anynines

DevOps Challenge - Experience Report

Horror and Success Story

Noah Ispas

anynines

Noah Ispas

- Software Engineer @anynines

Noah Ispas

- Software Engineer @anynines
- Used to work with Java (monoliths)

anynines

Noah Ispas

- Software Engineer @anynines
- Used to work with Java (monoliths)
- Got to know Node.js and Microservices

anynines

Noah Ispas

- Software Engineer @anynines
- Used to work with Java (monoliths)
- Got to know Node.js and Microservices
- Helping customer getting their Software in the cloud

anynines

Noah Ispas

- Software Engineer @anynines
- Used to work with Java (monoliths)
- Got to know Node.js and Microservices
- Helping customer getting their Software in the cloud
- Developing custom software around Cloud Foundry

anynines

Agenda

anynines

Agenda

- Horror Story

anynines

Agenda

- Horror Story
  - Initial Situation

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?
  - Chaos Prevention

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?
  - Chaos Prevention
- Success Story

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?
  - Chaos Prevention
- Success Story
  - Initial Situation

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?
  - Chaos Prevention
- Success Story
  - Initial Situation
  - Requirements/Constraints

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?
  - Chaos Prevention
- Success Story
  - Initial Situation
  - Requirements/Constraints
  - Challenges

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?
  - Chaos Prevention
- Success Story
  - Initial Situation
  - Requirements/Constraints
  - Challenges
  - Solutions

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?
  - Chaos Prevention
- Success Story
  - Initial Situation
  - Requirements/Constraints
  - Challenges
  - Solutions
  - Recap

anynines

Agenda

- Horror Story
  - Initial Situation
  - Transformation
  - Big Bang
  - What went wrong?
  - Chaos Prevention
- Success Story
  - Initial Situation
  - Requirements/Constraints
  - Challenges
  - Solutions
  - Recap
- Lessons Learned/Advices

anynines

Horror Story

anynines

Initial Situation

anynines

Initial Situation

- Small Company

anynines

Initial Situation

- Small Company
- Java EE Monolith

anynines

Initial Situation

- Small Company
- Java EE Monolith
- Very old legacy component

anynines

Initial Situation

- Small Company
- Java EE Monolith
- Very old legacy component
- Small test suites

anynines

Initial Situation

- Small Company
- Java EE Monolith
- Very old legacy component
- Small test suites
- Classic Dev and Ops silo

anynines

Initial Situation

anynines

Initial Situation

Developers

Operators

anynines

Initial Situation

Developers

Java Monolith

Operators

anynines

Initial Situation

Developers

Operators

Java Monolith

anynines

Initial Situation

Knowledge Barrier

Developers

Operators

Java Monolith

anynines

“

”

anynines

“

You build it, you run it

”

anynines

“

”

