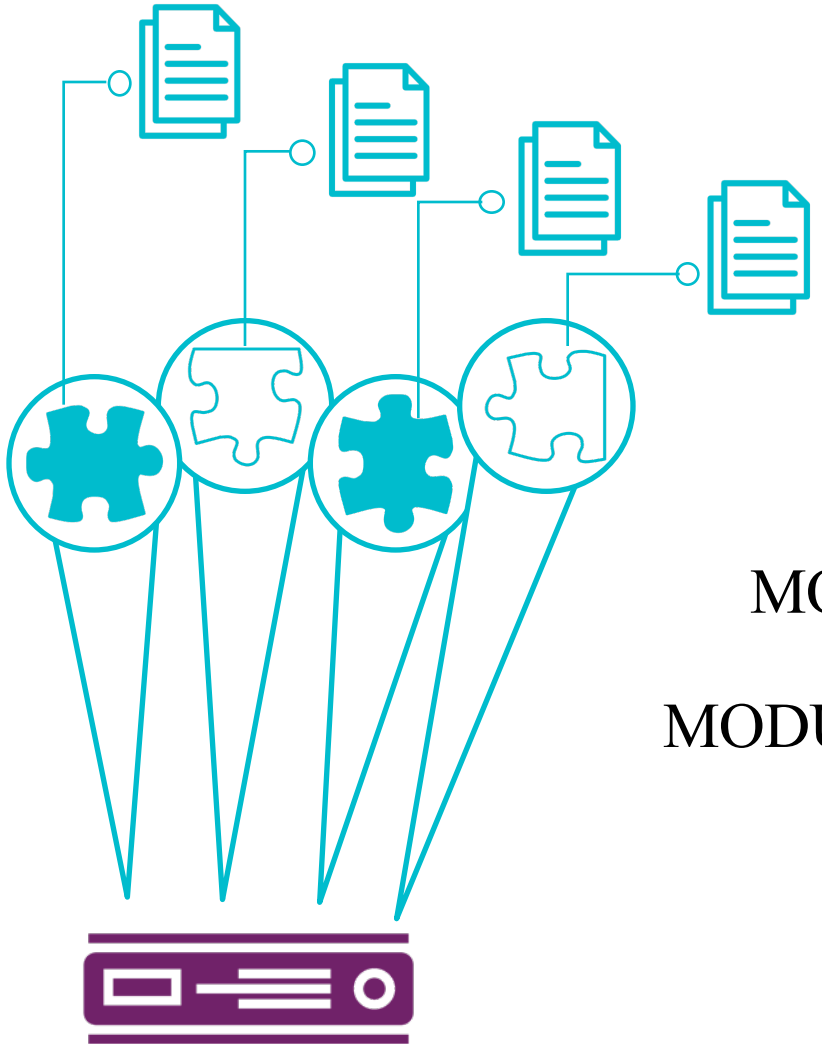
DEPLOYING TO A SERVER CLUSTER



DEPLOYING TO AN APPLICATION CLUSTER




SERVERS AS CODE



SERVER CONFIGURATION MODULES & ROLES

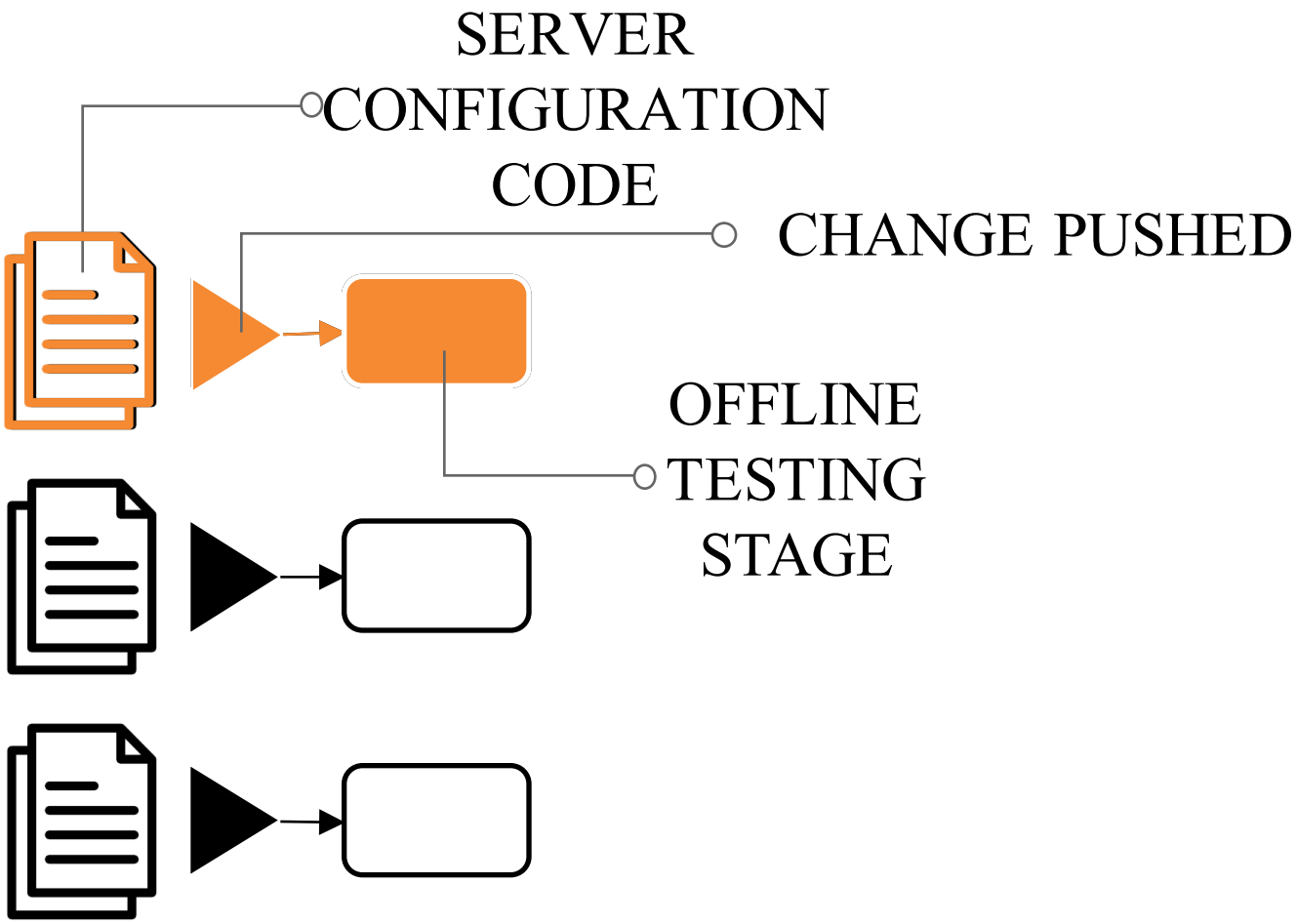
MODULE: JDK

MODULE: TOMCAT

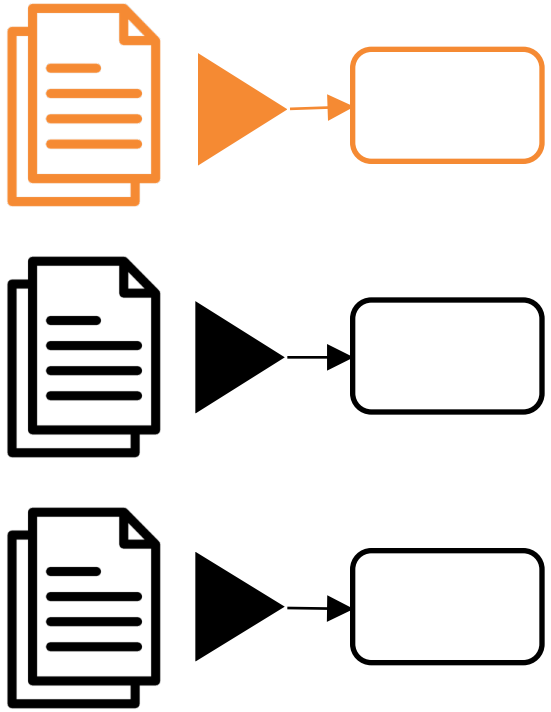
MODULE: MONITOR

MODULE: LOGGING

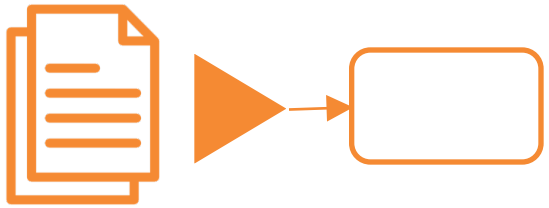
ROLE: APPLICATION SERVER


SERVER
CONFIGURATION
CODE
CHANGE PUSHED
OFFLINE
TESTING
STAGE

OFFLINE SERVER CONFIGURATION TESTING STAGE



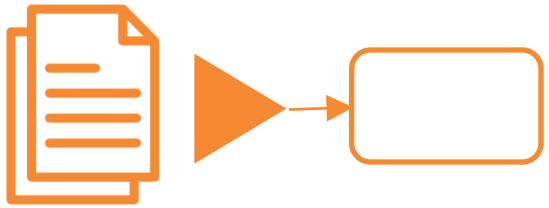
OFFLINE SERVER CONFIGURATION TESTING STAGE



EXAMPLE ENVIRONMENT:

- Basic Linux image in a Docker instance
- Runs on pipeline agent

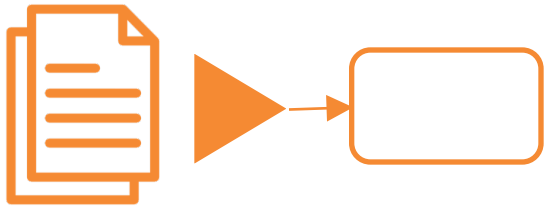
OFFLINE SERVER CONFIGURATION TESTING STAGE



SOME TESTING TOOLS:

- Serverspec
- Inspec

OFFLINE SERVER CONFIGURATION TESTING STAGE



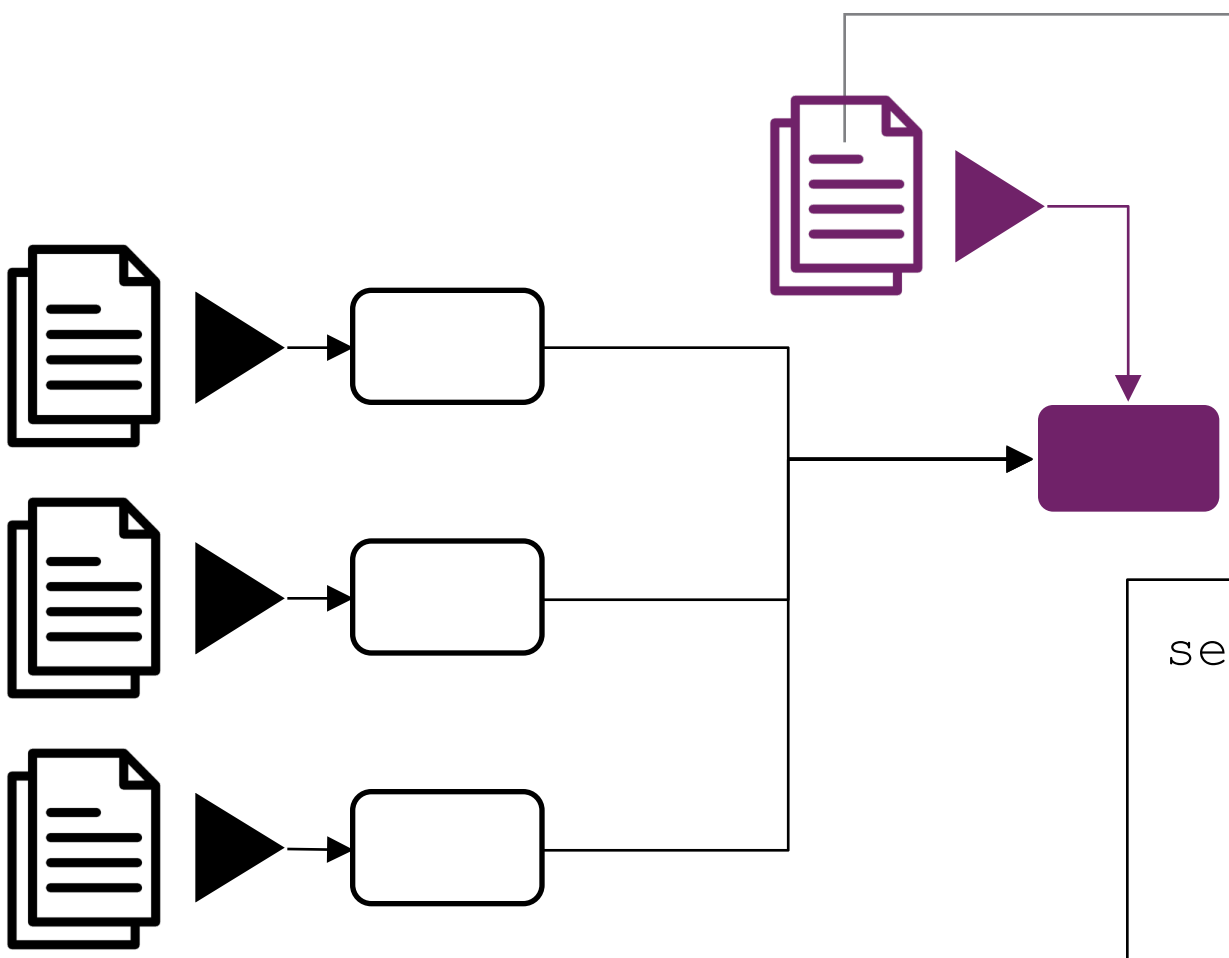
ORCHESTRATION:

- Set up testing environment
- Apply configuration code
- Run tests
- Record results

SOME TOOLS:

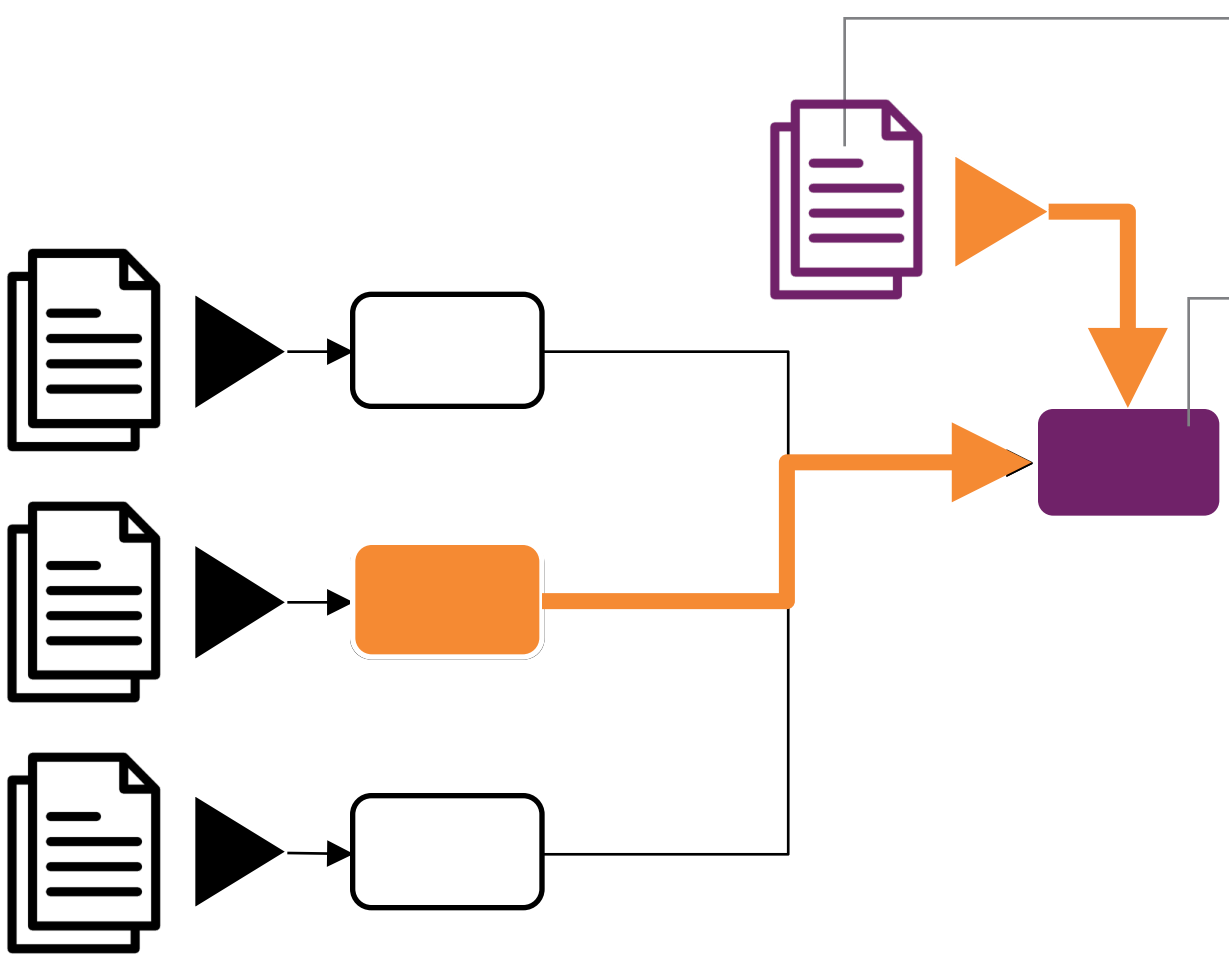
- test-kitchen
- molecule

SERVER
ROLE
CODE



```
server_role: application_server
  elements:
    jdk:
       distro: openjdk
       version: 11.0.5
    tomcat:
       version: 9.0.27
```

SERVER ROLE TESTING



SERVER ROLE CODE

OFFLINE STAGE TO TEST SERVER ROLE

Stage triggers:

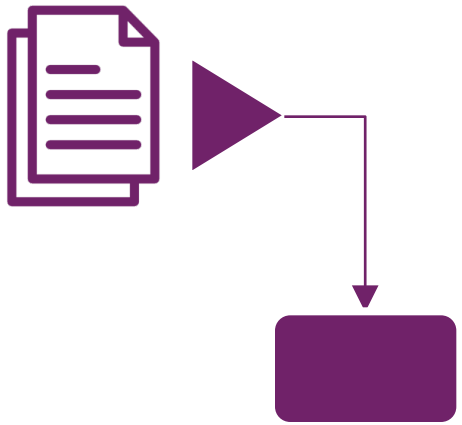
- On change to server role code
- On successful test of a server element's code

SERVER ROLE TESTING

SERVER ROLE TESTING - OFFLINE

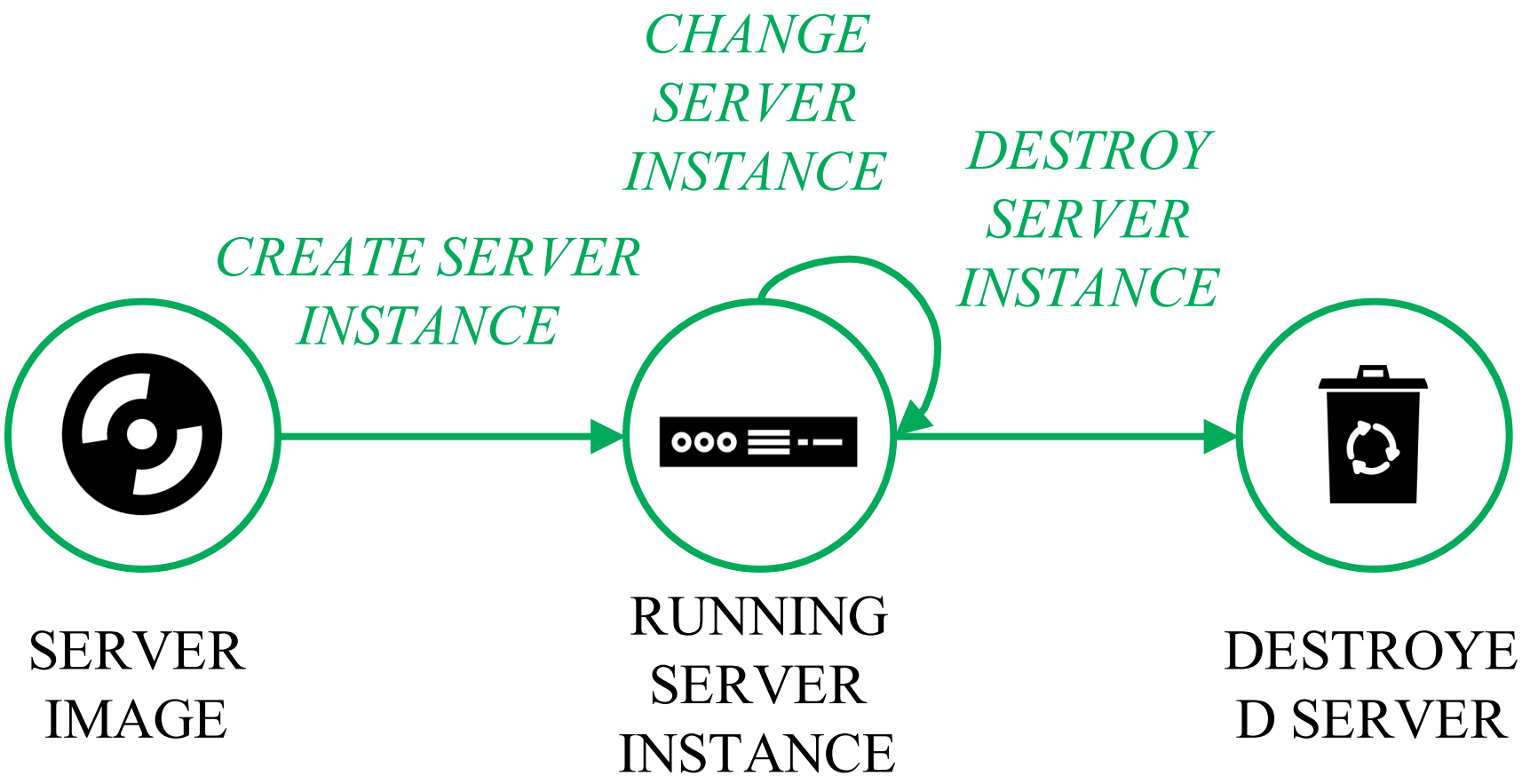
IMPLEMENTATION :

Similar environment, orchestration, and tools as those used to test server configuration modules



SERVER LIFECYCLE

SERVER LIFECYCLE

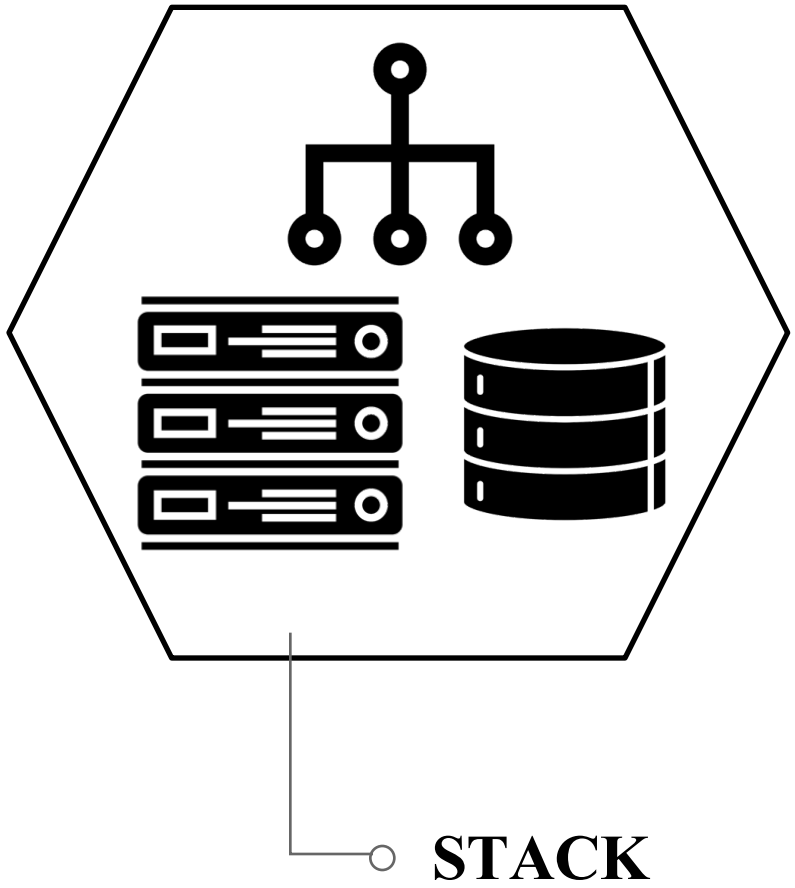


CREATING NEW SERVER INSTANCES

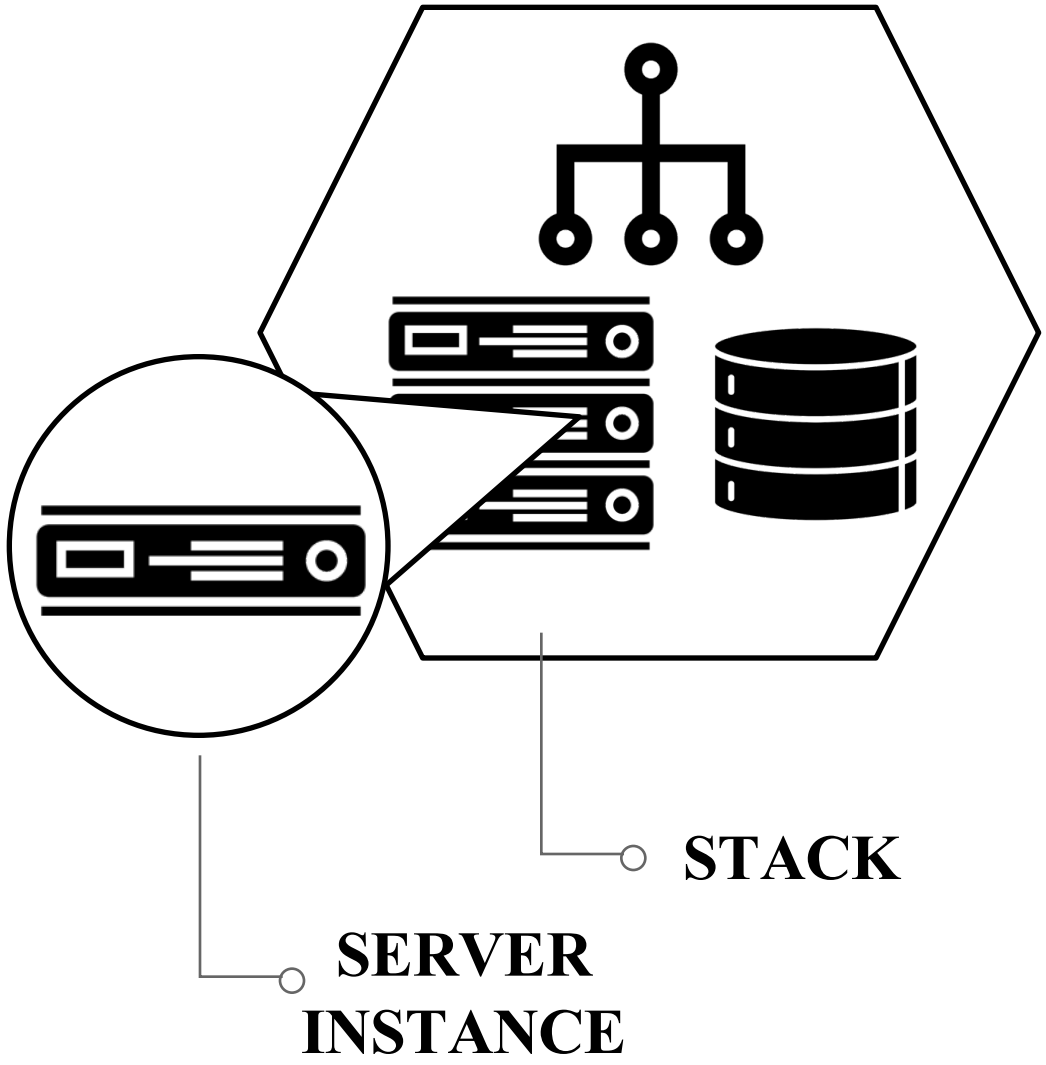
Different ways to create an instance

- Hand-build
- Script
- From stack code
- Automatically (e.g. auto-scaling)
- Boot-provisioning (e.g. PXE)