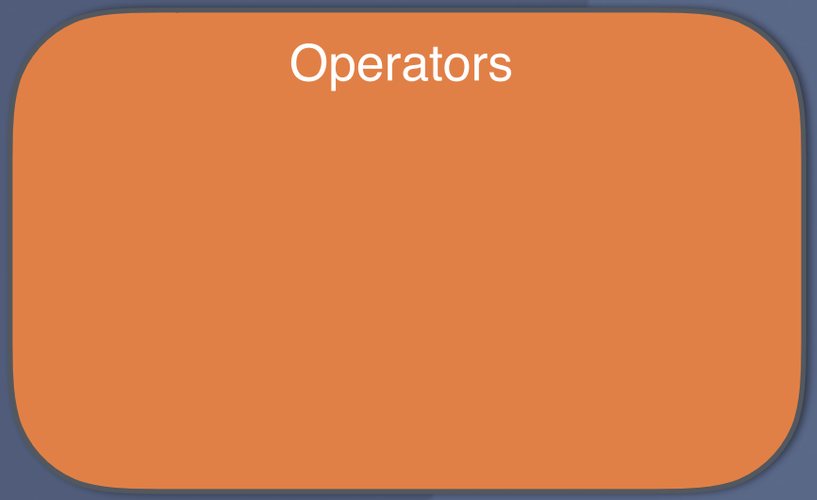
anynines

Initial Situation

Bare Metal

anynines

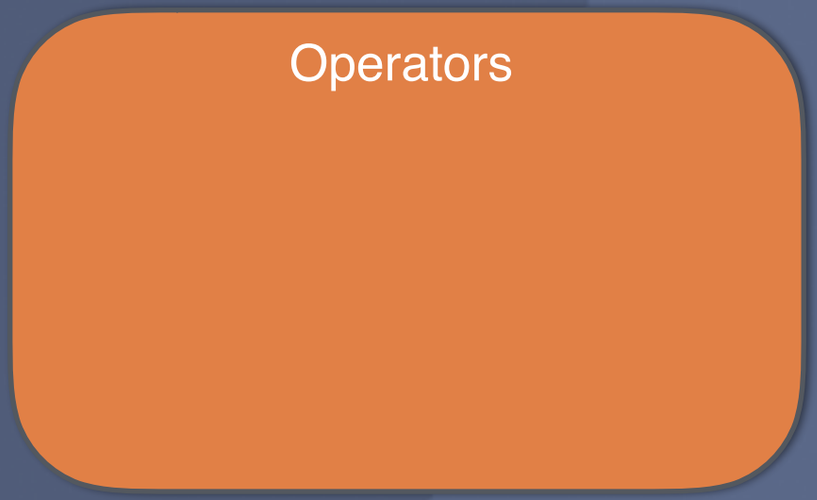
Initial Situation



Bare Metal

anynines

Initial Situation



Bare Metal

Performance Issues

anynines

Transformation

Operators

Bare Metal

JBoss

Java Monolith

anynines

Transformation

Operators

Bare Metal

JBoss

Java Monolith

Performance Limit

anynines

Transformation

Operators

Bare Metal

JBoss

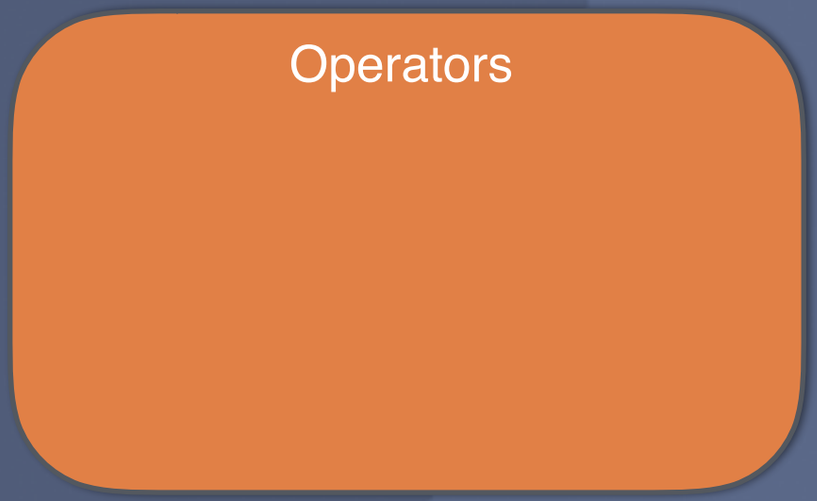
Java Monolith

Performance Limit

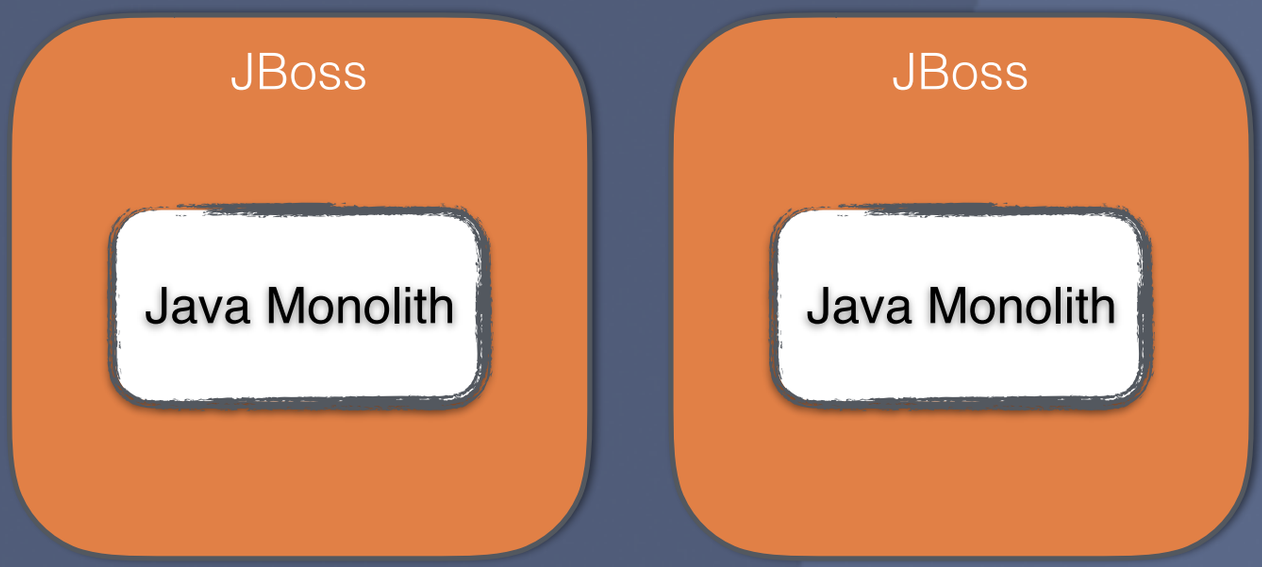
No Resilience

anynines

Transformation

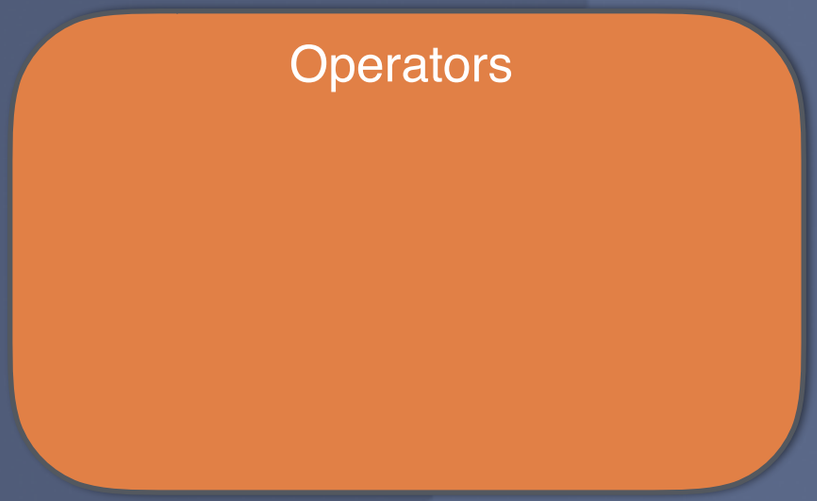


Bare Metal

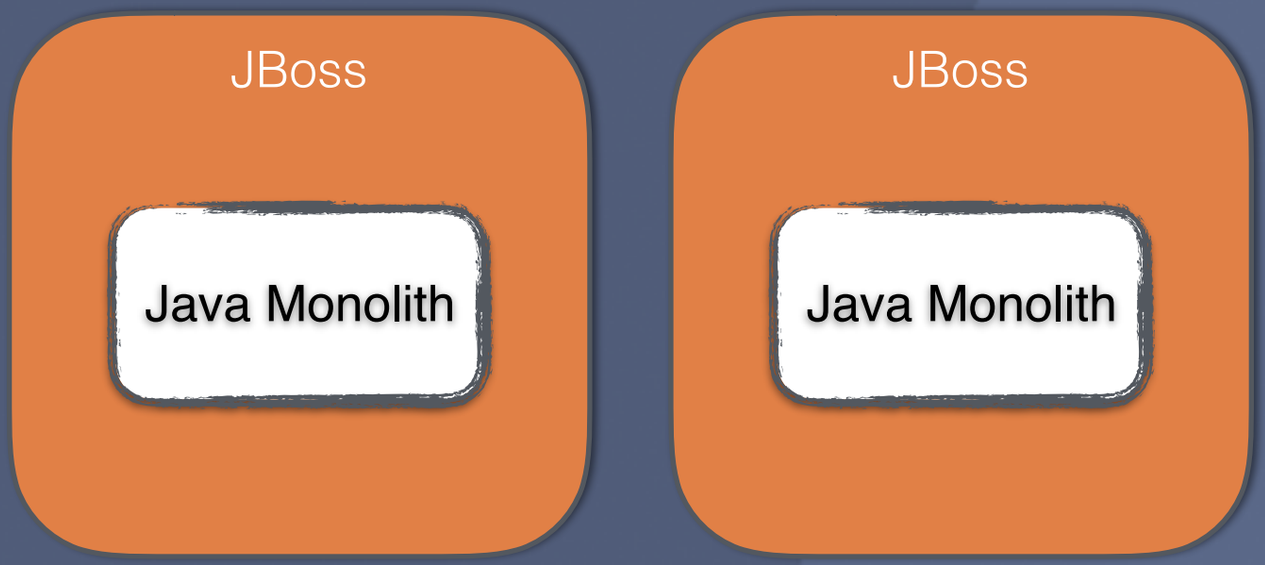


anynines

Transformation



Bare Metal



Singletons

anynines

Transformation

Knowledge Barrier

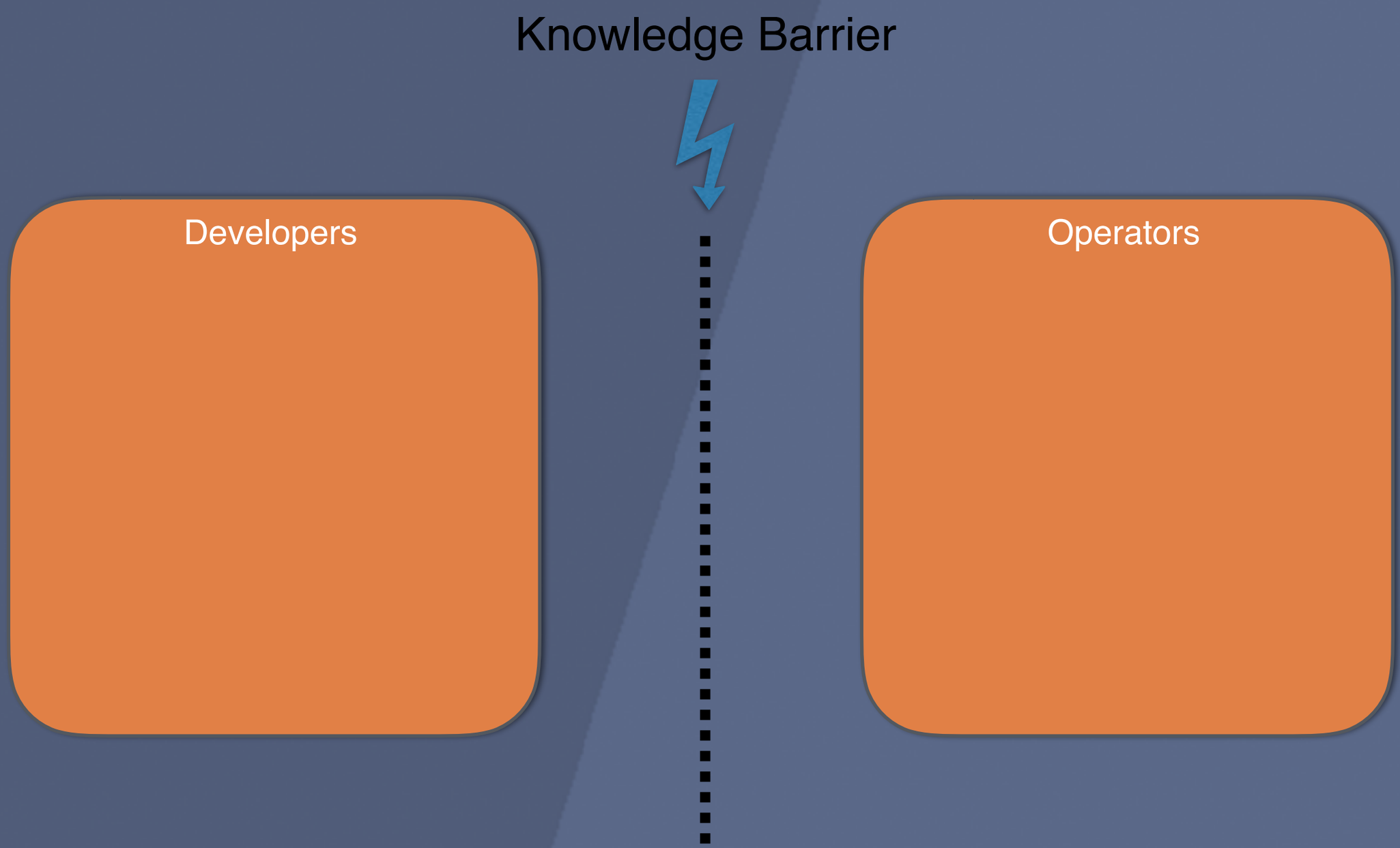
Developers

Operators

Java Monolith

anynines

Transformation



anynines

Transformation

Knowledge Barrier

Developers

Operators

Java Monolith

anynines

Transformation

Operators

Bare Metal

anynines

Transformation

Operators

Bare Metal

JBoss

Java Monolith

JBoss

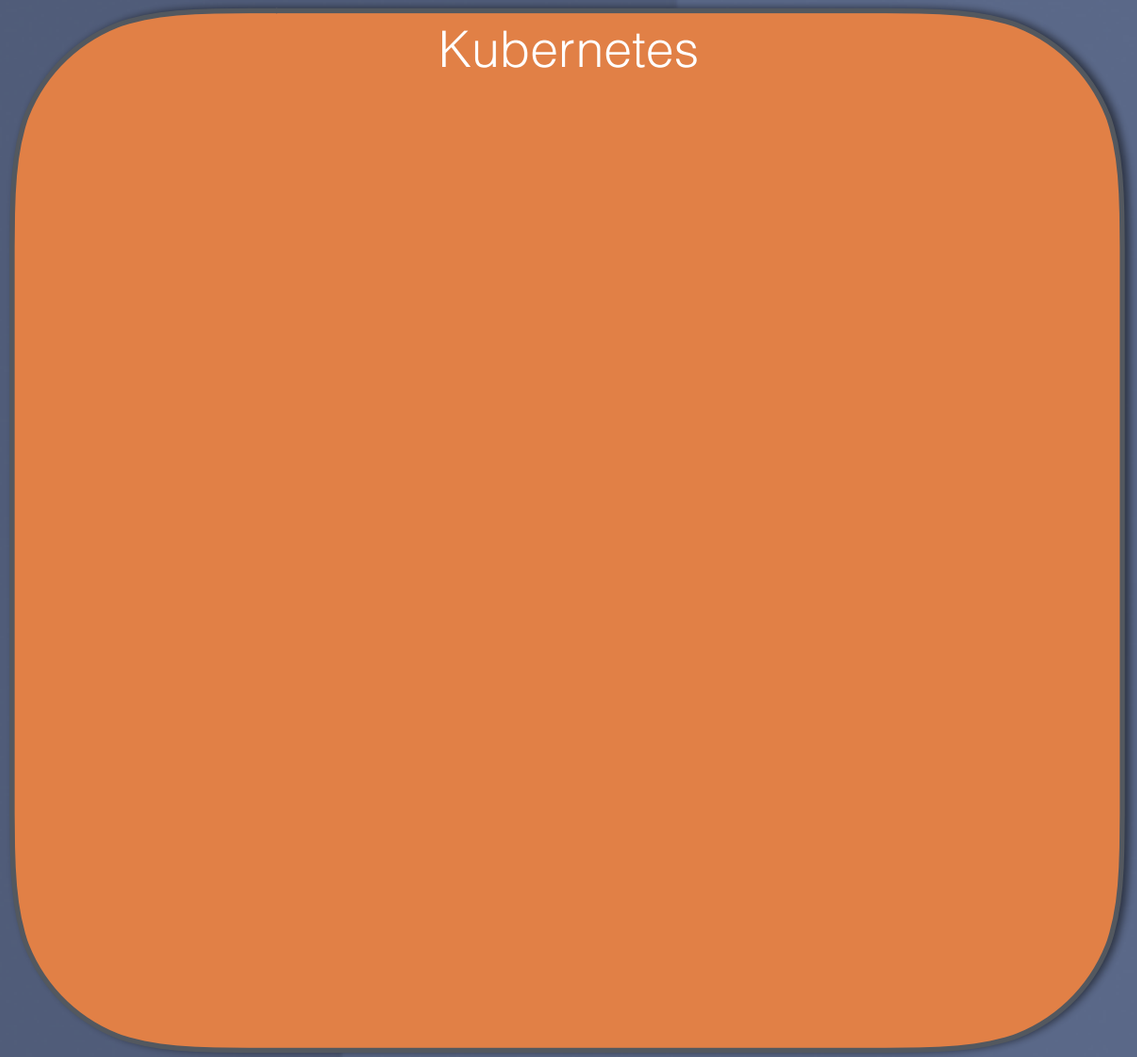
Java Monolith

Kubernetes

DB

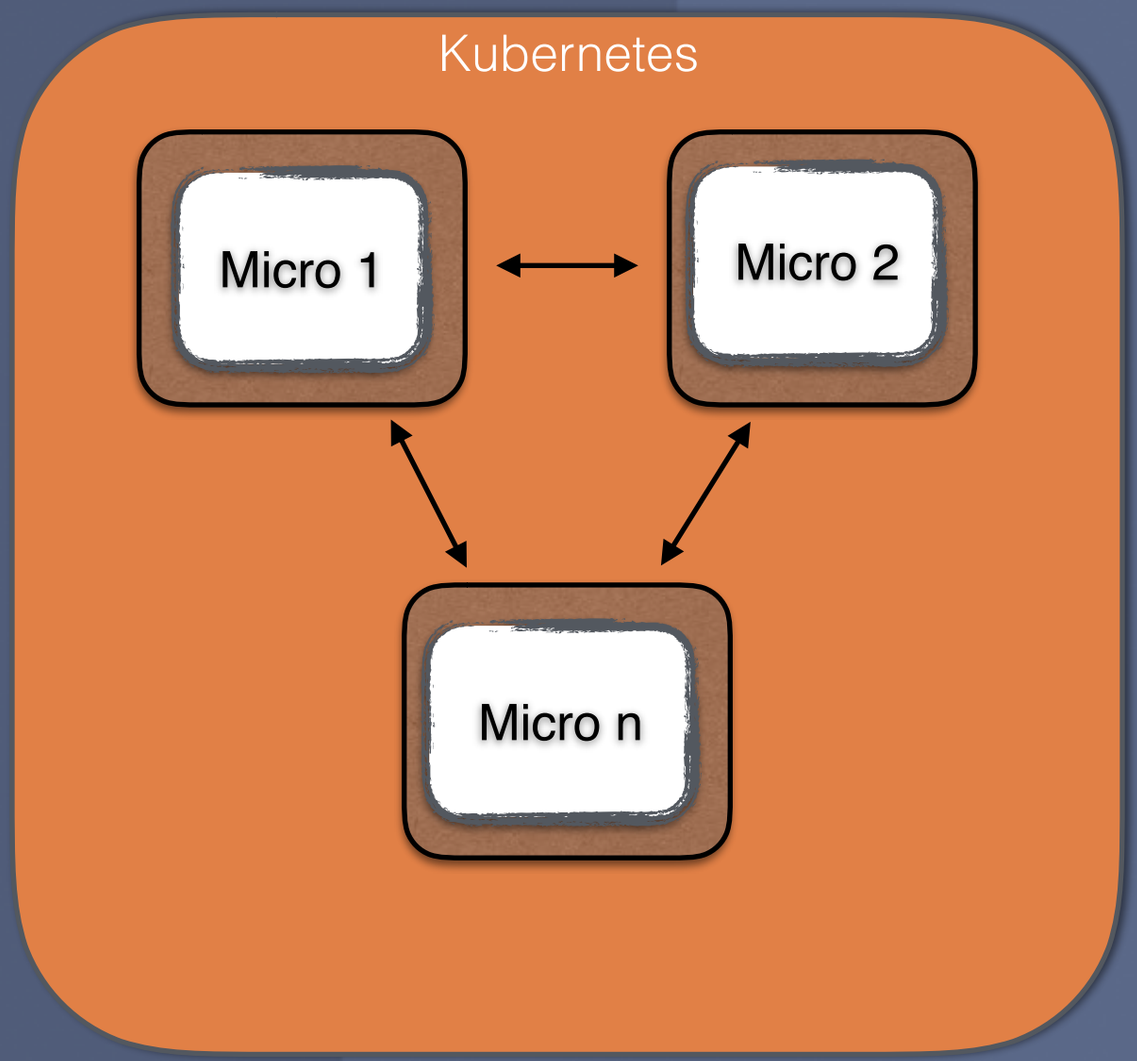
anynines

Transformation



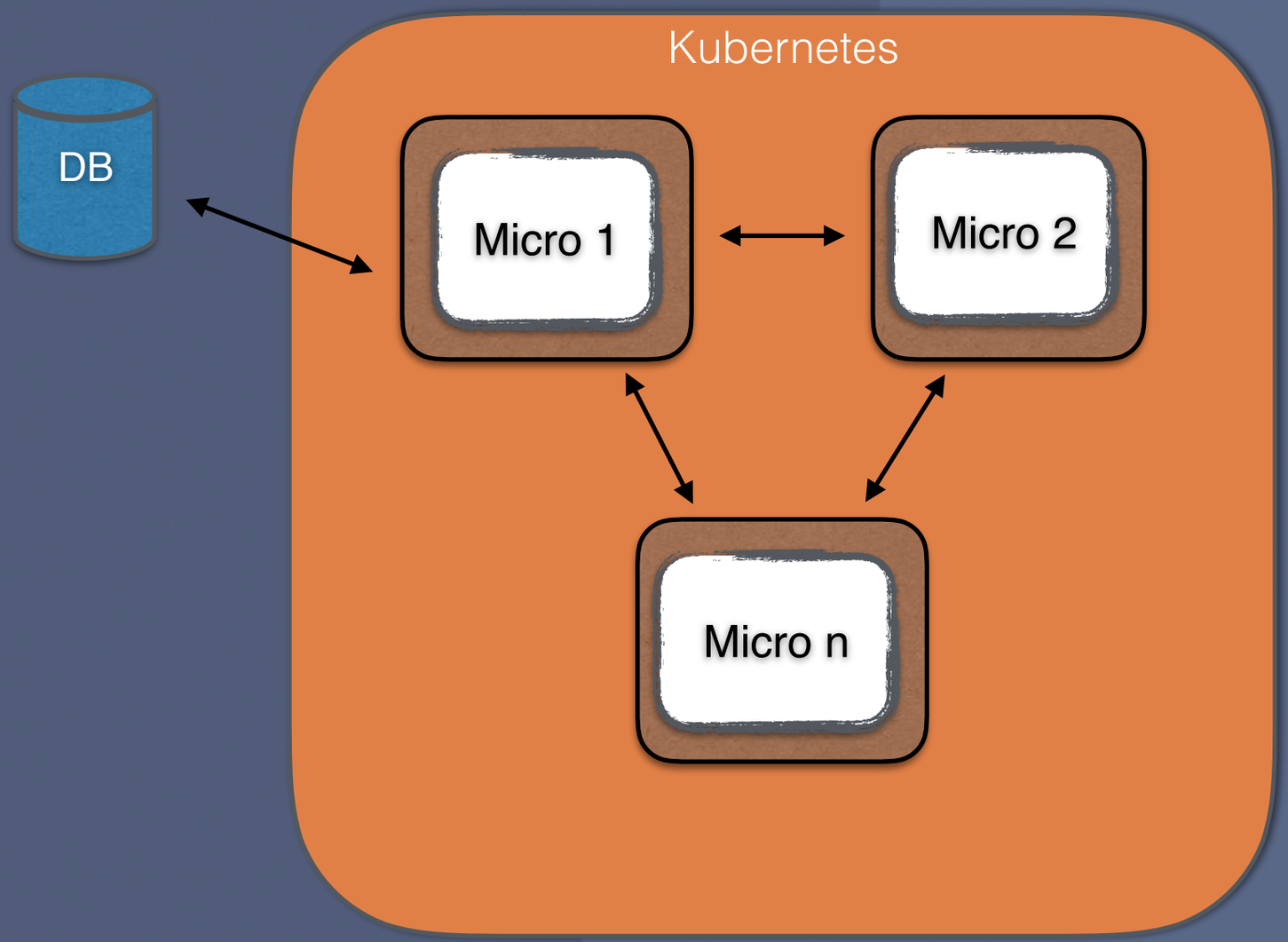
anynines

Transformation



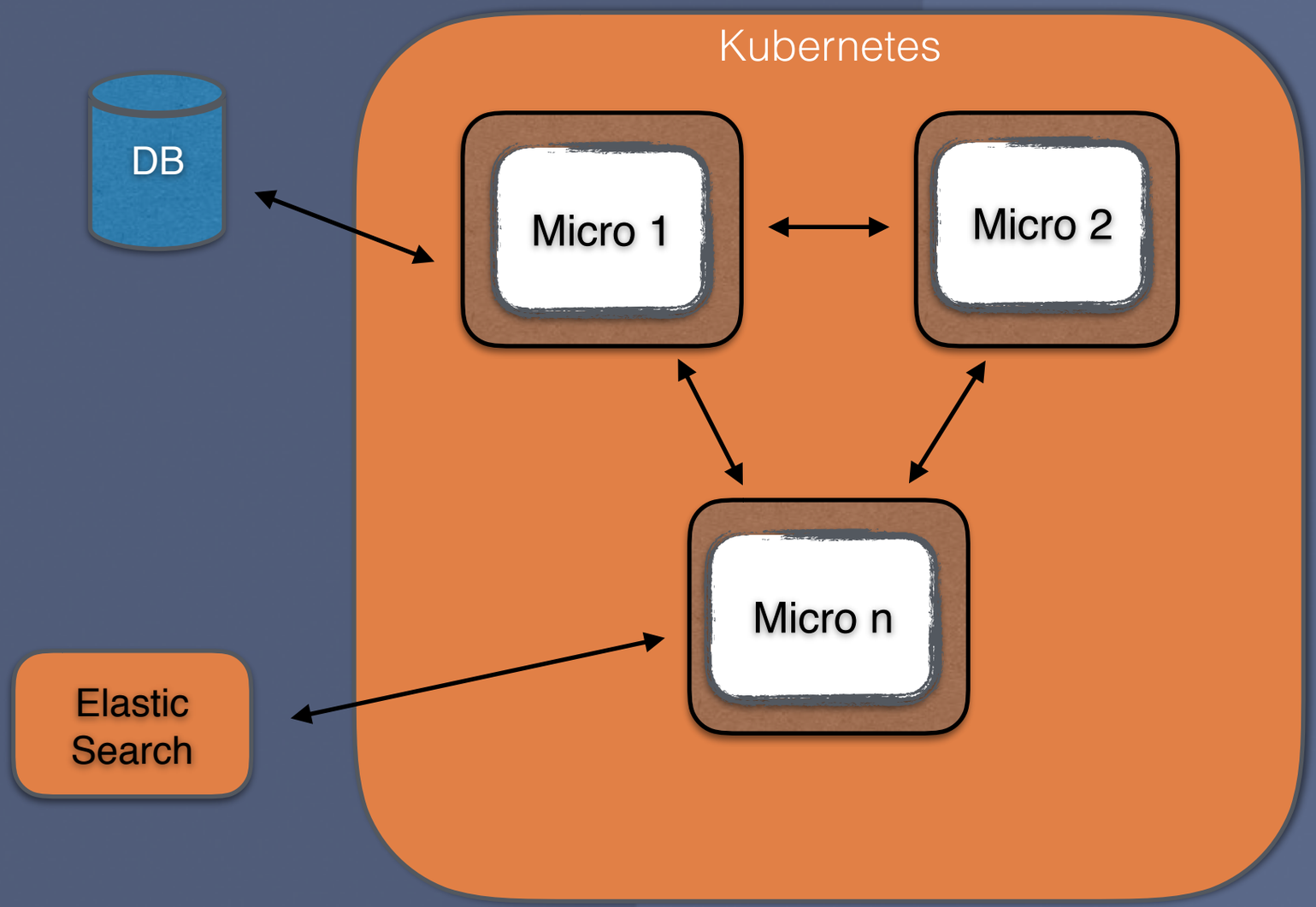
anynines

Transformation



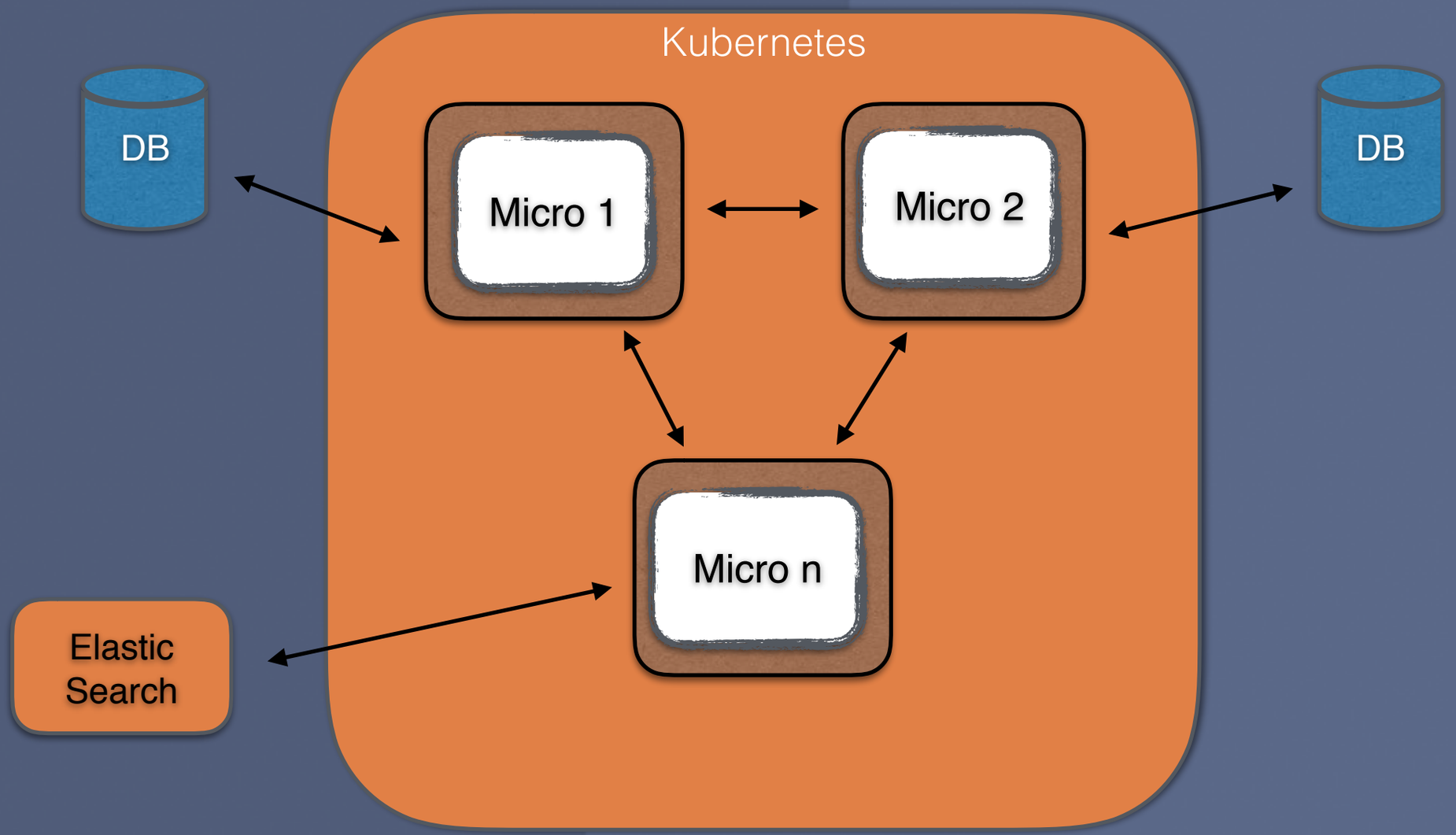
anynines

Transformation



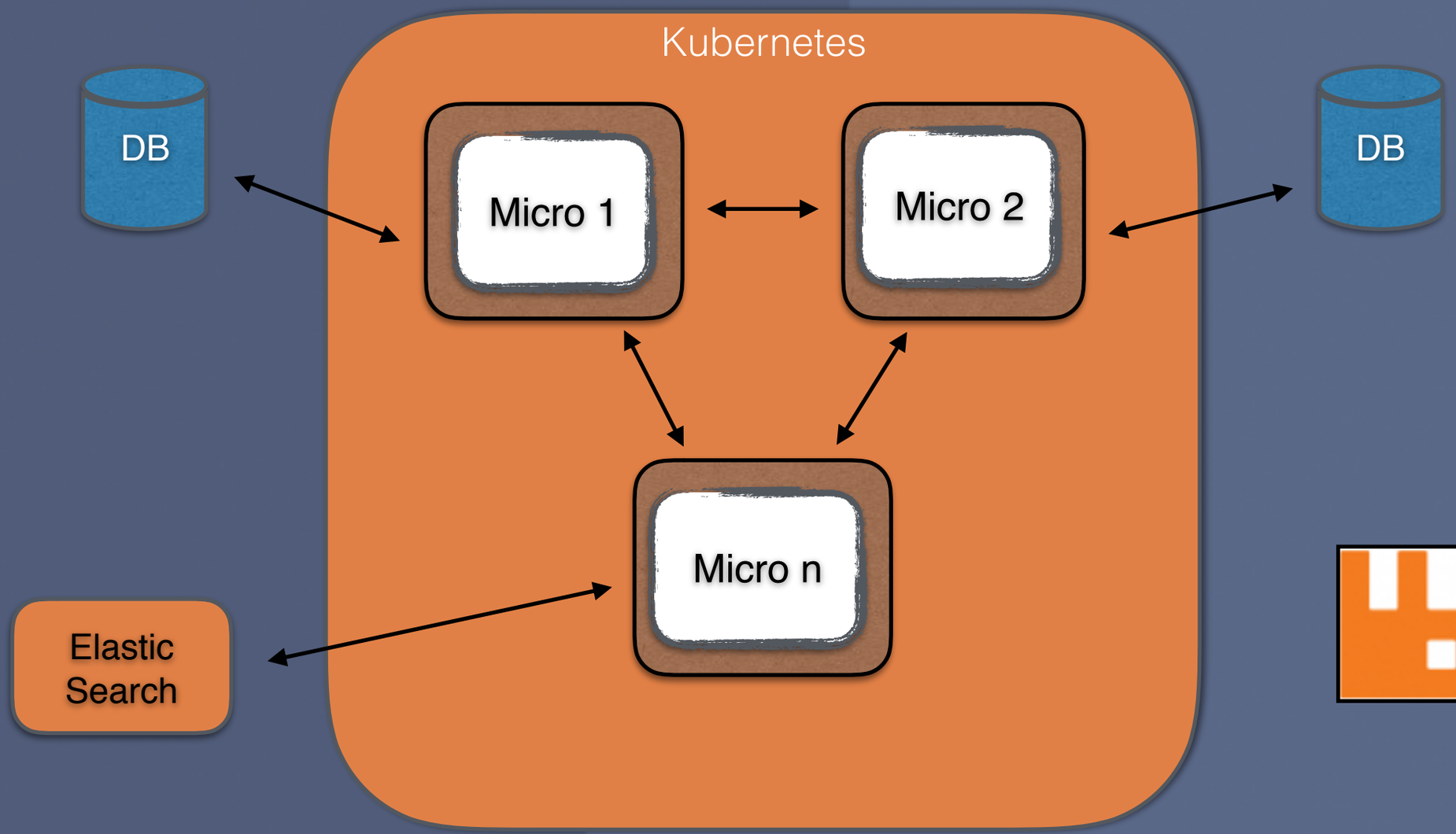
anynines

Transformation



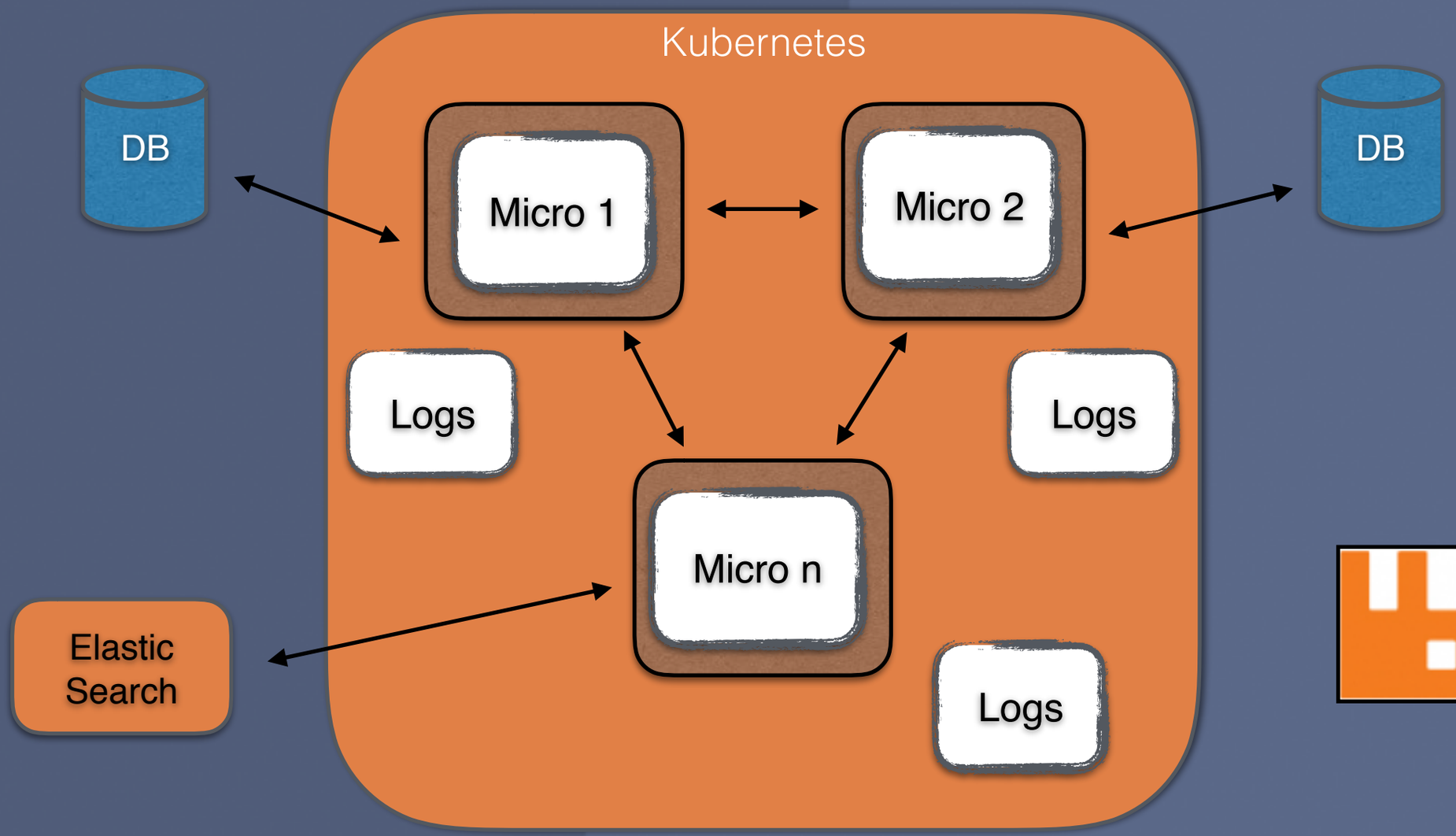
anynines

Transformation



anynines

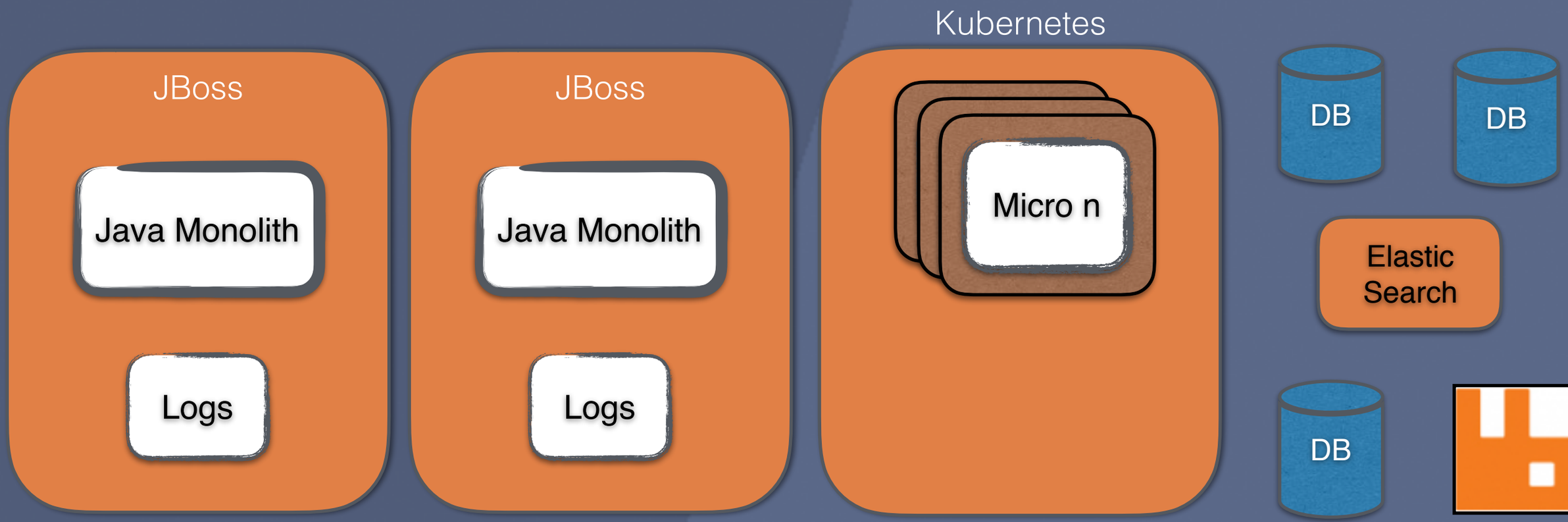
Transformation



anynines

Big Bang

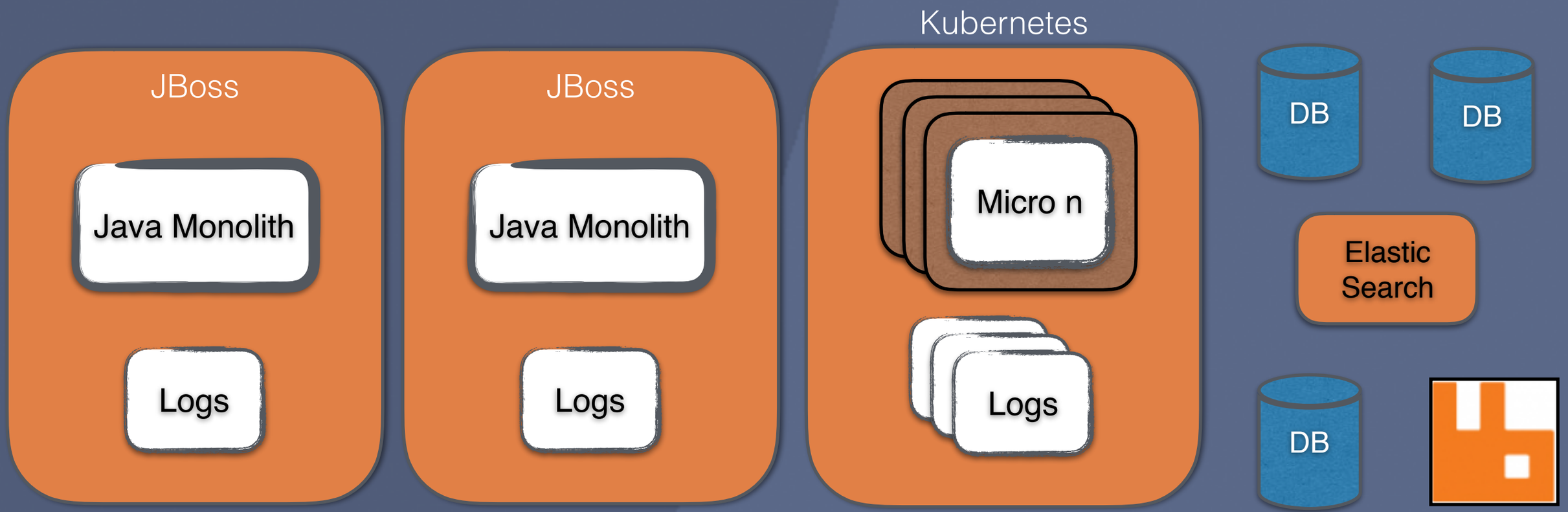
Bare Metal



anynines

Big Bang

Bare Metal



anynines

Big Bang

Bare Metal .....

anynines

Conway’s Law

“

”

anynines

Conway's Law

Organizations which design Systems … are constrained to produce designs which are copies of the communication structures of these organizations

anynines

Inverse Conway’s Law

“

”

anynines

Inverse Conway's Law

> *The organizational structure of a company is determined by the architecture of its product*

anynines

Big Bang

anynines

Big Bang

- Microservices Crashed

anynines

Big Bang

- Microservices Crashed
- Logs were hard to read (not centralized, no tracing)

anynines

Big Bang

- Microservices Crashed
- Logs were hard to read (not centralized, no tracing)
- Blaming on Devs

anynines

Big Bang

- Microservices Crashed