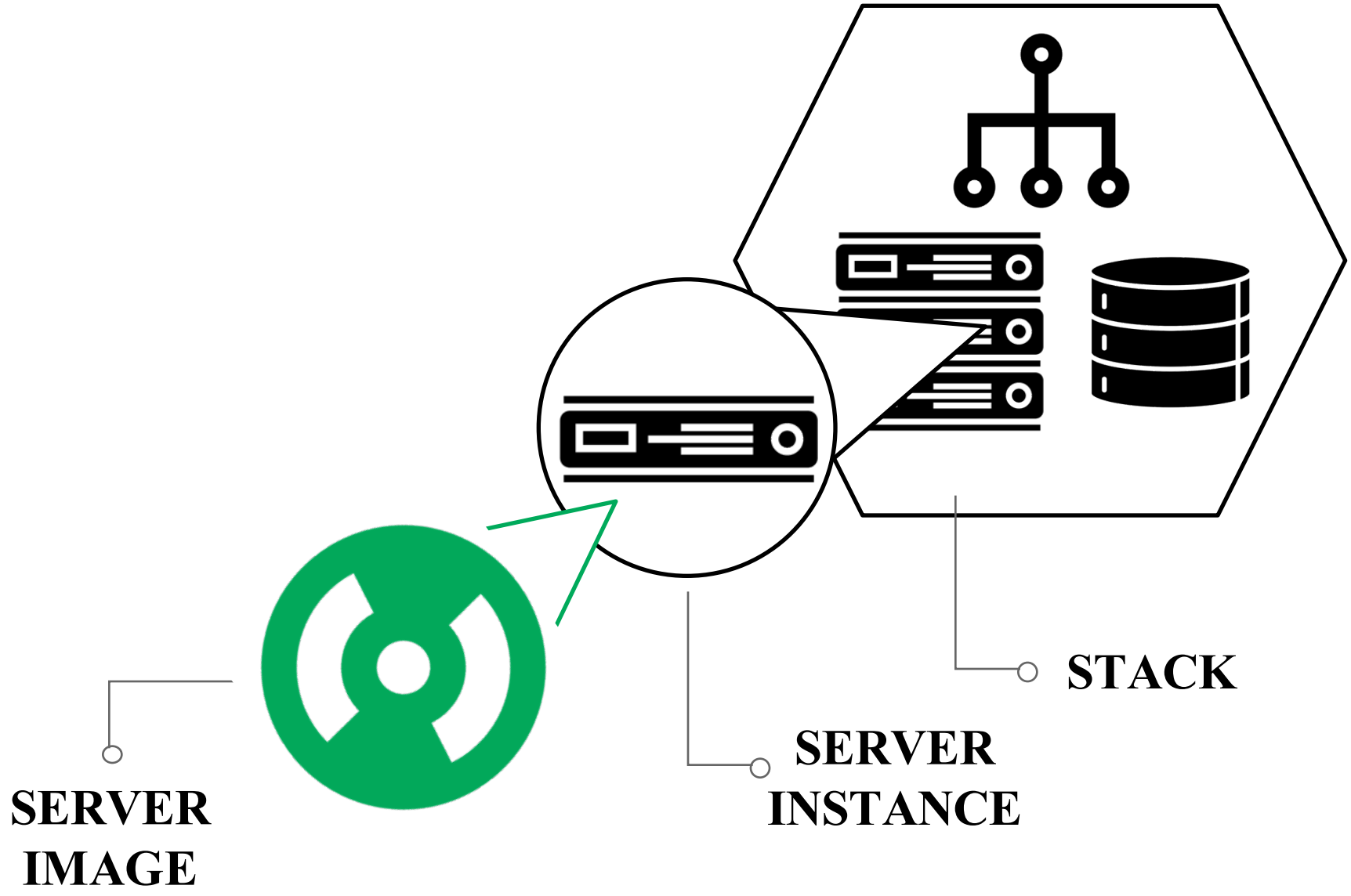
PROVISIONING SERVERS FROM A STACK



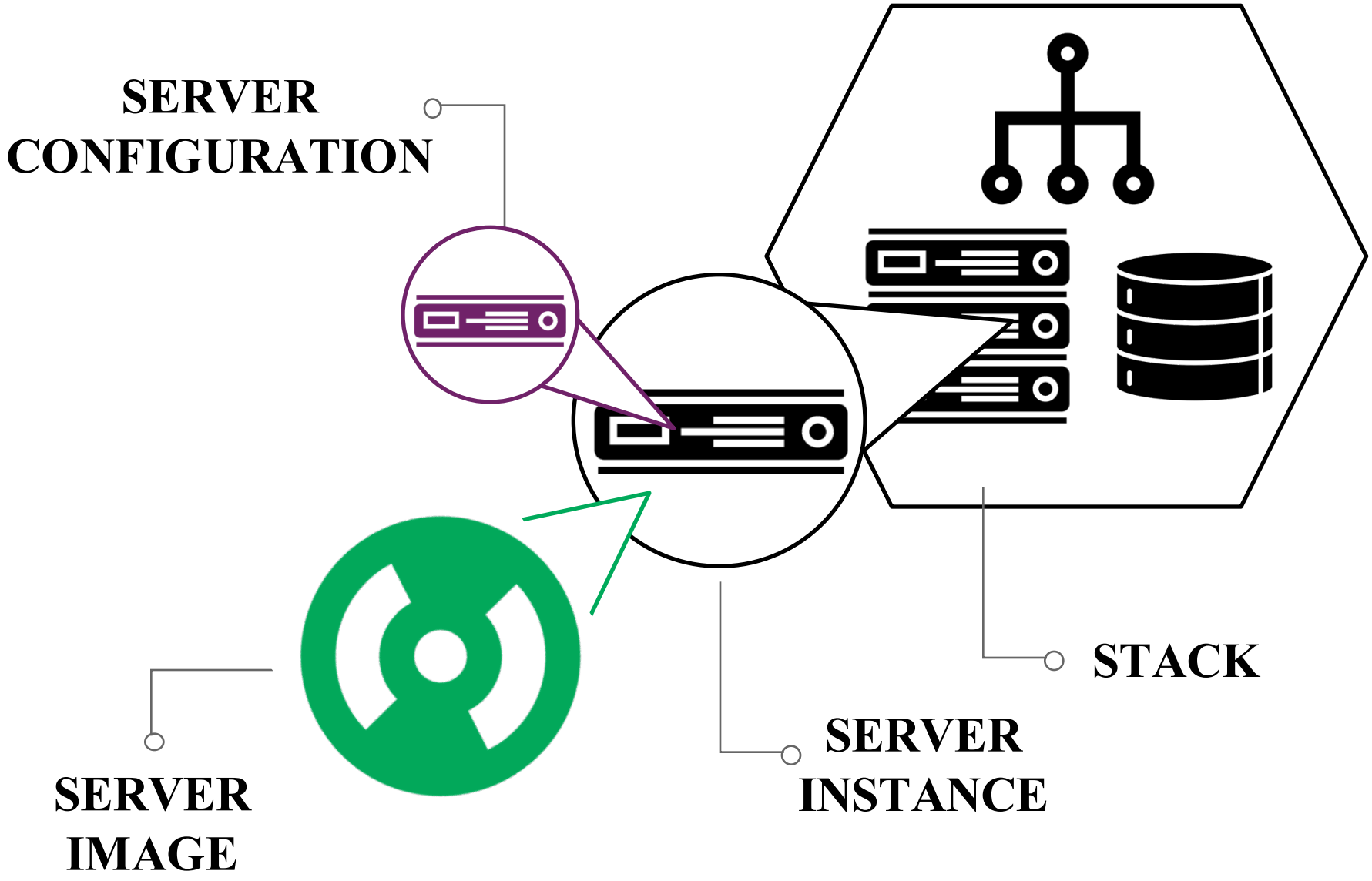
PROVISIONING SERVERS FROM A STACK



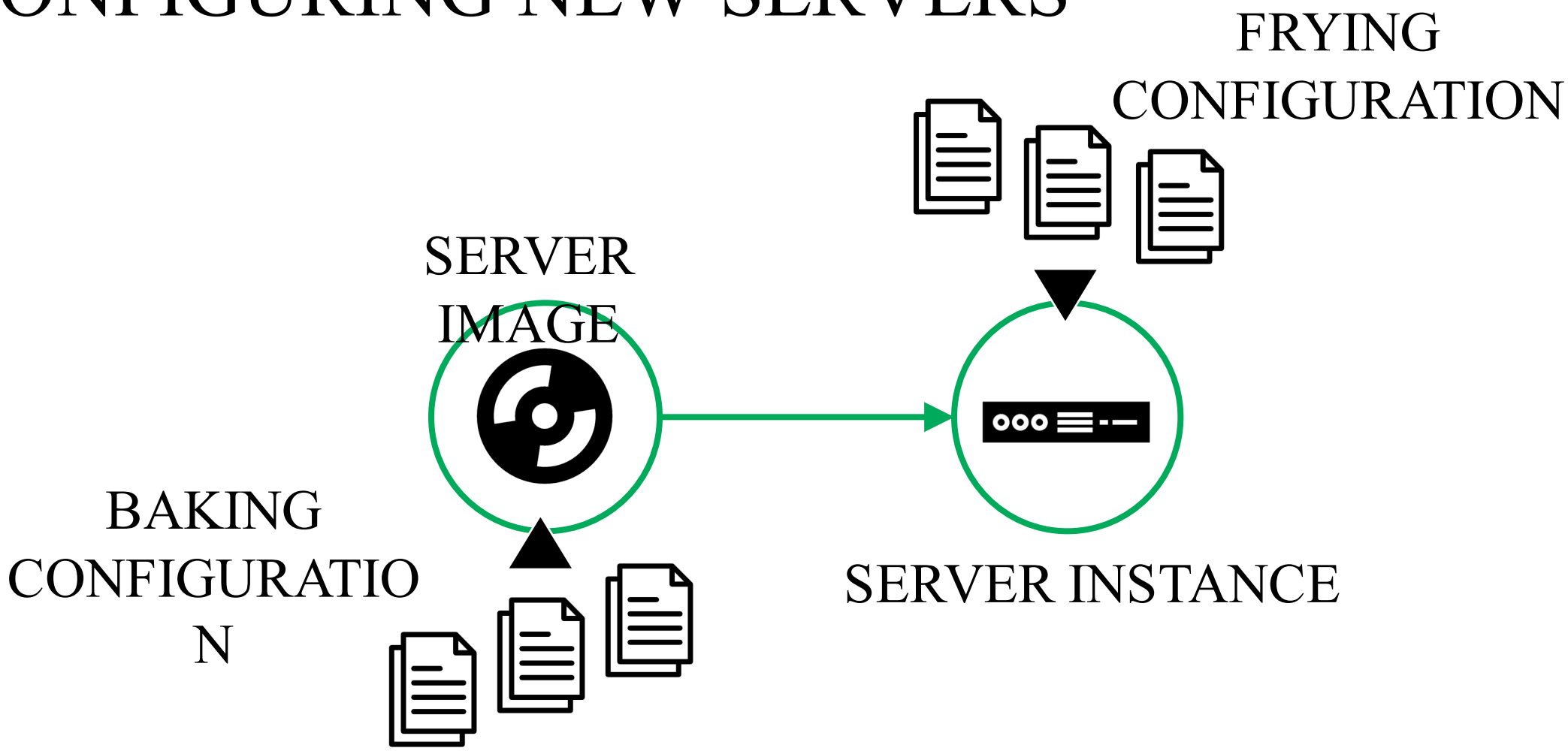
PROVISIONING SERVERS FROM A STACK



PROVISIONING SERVERS FROM A STACK



CONFIGURING NEW SERVERS

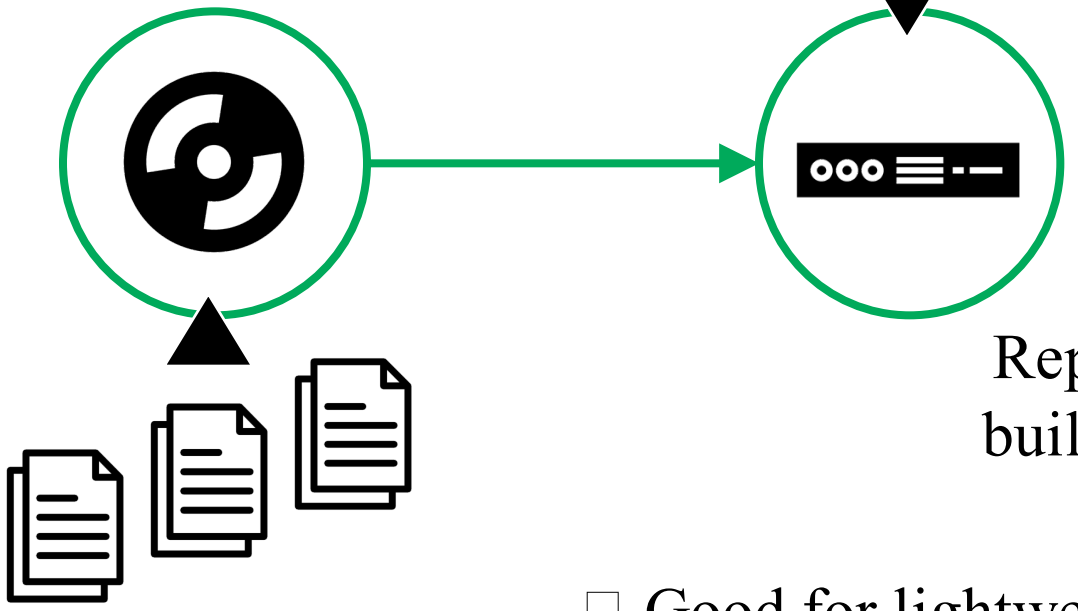


BAKING

Longer cycle time for changes

Build once, test once, use many

 Good for heavy things that change less often



FRYING

Faster cycle time for changes

Repeat the same build activities

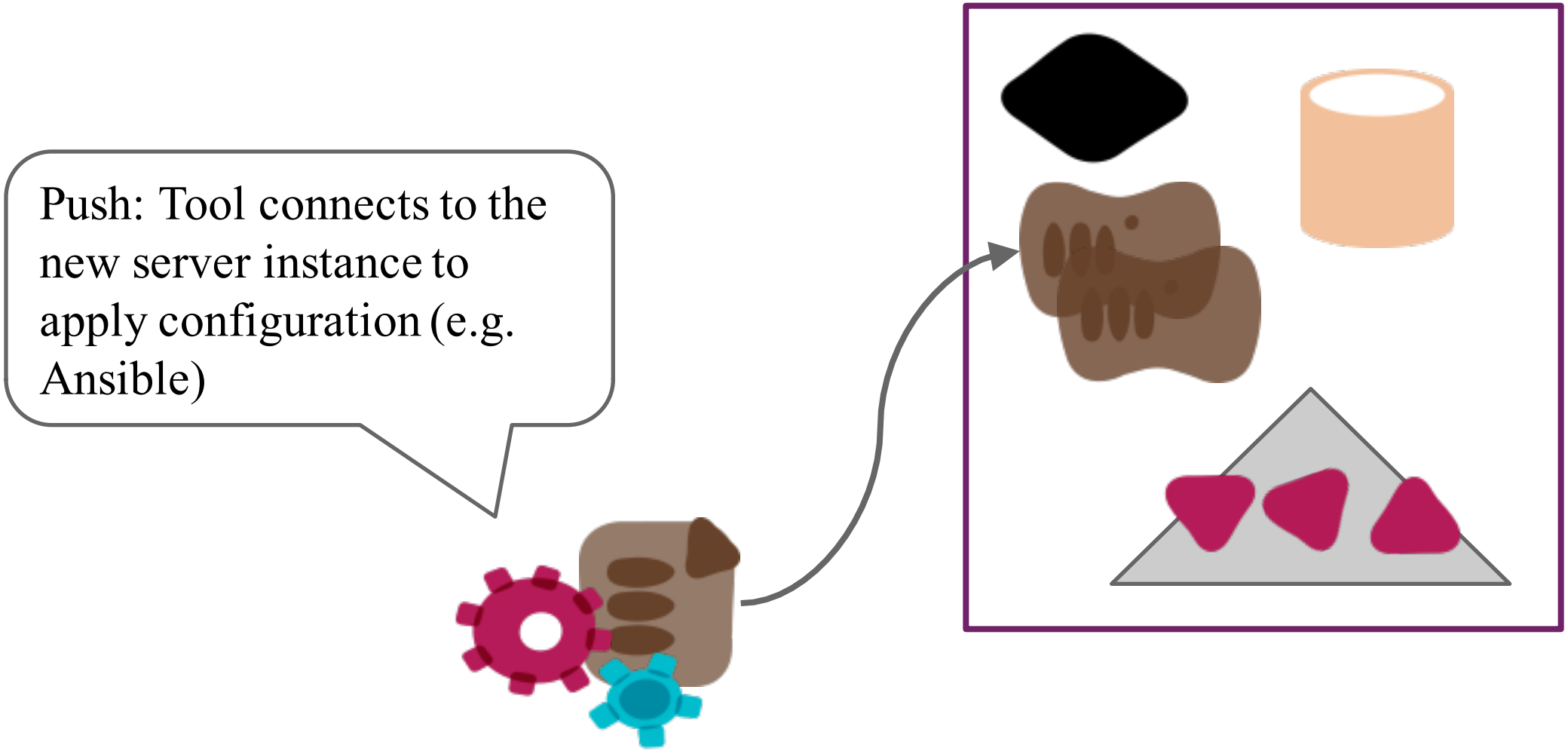
 Good for lightweight things that change more often

CONFIGURING NEW SERVER INSTANCES

Push versus Pull

- Should your tool connect to a newly created server to configure it?
- Or should you have a pre-installed agent that pulls configuration when the new instance starts up?

PUSH TO CONFIGURE

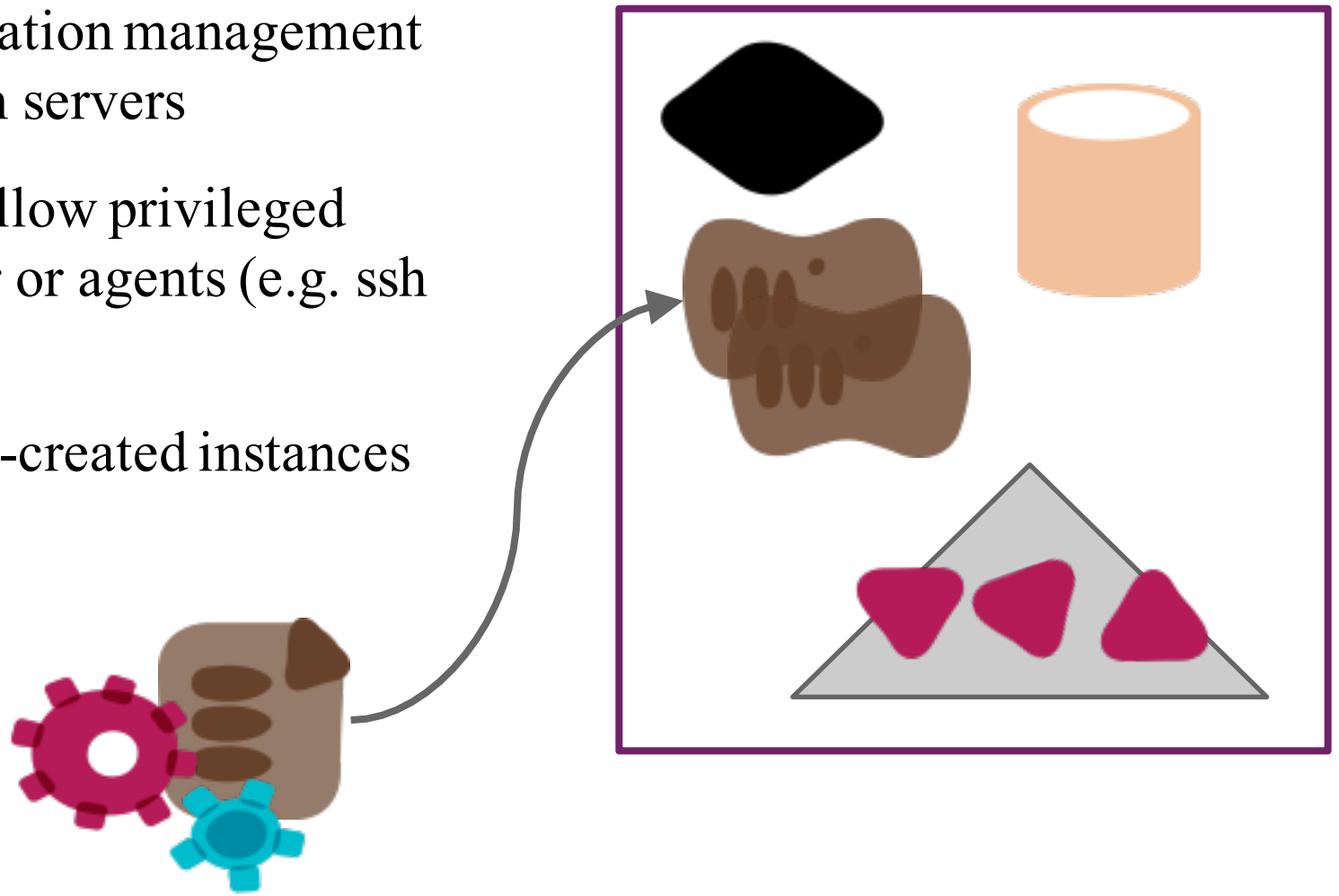


PUSH TO CONFIGURE

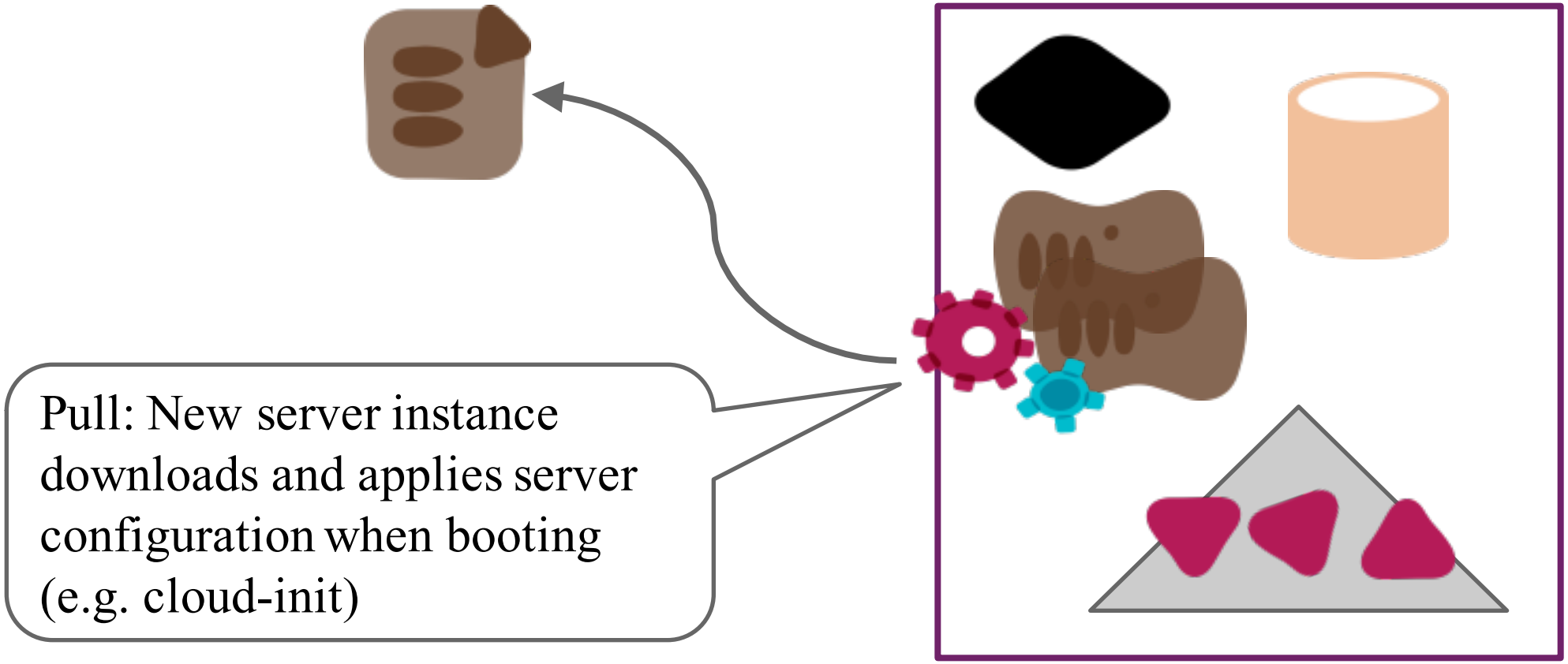
Does not require configuration management software to be installed on servers

Requires new servers to allow privileged access from central server or agents (e.g. ssh key pre-installed)

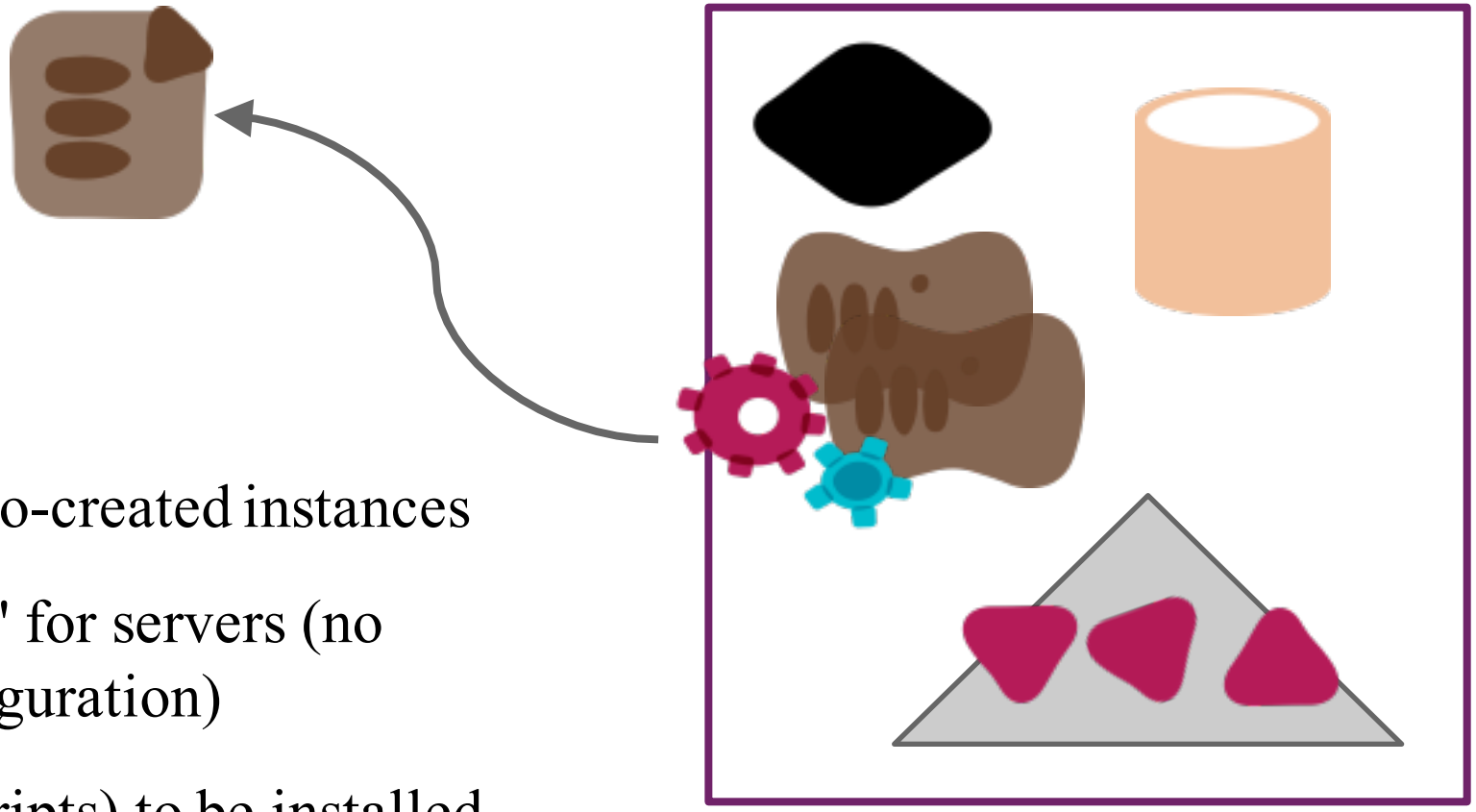
Awkward to apply to auto-created instances (e.g. for auto-scaling)



PULL TO CONFIGURE



PULL TO CONFIGURE

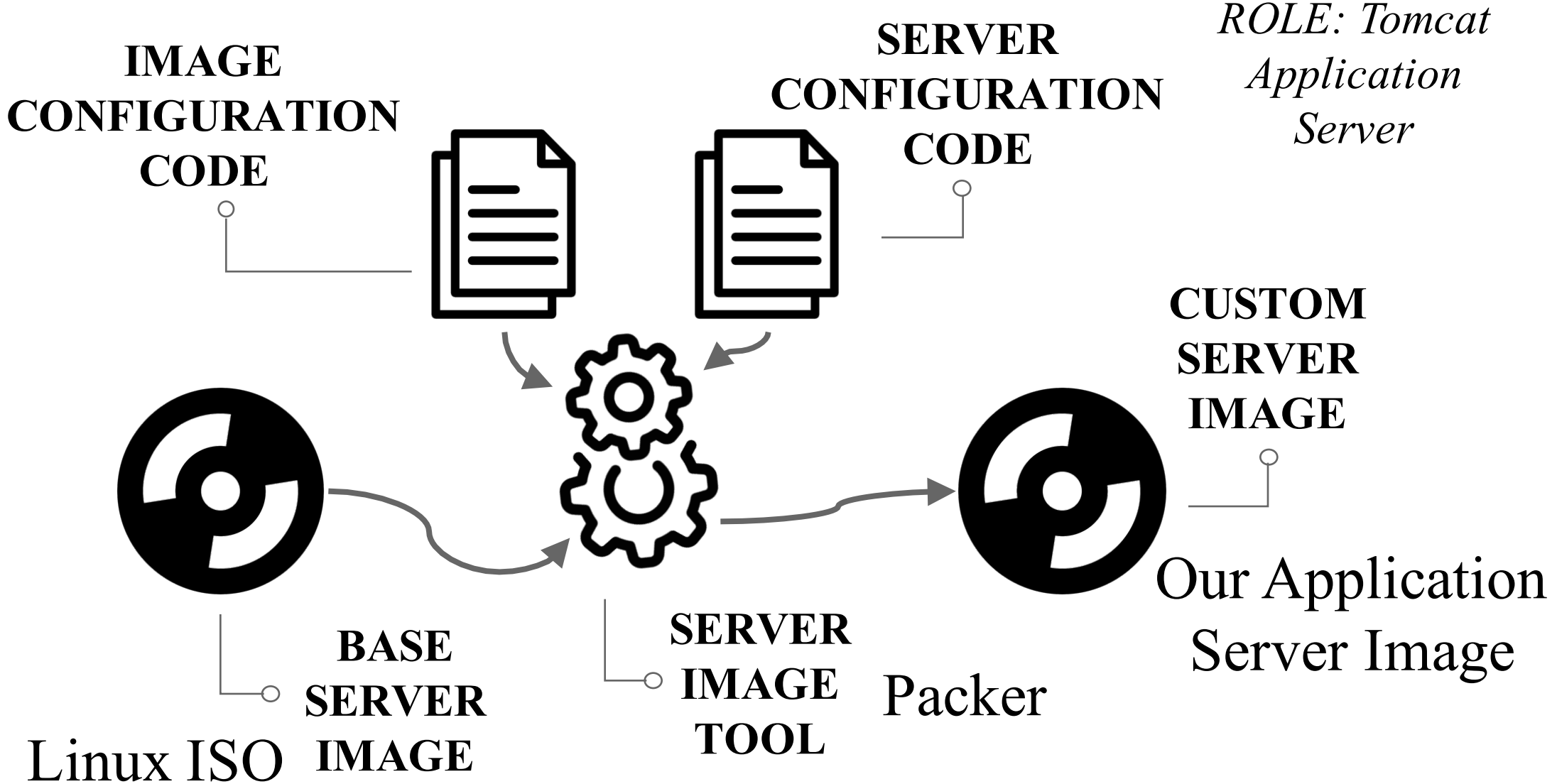


Works naturally with auto-created instances

Reduced "attack surface" for servers (no inbound access for configuration)