- Logs were hard to read (not centralized, no tracing)
- Blaming on Devs
- „You have to run it“, but there was no ops knowledge

anynines

Big Bang

- Microservices Crashed
- Logs were hard to read (not centralized, no tracing)
- Blaming on Devs
- „You have to run it“, but there was no ops knowledge
- „everything was better in the past“

anynines

Big Bang

- Microservices Crashed
- Logs were hard to read (not centralized, no tracing)
- Blaming on Devs
- „You have to run it“, but there was no ops knowledge
- „everything was better in the past“
- Bleading Edge Kubernetes

anynines

What went wrong?

anynines

What went wrong?

- Trust Issues

anynines

What went wrong?

- Trust Issues
- No Knowledge Sharing

anynines

What went wrong?

- Trust Issues
- No Knowledge Sharing
- Fear of change

anynines

What went wrong?

- Trust Issues
- No Knowledge Sharing
- Fear of change
- Blaming Culture

anynines

What went wrong?

- Trust Issues
- No Knowledge Sharing
- Fear of change
- Blaming Culture
- Introduction of too many things „at once“

anynines

What went wrong?

- Trust Issues
- No Knowledge Sharing
- Fear of change
- Blaming Culture
- Introduction of too many things „at once“
- Too less knowledge about DevOps practices

anynines

What went wrong?

- Trust Issues
- No Knowledge Sharing
- Fear of change
- Blaming Culture
- Introduction of too many things „at once“
- Too less knowledge about DevOps practices
- No transformation plan

anynines

What went wrong?

- Trust Issues
- No Knowledge Sharing
- Fear of change
- Blaming Culture
- Introduction of too many things „at once“
- Too less knowledge about DevOps practices
- No transformation plan
- Poor C level support

anynines

What went wrong?

- Trust Issues
- No Knowledge Sharing
- Fear of change
- Blaming Culture
- Introduction of too many things „at once“
- Too less knowledge about DevOps practices
- No transformation plan
- Poor C level support
- No Mindset Development

anynines

Thomas Bradford
@kode4food

Folge ich

People, Culture, Collaboration, Customers, Code, and Processes. In that order.

Tweet übersetzen

09:13 - 3. Sep. 2018

1 Retweet **4** „Gefällt mir“-Angaben

1 4

anynines

“

”

anynines

“Individuals and interactions over processes and tools”

anynines

Chaos prevention

anynines

Chaos prevention

- Understand Conway's Law

anynines

Chaos prevention

- Understand Conway's Law
- Throw Blaming Culture away

anynines

Chaos prevention

- Understand Conway's Law
- Throw Blaming Culture away
- Gain and share knowledge about technologies and DevOps practices

anynines

Chaos prevention

- Understand Conway's Law
- Throw Blaming Culture away
- Gain and share knowledge about technologies and DevOps practices
- Develop a DevOps mindset/culture

anynines

Chaos prevention

- Understand Conway's Law
- Throw Blaming Culture away
- Gain and share knowledge about technologies and DevOps practices
- Develop a DevOps mindset/culture
- Make a plan

anynines

Chaos prevention

- Understand Conway's Law
- Throw Blaming Culture away
- Gain and share knowledge about technologies and DevOps practices
- Develop a DevOps mindset/culture
- Make a plan
- Start with baby steps

anynines

Chaos prevention

- Understand Conway's Law
- Throw Blaming Culture away
- Gain and share knowledge about technologies and DevOps practices
- Develop a DevOps mindset/culture
- Make a plan
- Start with baby steps
- Ensure C Level support

anynines

“

”

anynines

“Give a motivated team the environment and support they need, and trust them to get their job done”

anynines

Success Story

anynines

Success Story - Initial Situation

Success Story - Initial Situation

- Big Customer

anynines

Success Story - Initial Situation

- Big Customer
- Ongoing Digital Transformation

anynines

Success Story - Initial Situation

- Big Customer
- Ongoing Digital Transformation
- Cloud Platform (Cloud Foundry) in place

anynines

Success Story - Initial Situation

- Big Customer
- Ongoing Digital Transformation
- Cloud Platform (Cloud Foundry) in place
- Mission was to develop a piece of Software that uses CF

anynines

Success Story - Initial Situation

- Big Customer
- Ongoing Digital Transformation
- Cloud Platform (Cloud Foundry) in place
- Mission was to develop a piece of Software that uses CF
- No access to infrastructure

anynines

Success Story - Requirements/Constraints

Success Story - Requirements/Constraints

- Cost Efficient

Success Story - Requirements/Constraints

- Cost Efficient
- Resilient

anynines

Success Story - Requirements/Constraints

- Cost Efficient
- Resilient
- a9s On Call for Incidents

anynines

Success Story - Requirements/Constraints

- Cost Efficient
- Resilient
- a9s On Call for Incidents
- AWS native

anynines

Success Story - Requirements/Constraints

- Cost Efficient
- Resilient
- a9s On Call for Incidents
- AWS native
- Serverless technologies

anynines

Success Story - Requirements/Constraints

- Cost Efficient
- Resilient
- a9s On Call for Incidents
- AWS native
- Serverless technologies
- Scrum