Requires software (or scripts) to be installed on new server instances

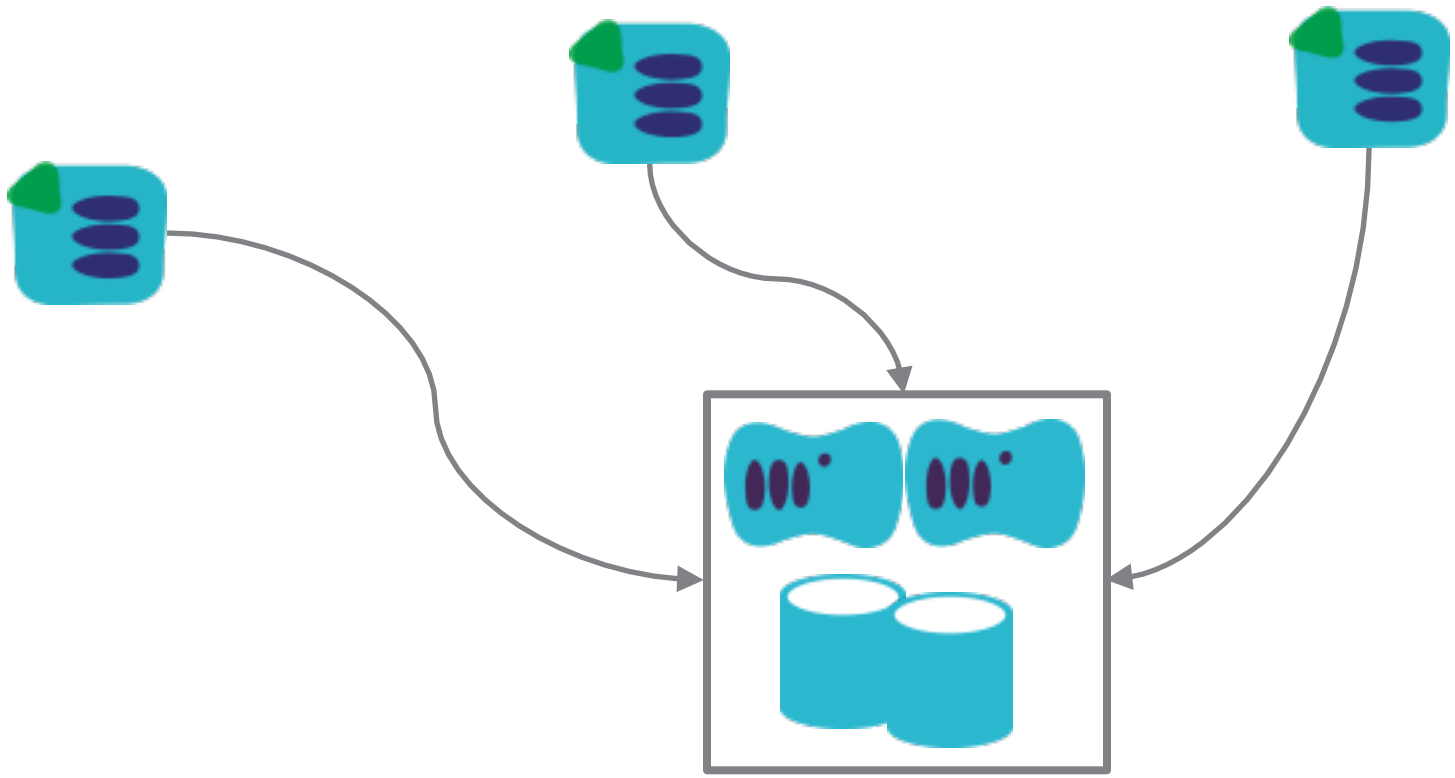
BUILDING A SERVER IMAGE



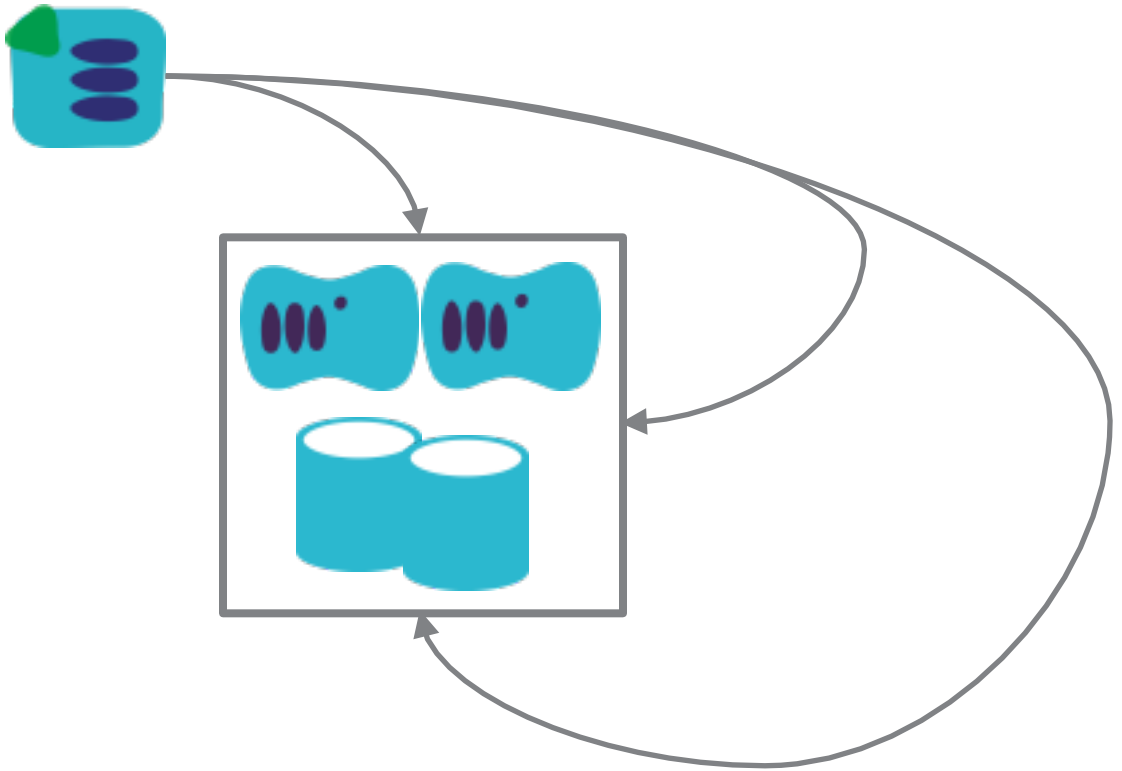
CHANGING SERVERS

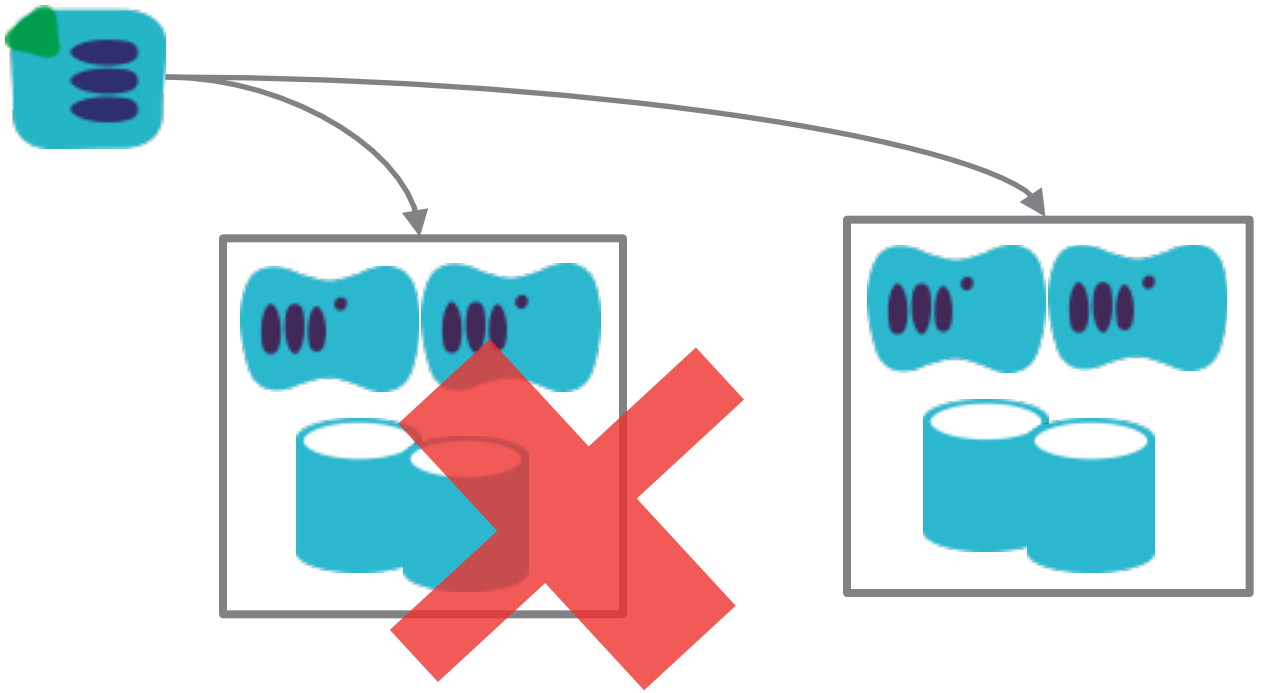
- Ad-hoc changes
- Continuous synchronization
- Rebuild continuously
- Immutable servers