anynines

Challenges

anynines

Process Divergence

anynines

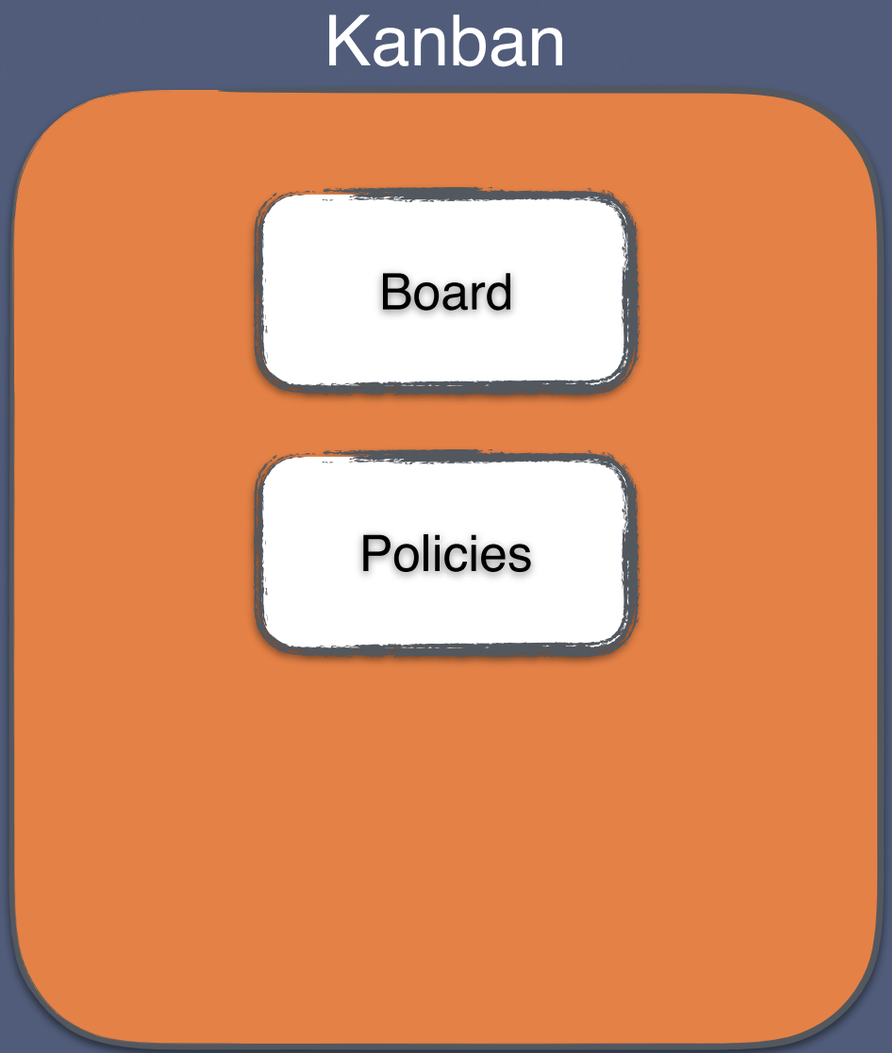
Process Divergence

Kanban

Process Divergence

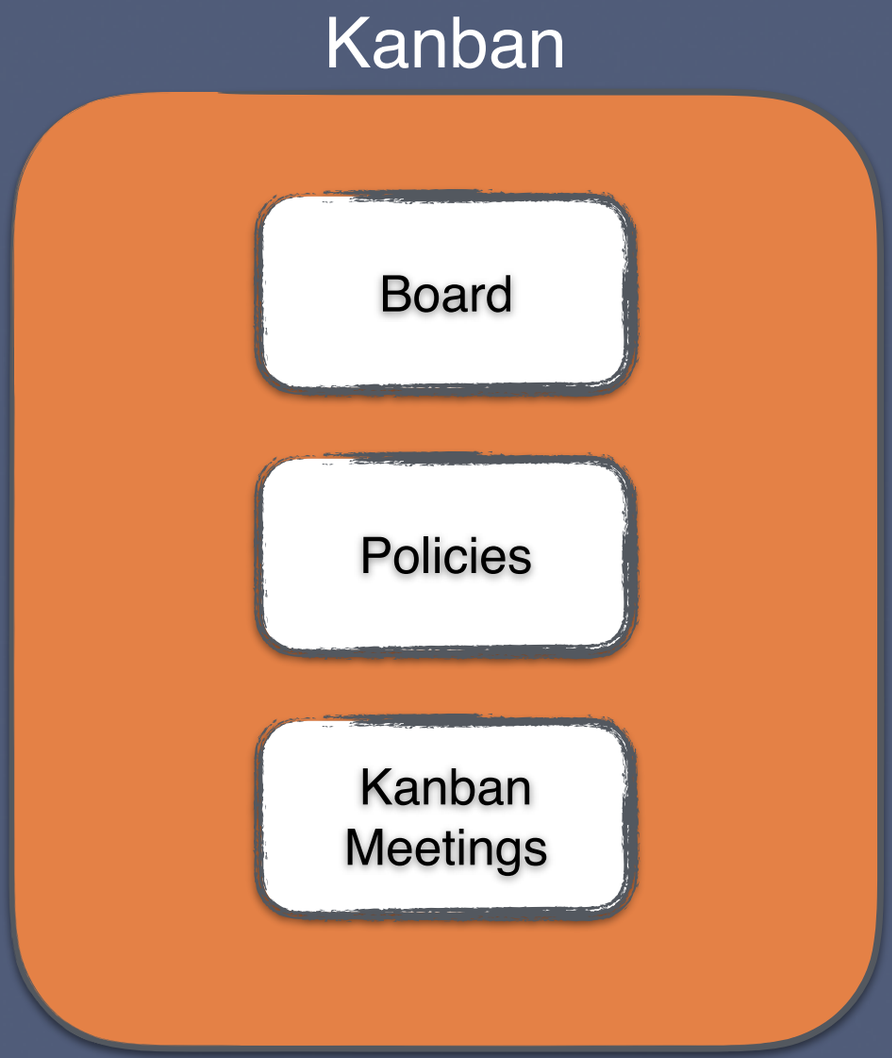
anynines

Process Divergence



anynines

Process Divergence



anynines

Process Divergence

Kanban

Board

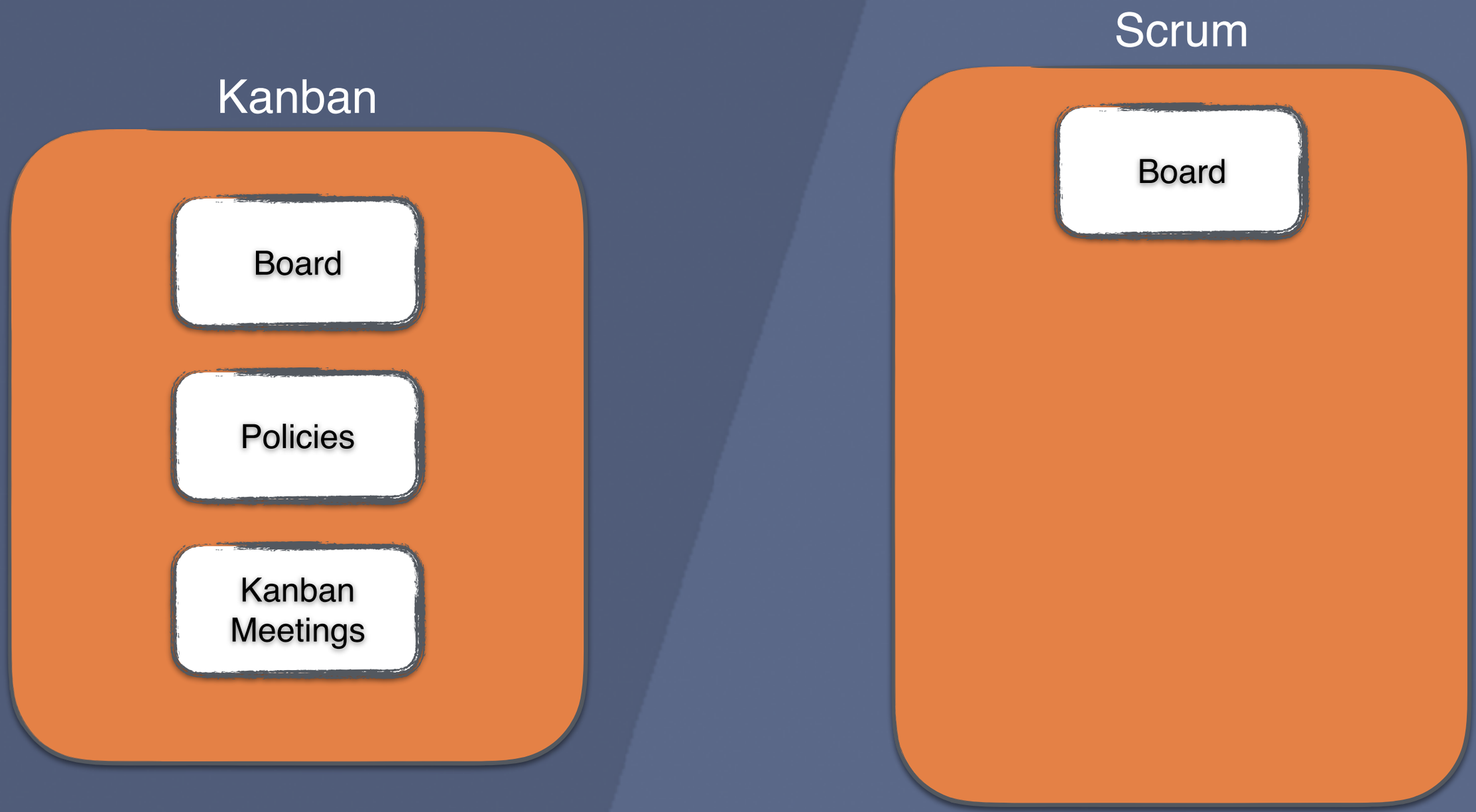
Policies

Kanban Meetings

Scrum

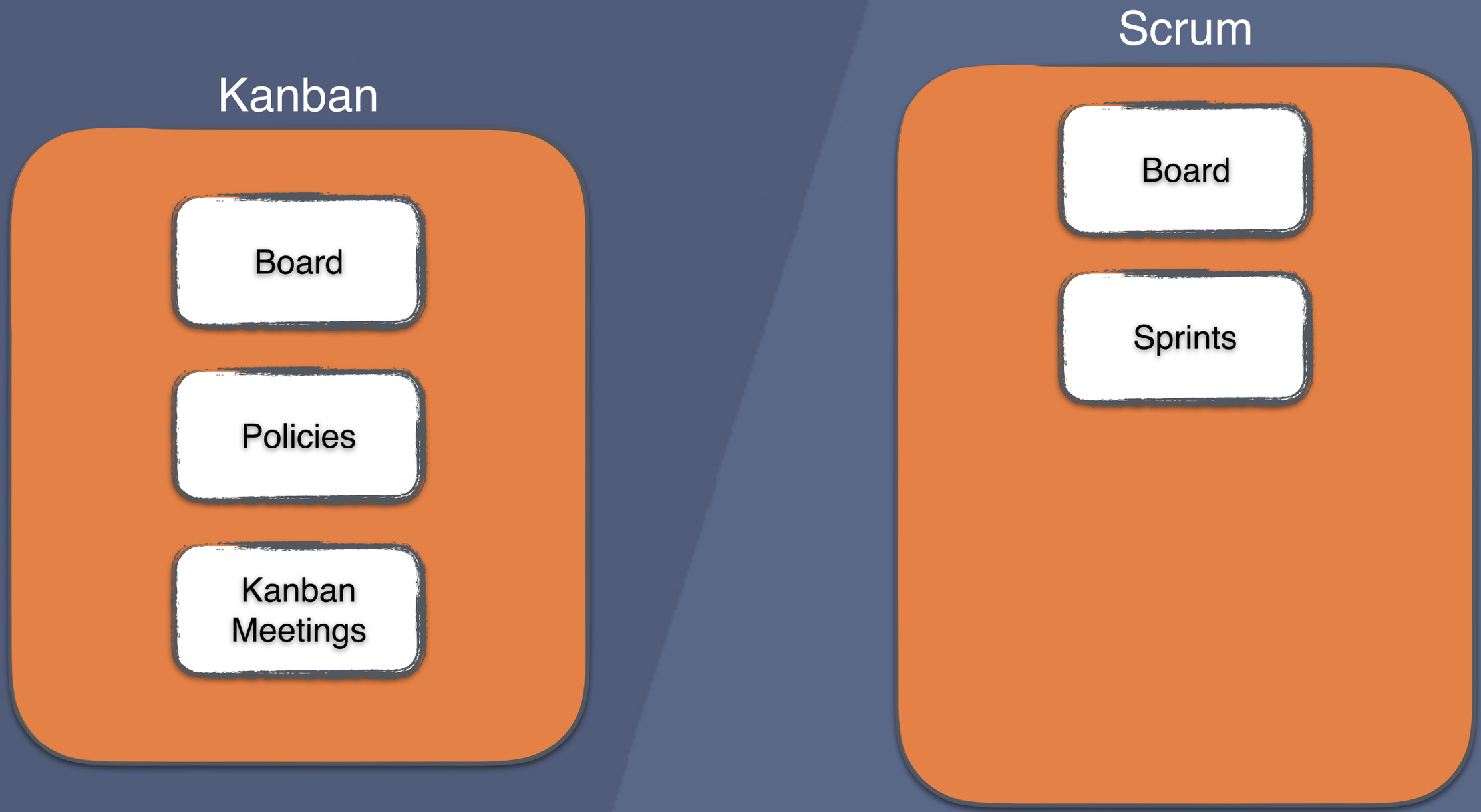
anynines

Process Divergence



anynines

Process Divergence



anynines

Process Divergence

Kanban

Board

Policies

Kanban Meetings

Scrum

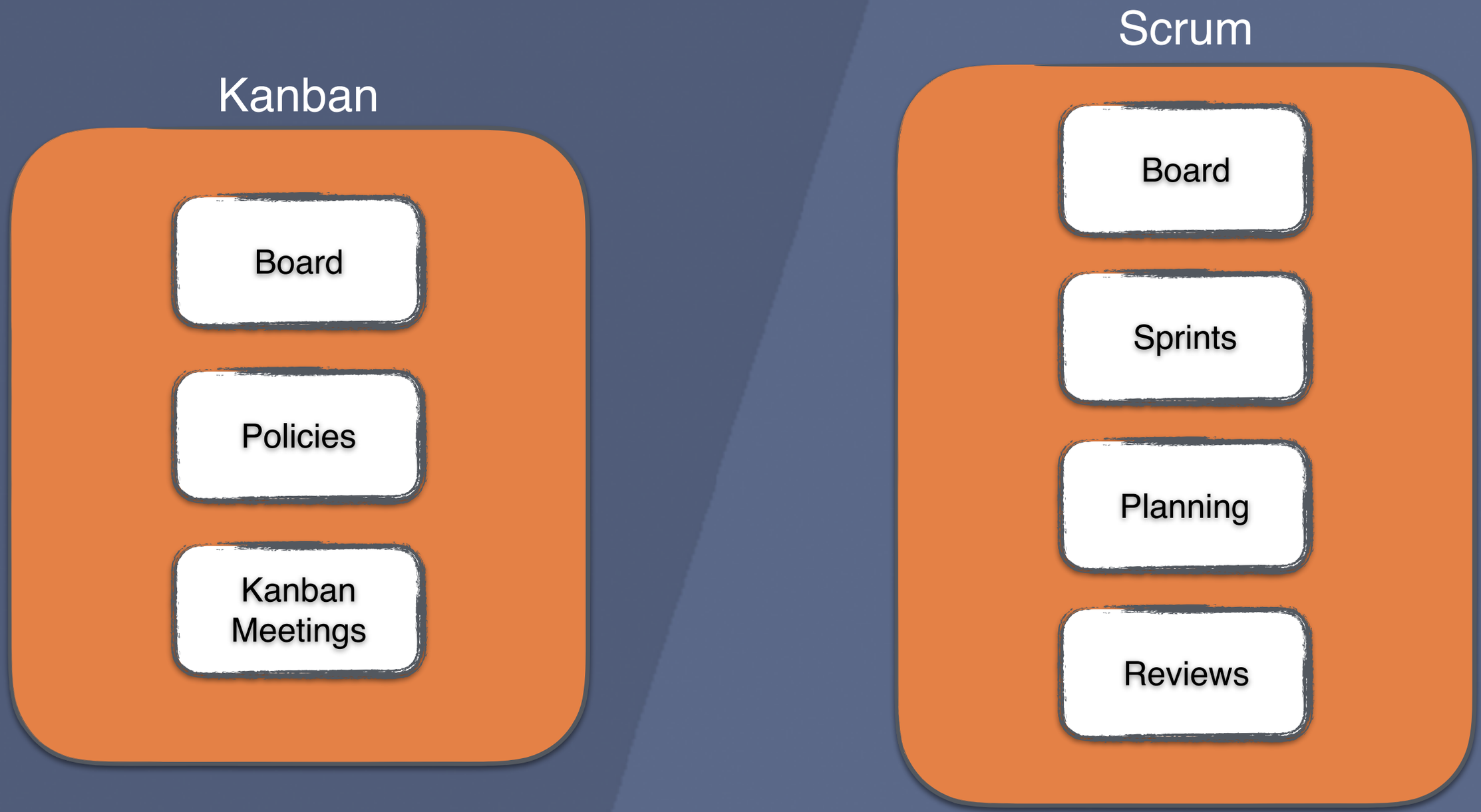
Board

Sprints

Planning

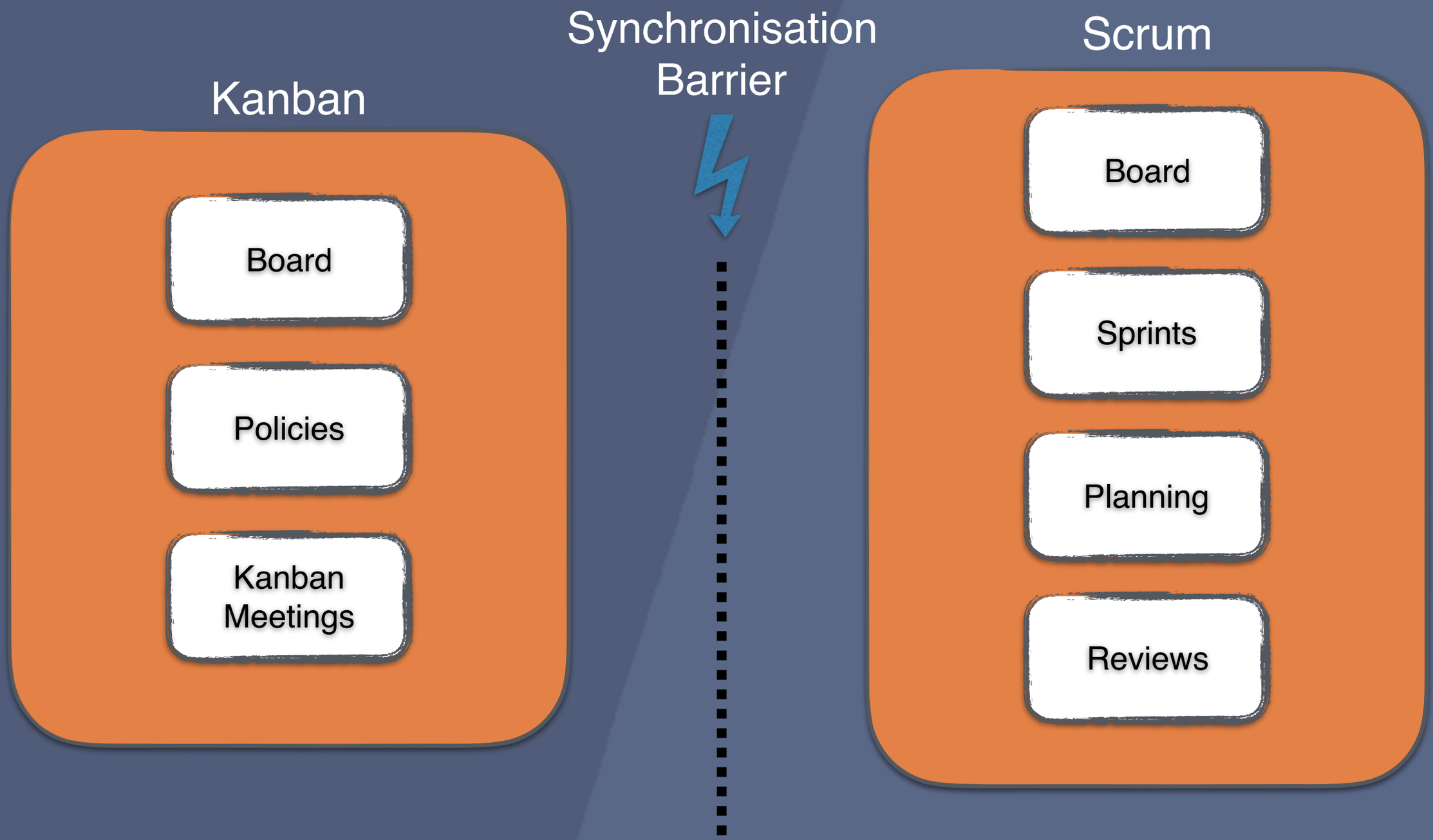
anynines

Process Divergence



anynines

Process Divergence



anynines

Context/Security Divergence

anynines

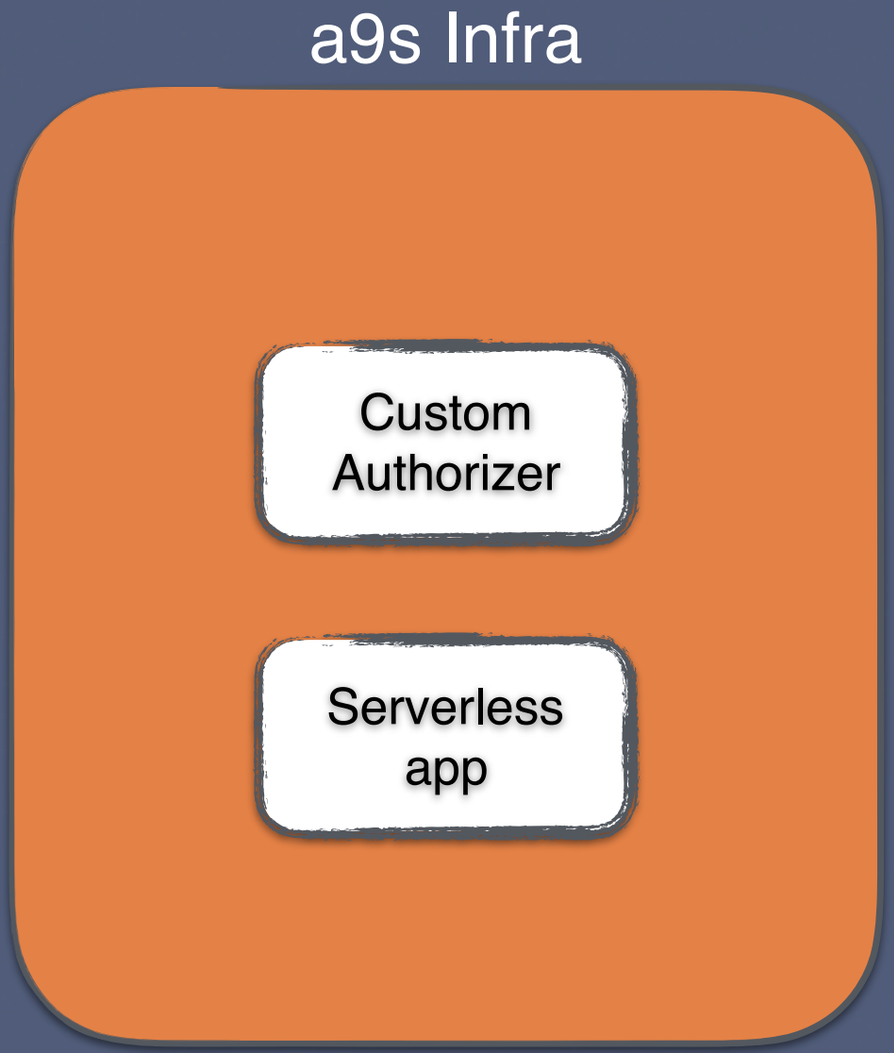
Context/Security Divergence