AD-HOC CHANGES

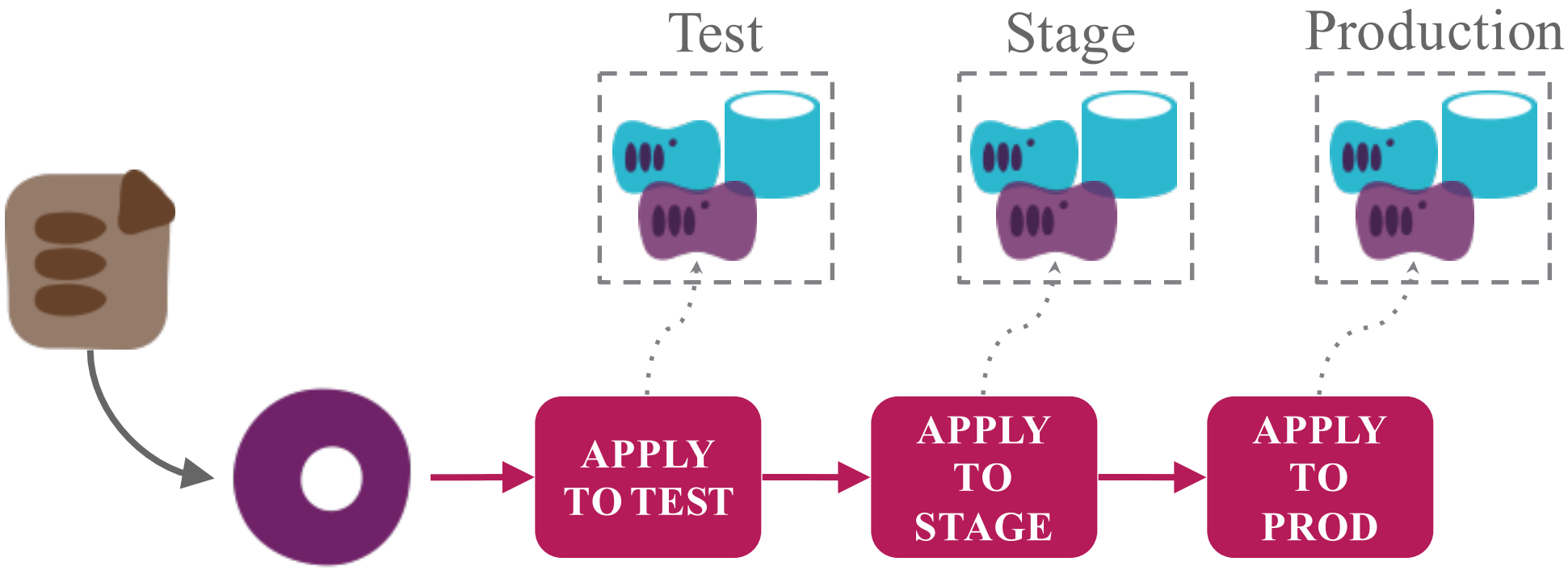


CONTINUOUS SYNCHRONIZATION





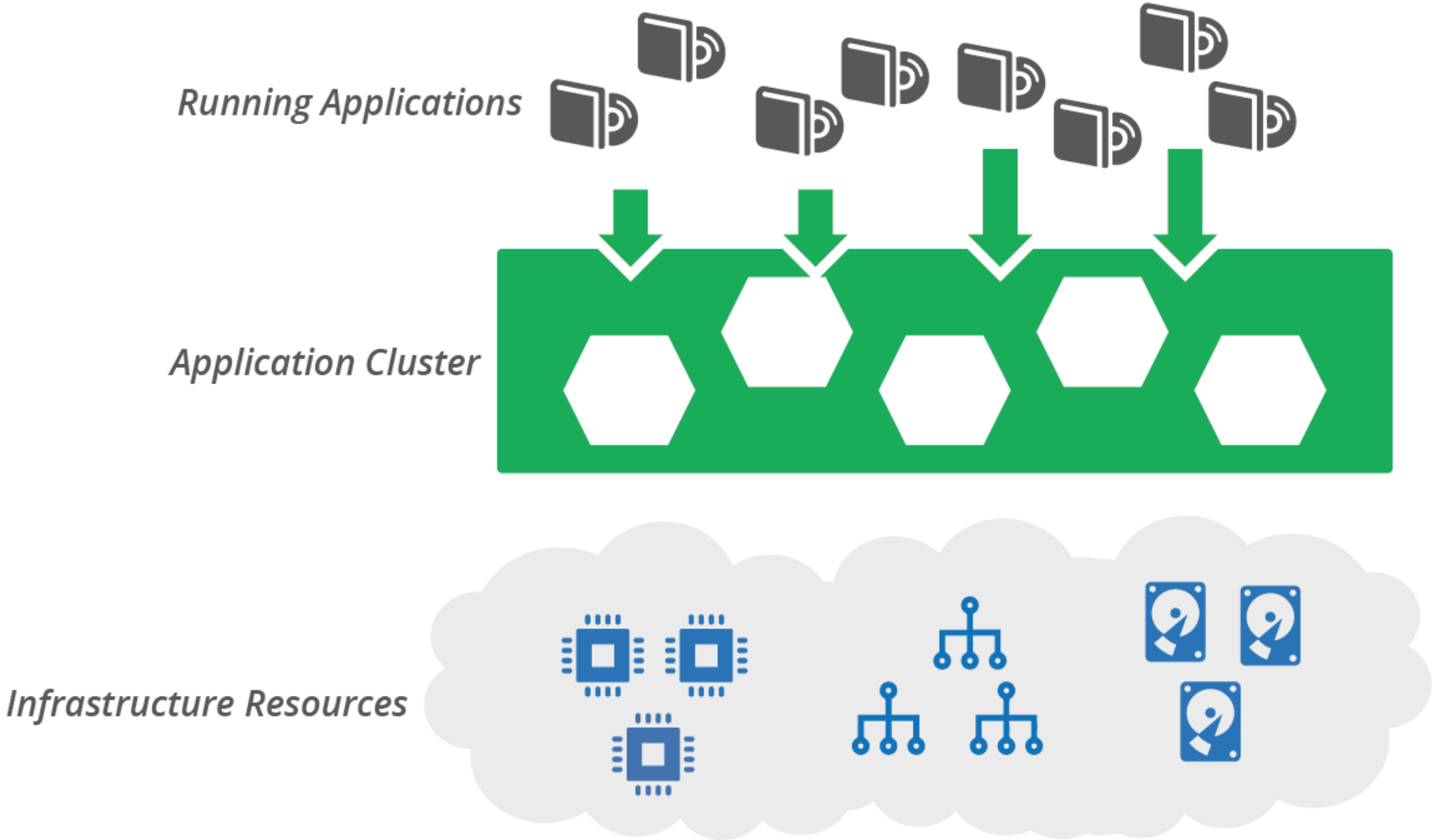
REBUILD CONTINUOUSLY



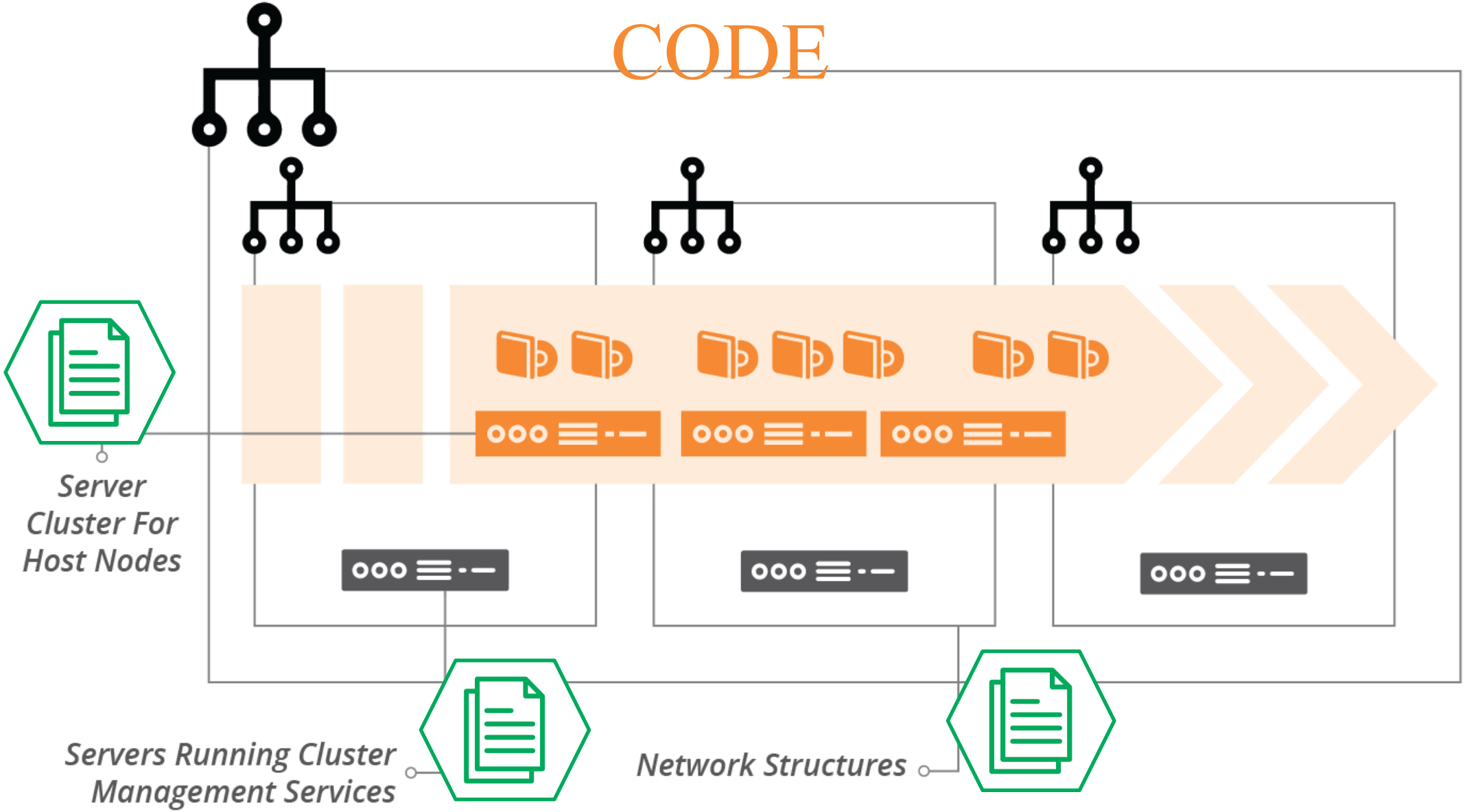
IMMUTABLE SERVERS

Change by rebuilding

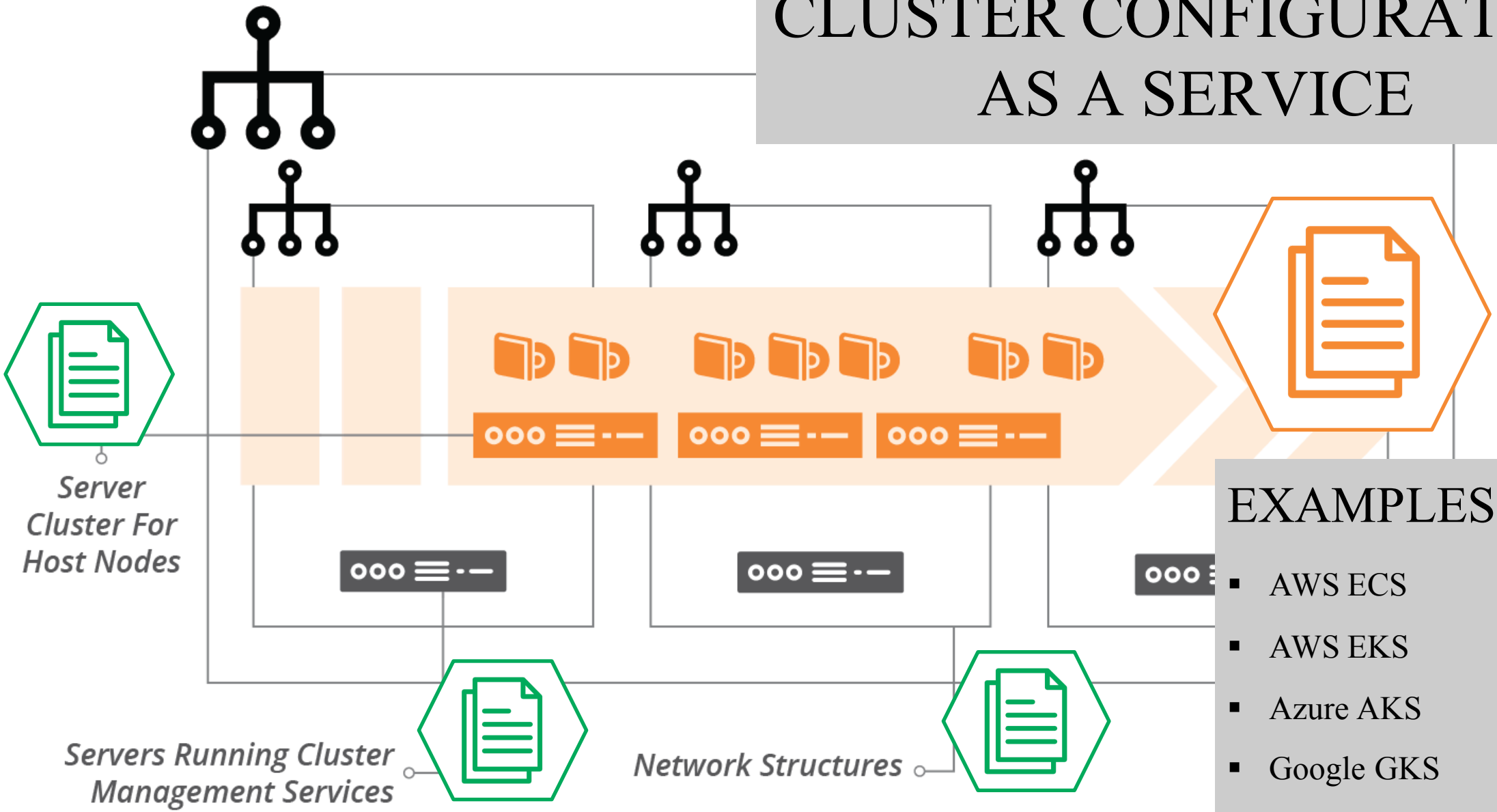

CLUSTERS AS CODE


Running Applications