anynines

Context/Security Divergence

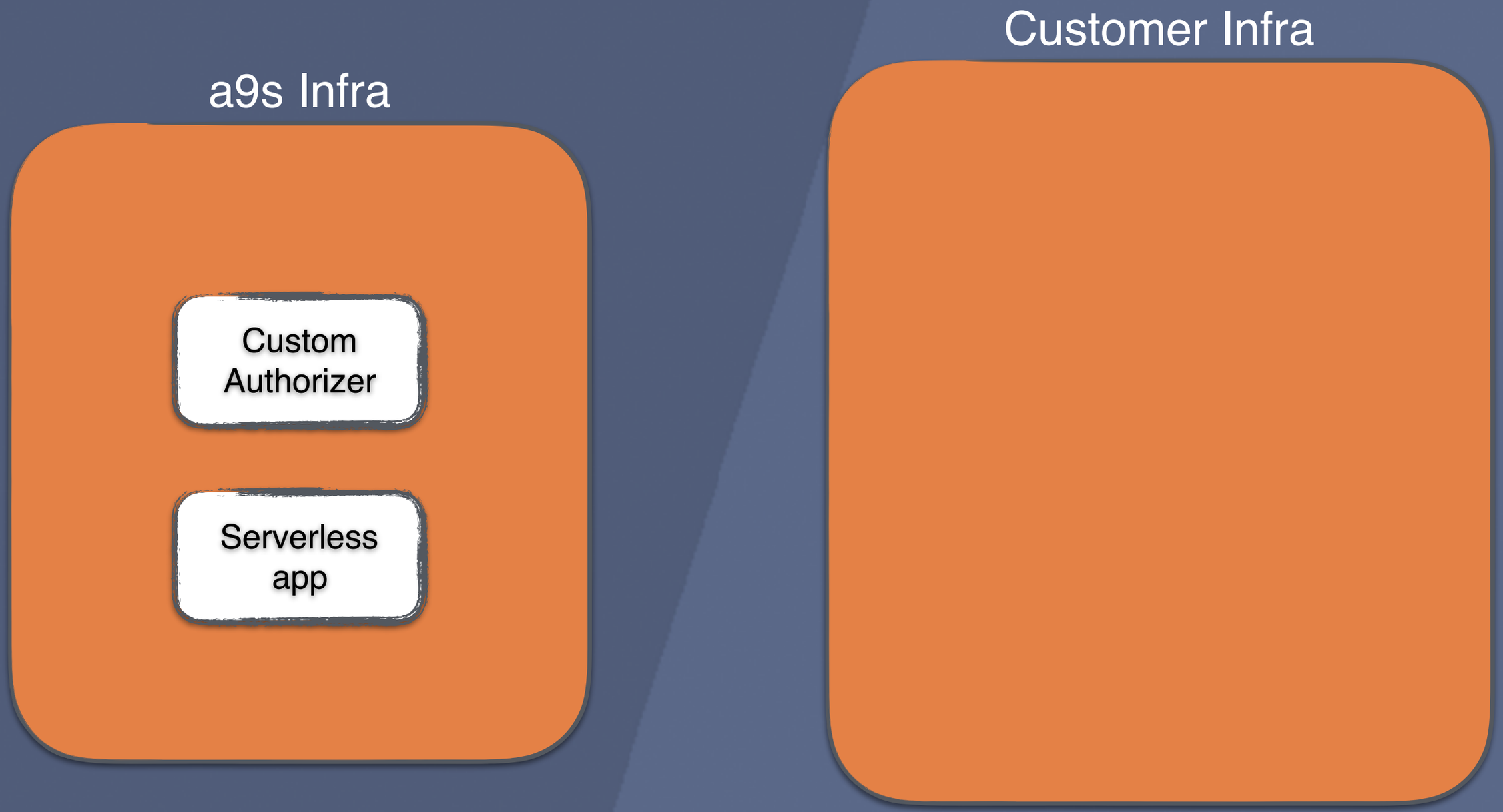
anynines

Context/Security Divergence



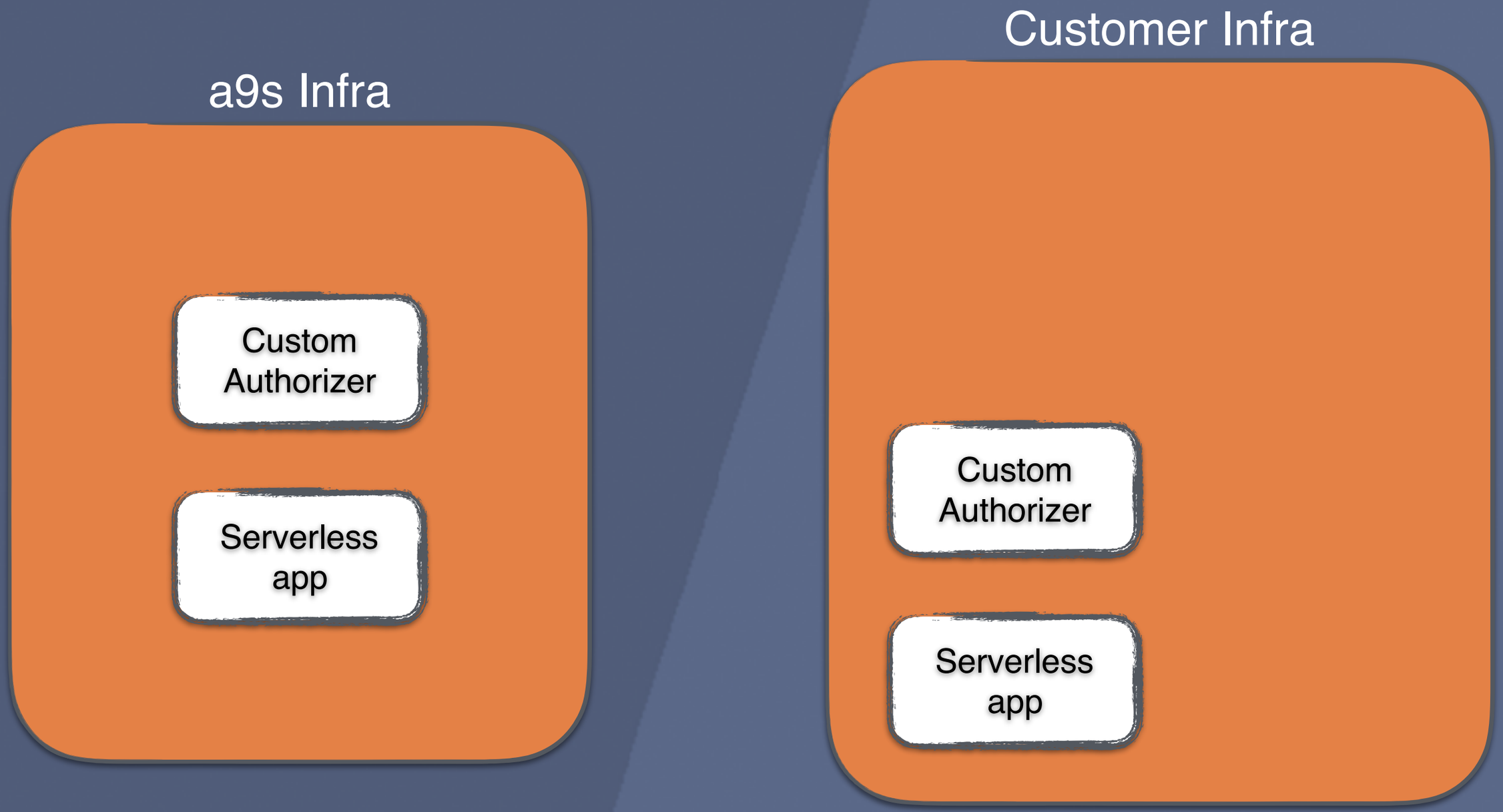
anynines

Context/Security Divergence



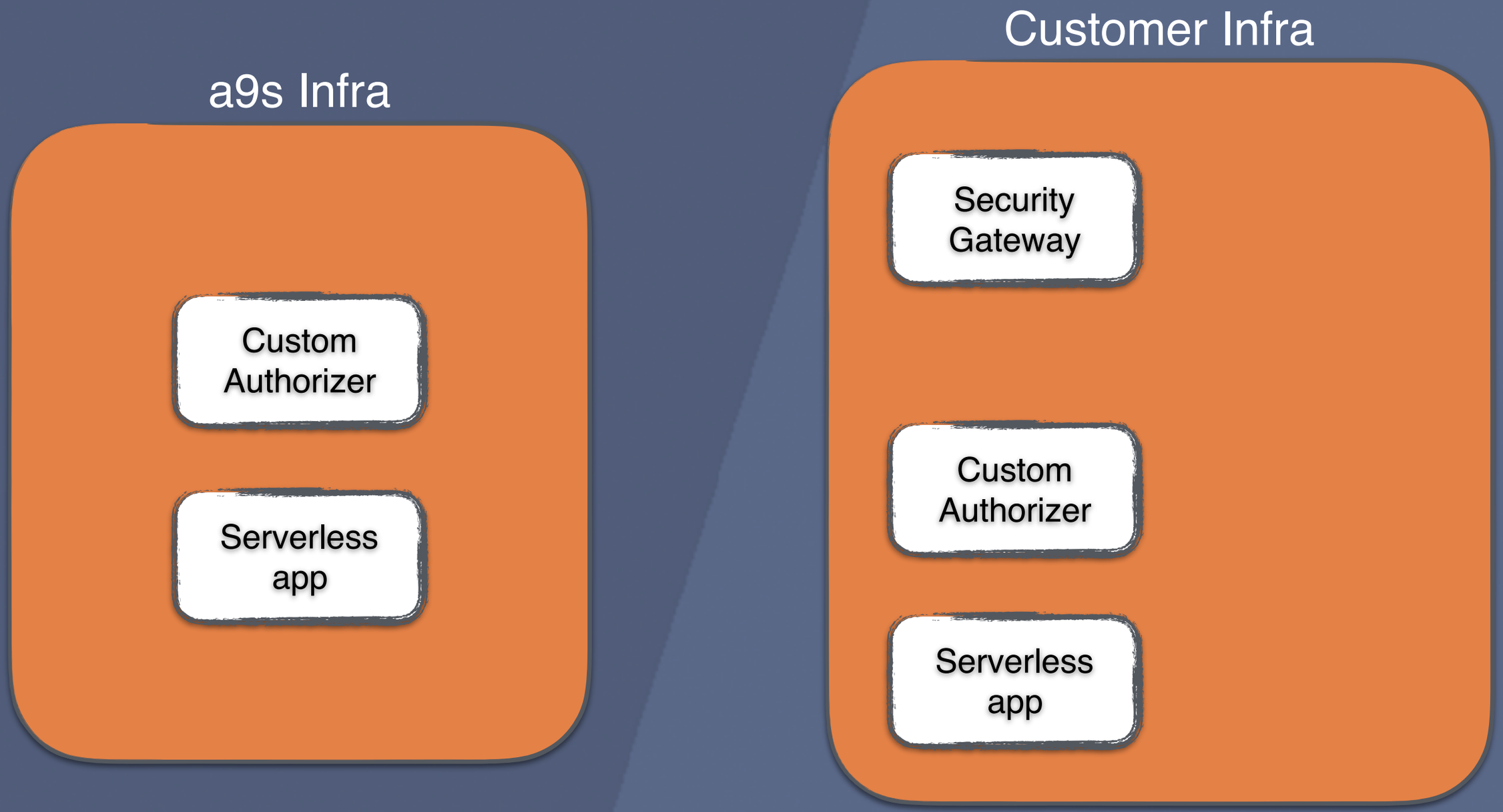
anynines

Context/Security Divergence



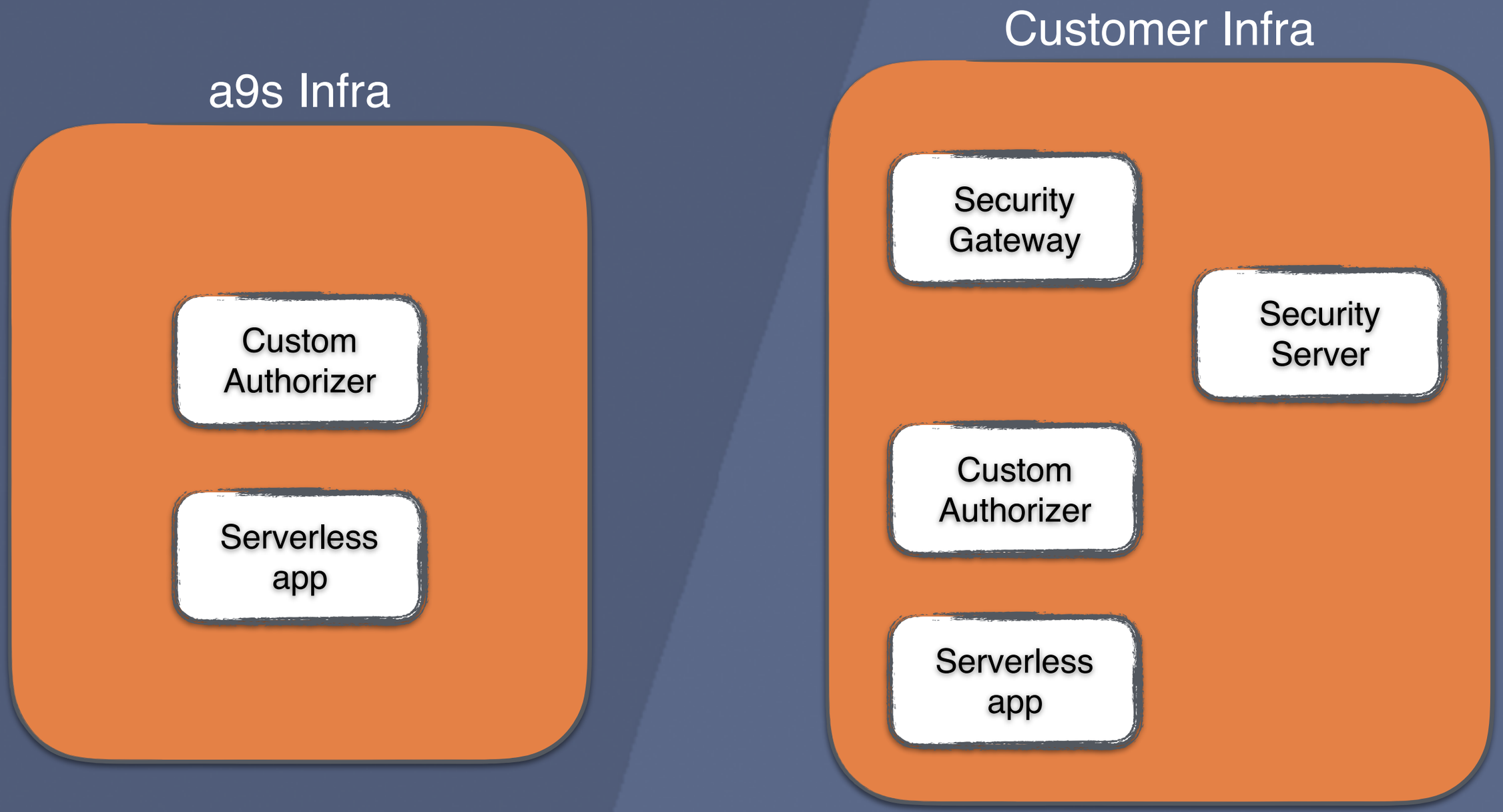
anynines

Context/Security Divergence



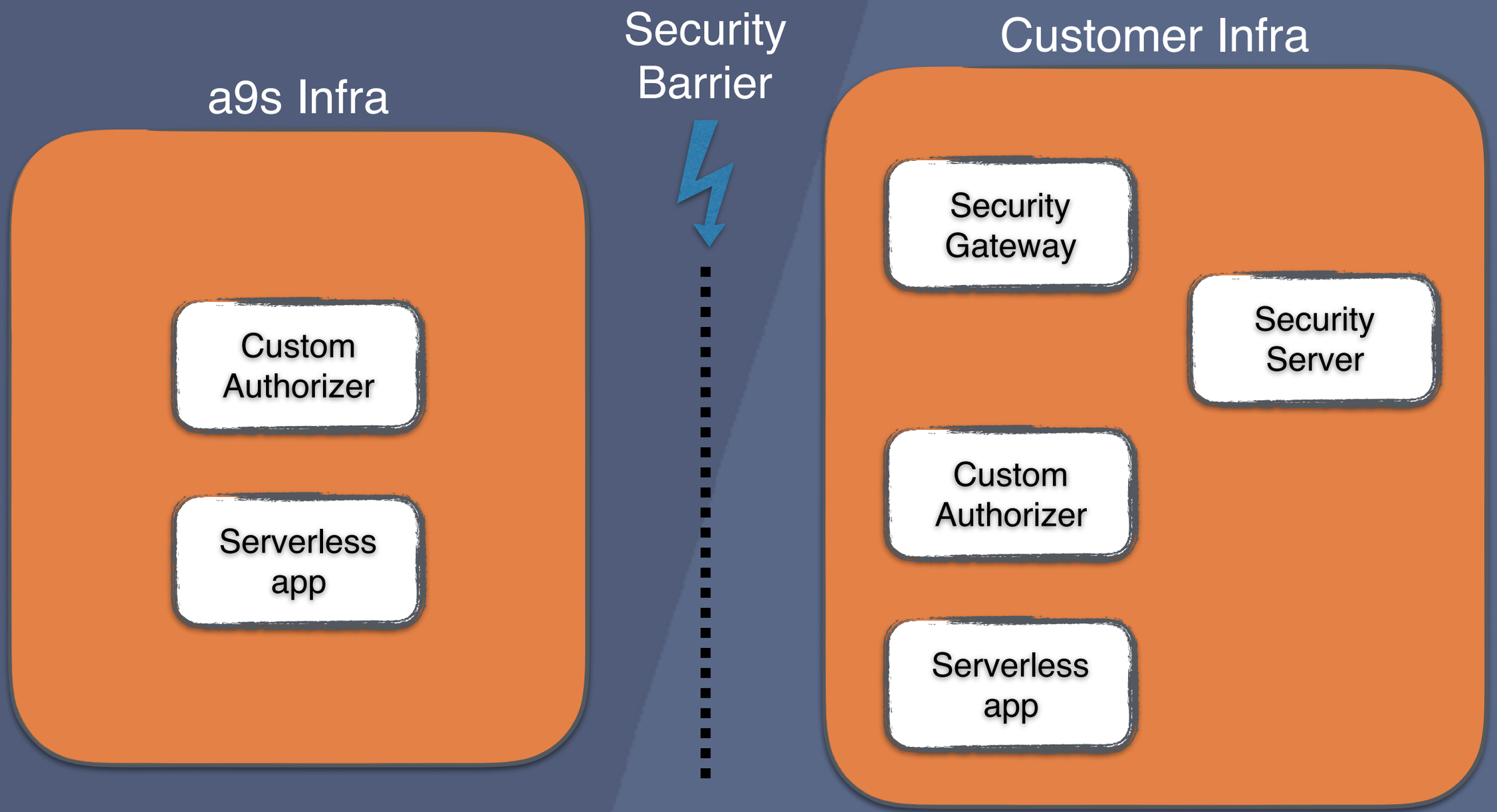
anynines