Application Cluster
Infrastructure Resources

CLUSTERS AS CODE



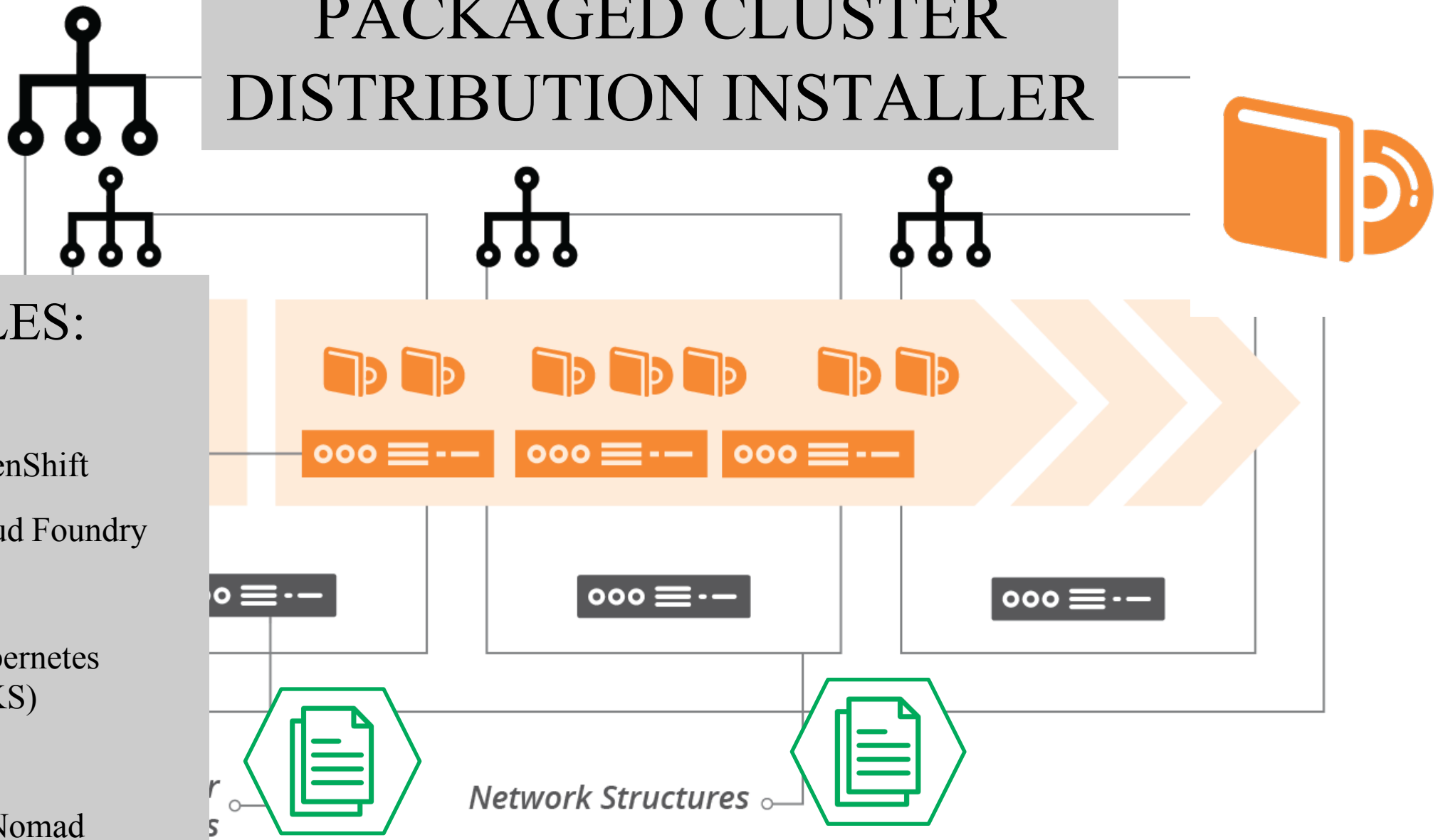
CLUSTER CONFIGURATION AS A SERVICE



EXAMPLES:

- AWS ECS
- AWS EKS
- Azure AKS
- Google GKS

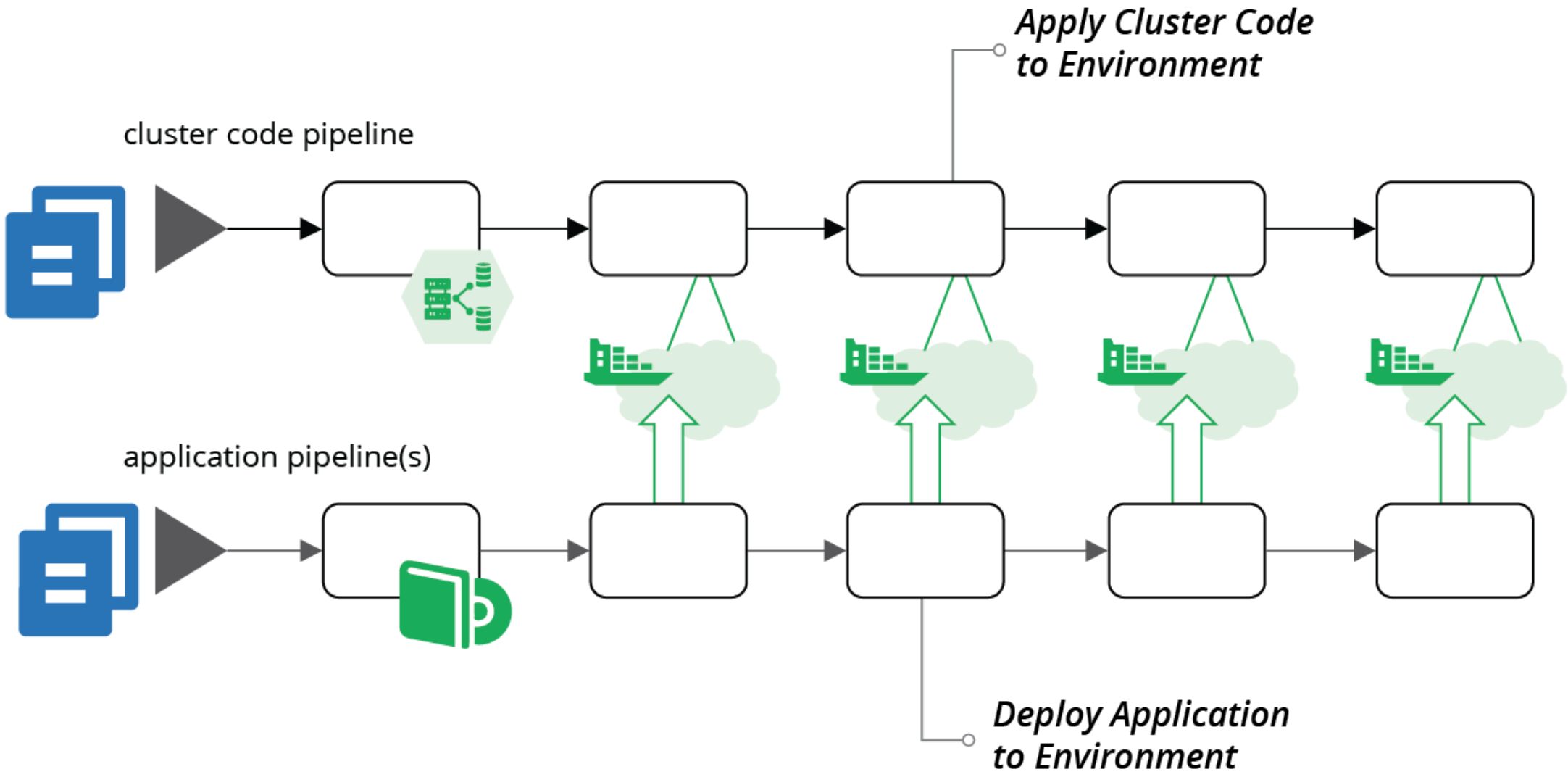
PACKAGED CLUSTER DISTRIBUTION INSTALLER



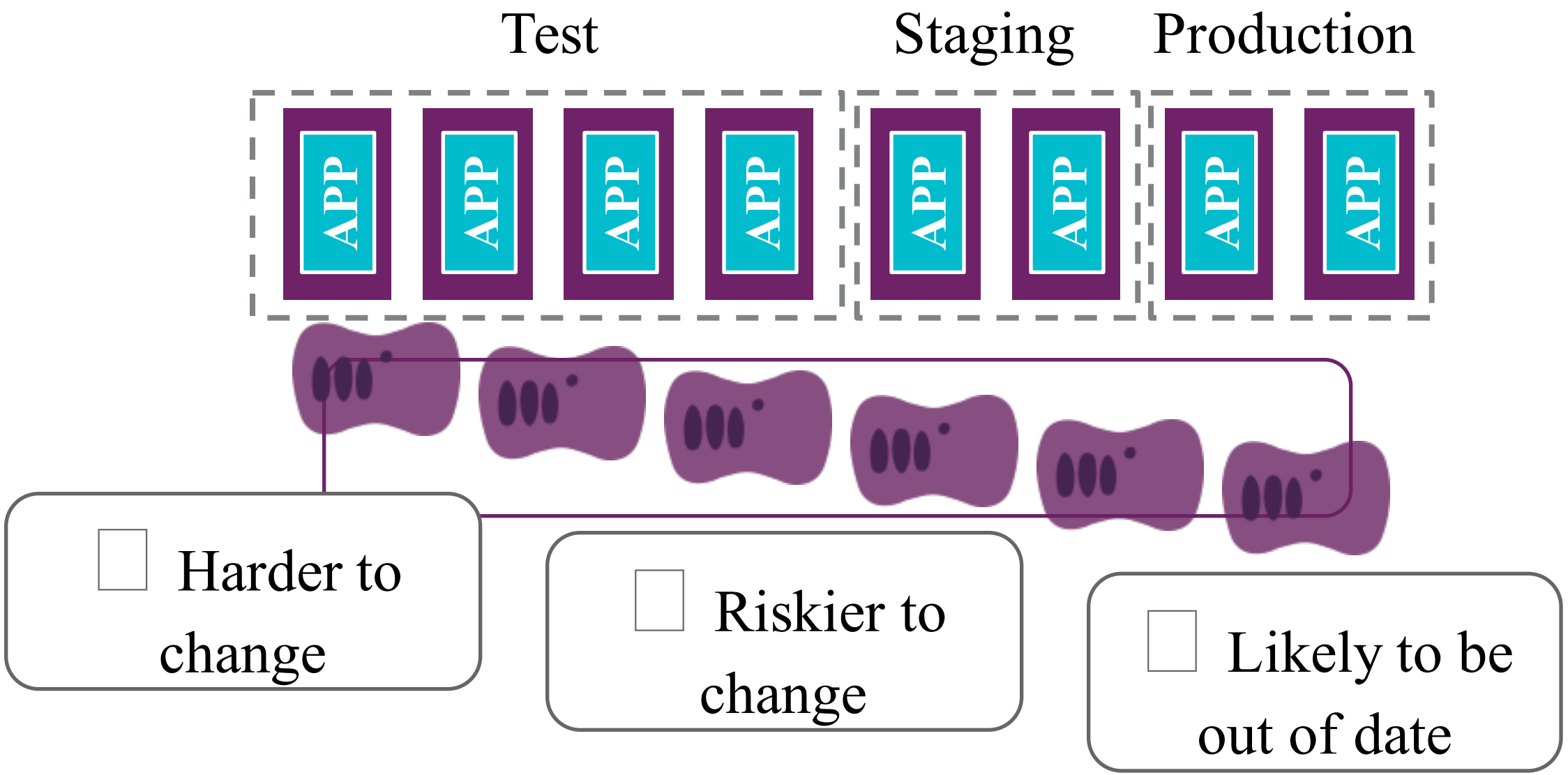
EXAMPLES:

- Kubadm
- RedHat OpenShift
- Pivotal Cloud Foundry
- Rancher
- Docker Kubernetes Service (DKS)
- Mesosphere
- Hashicorp Nomad

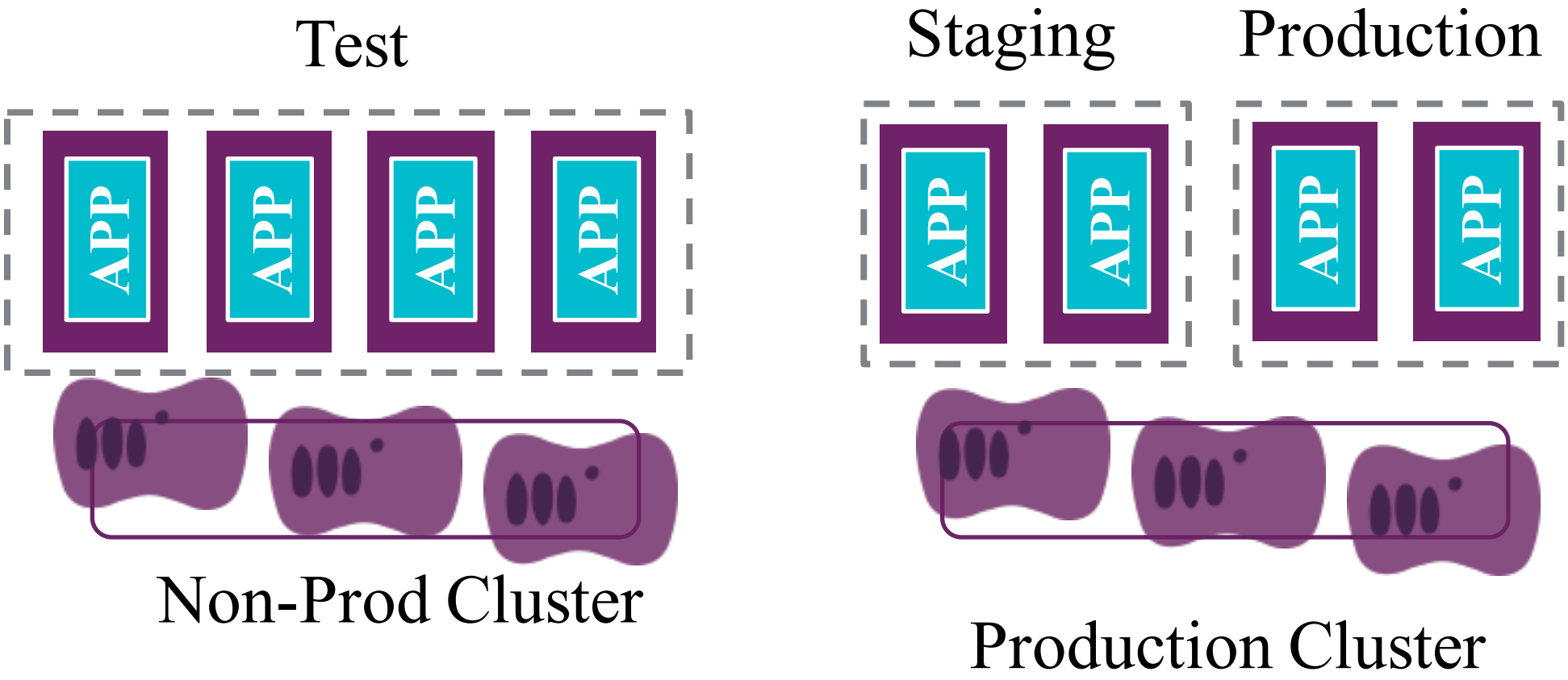
Use a pipeline to deliver changes to the cluster



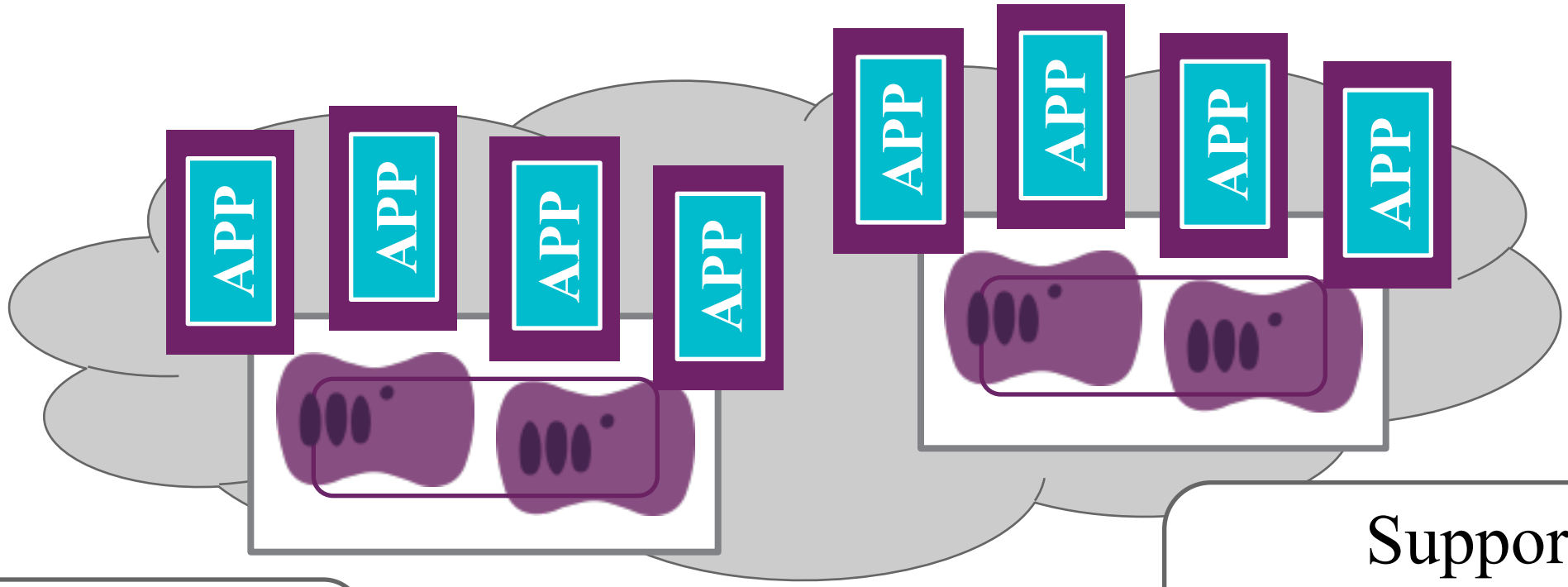
Monolithic Container Cluster



Environment Clusters



Team clusters (per environment)



Avoid "big bang" updates

Enable per-team configuration

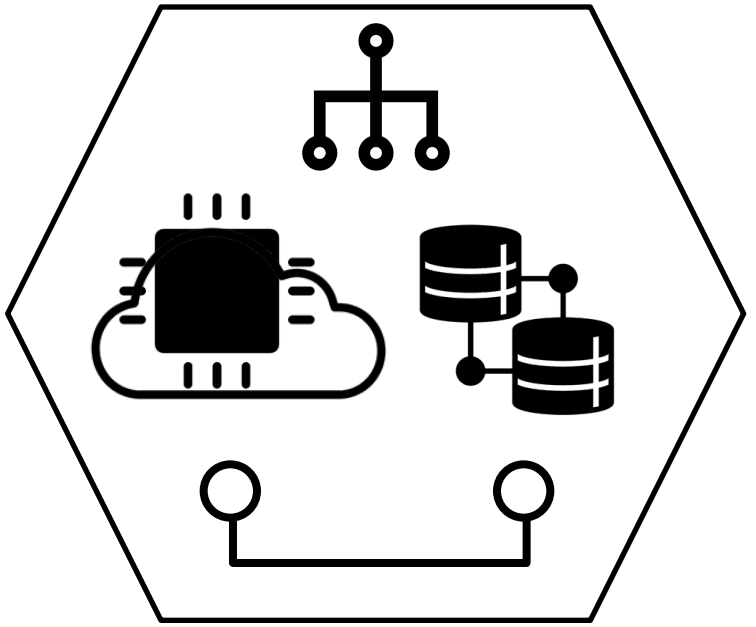
Support governance boundaries

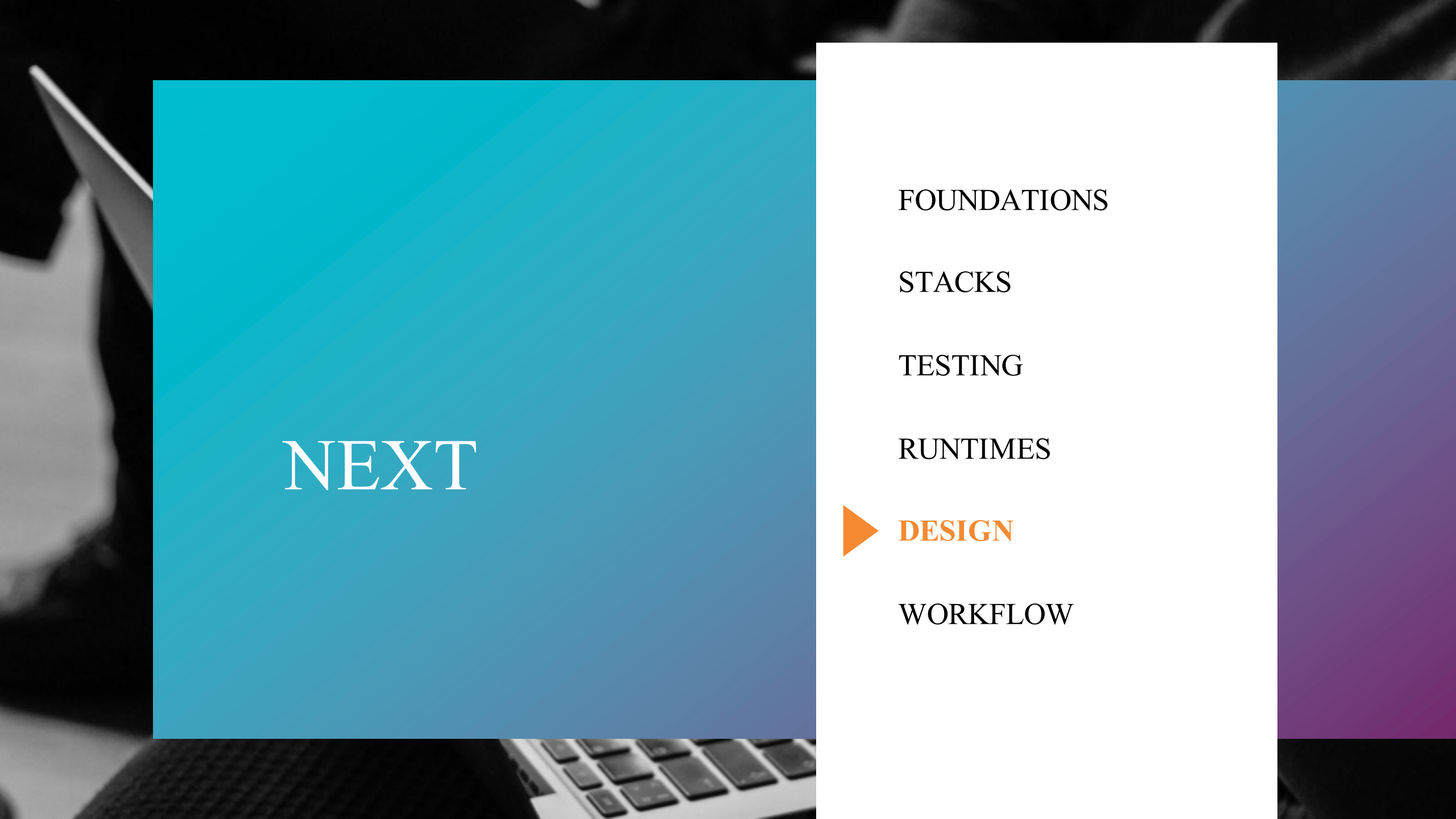
SERVERLESS INFRASTRUCTURE AS CODE

AN EXAMPLE STACK

INCLUDES

- Network Routing
- Data Storage
- Message Queues





NEXT

FOUNDATIONS

STACKS

TESTING

RUNTIMES

DESIGN

WORKFLOW