Context/Security Divergence



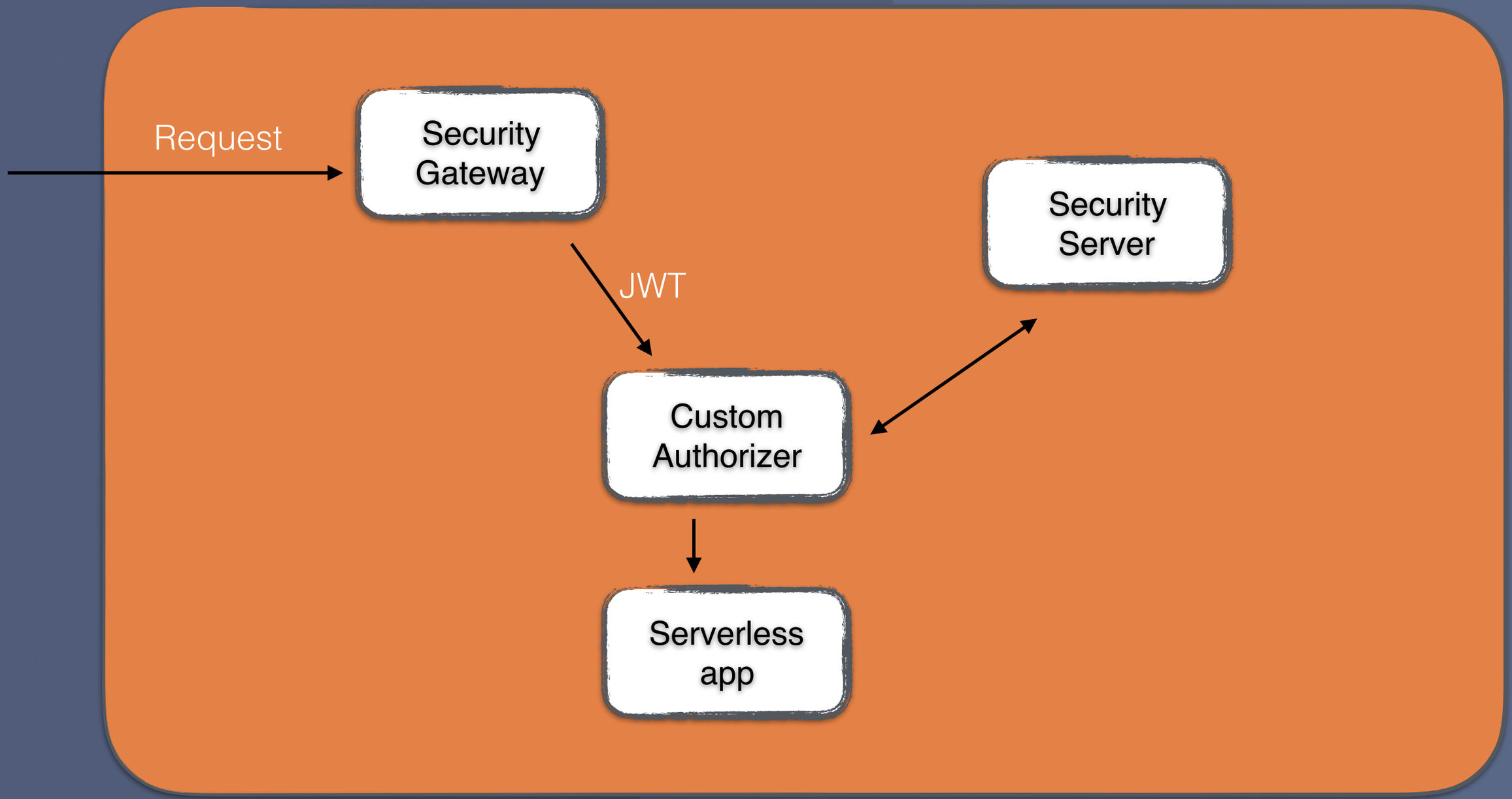
anynines

Context/Security Divergence



anynines

Context/Security Divergence



anynines

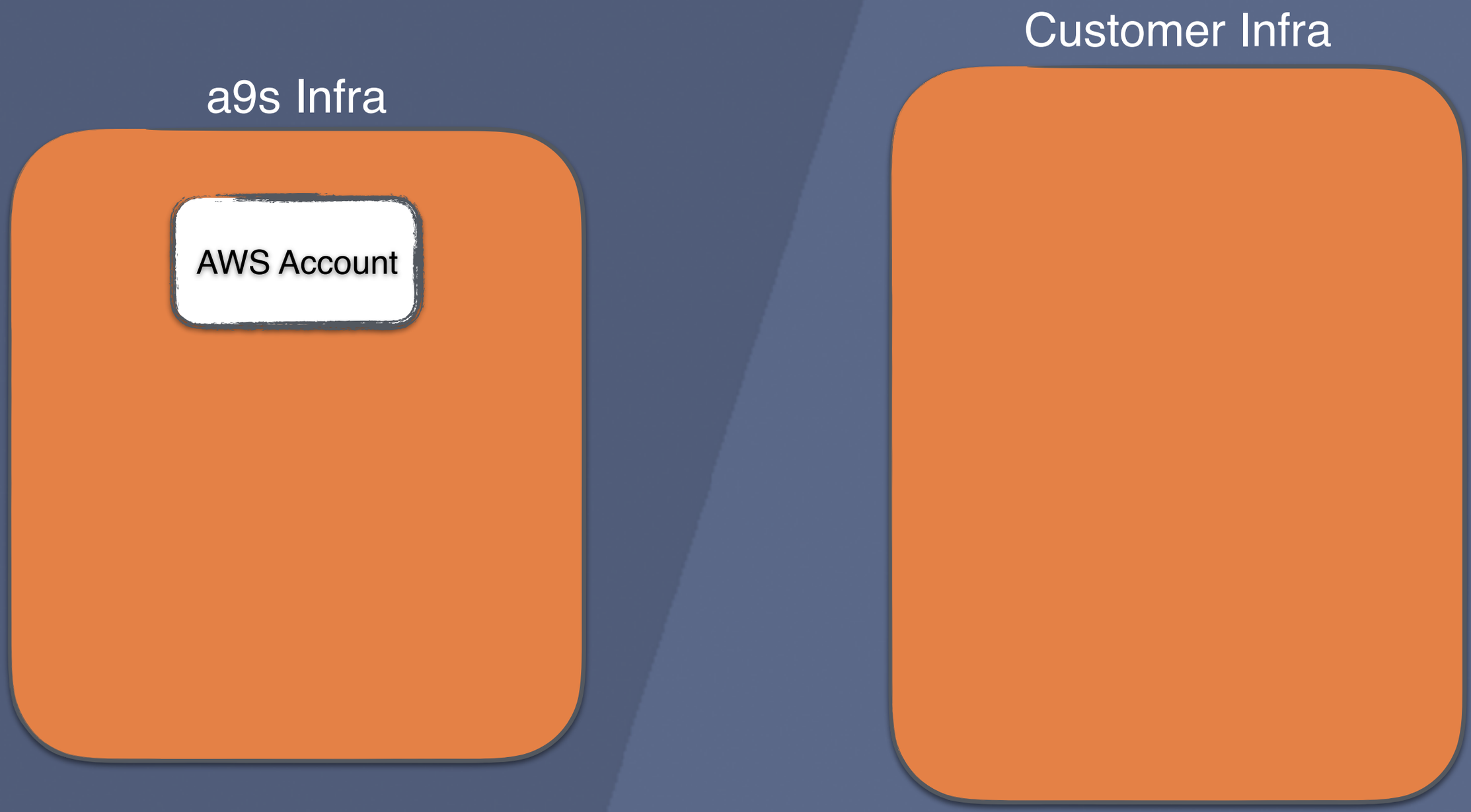
Infrastructure Divergence

a9s Infra

Customer Infra

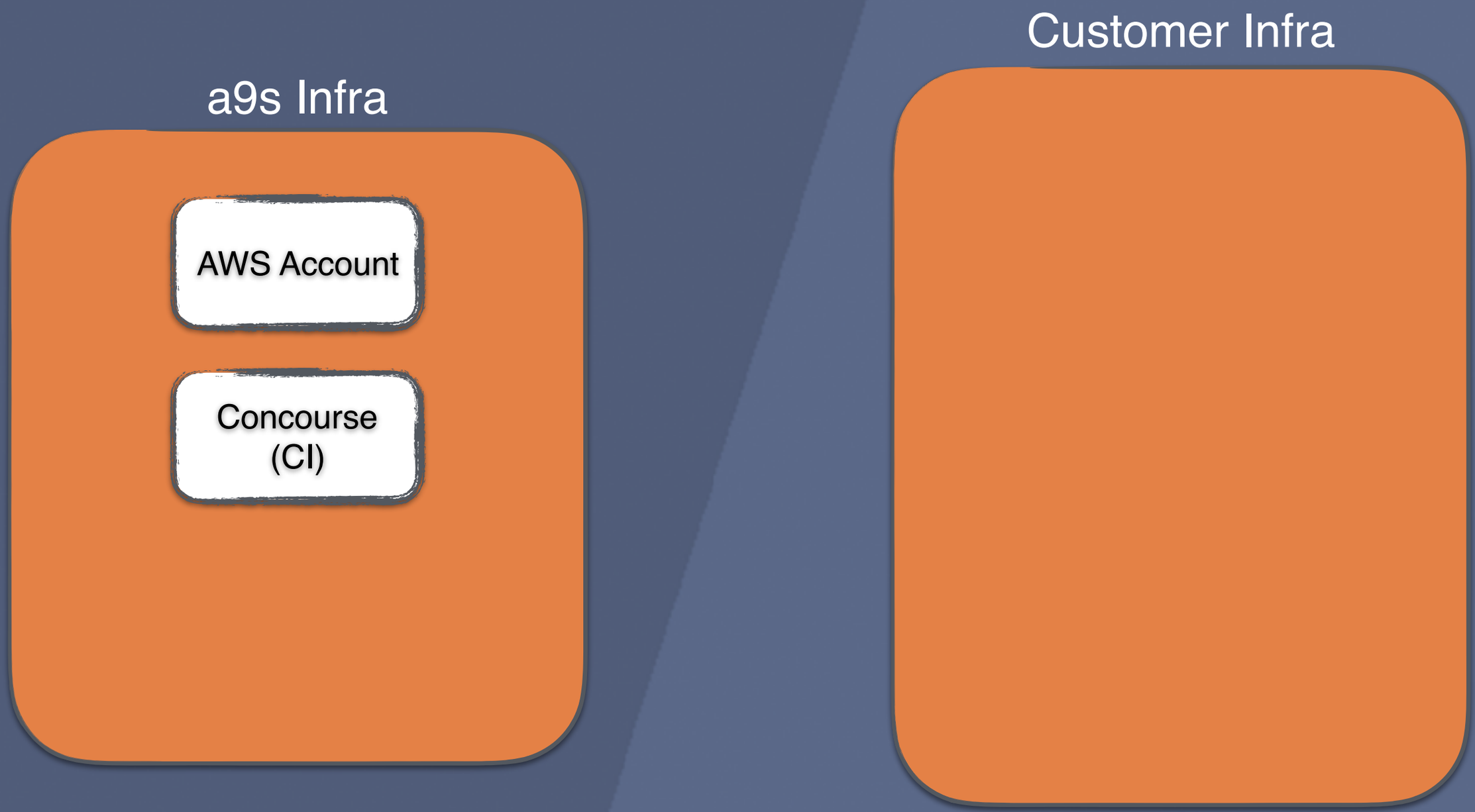
anynines

Infrastructure Divergence



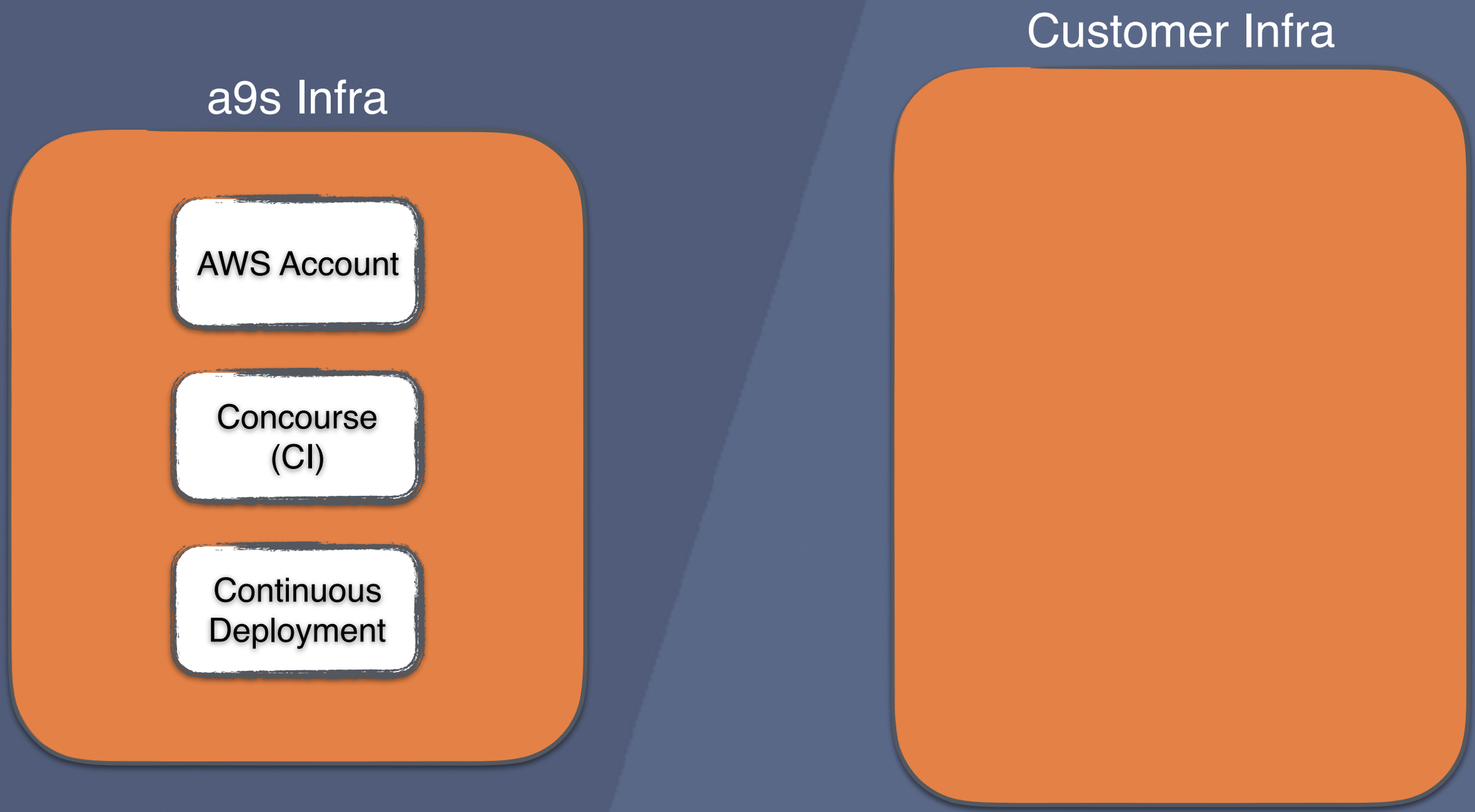
anynines

Infrastructure Divergence



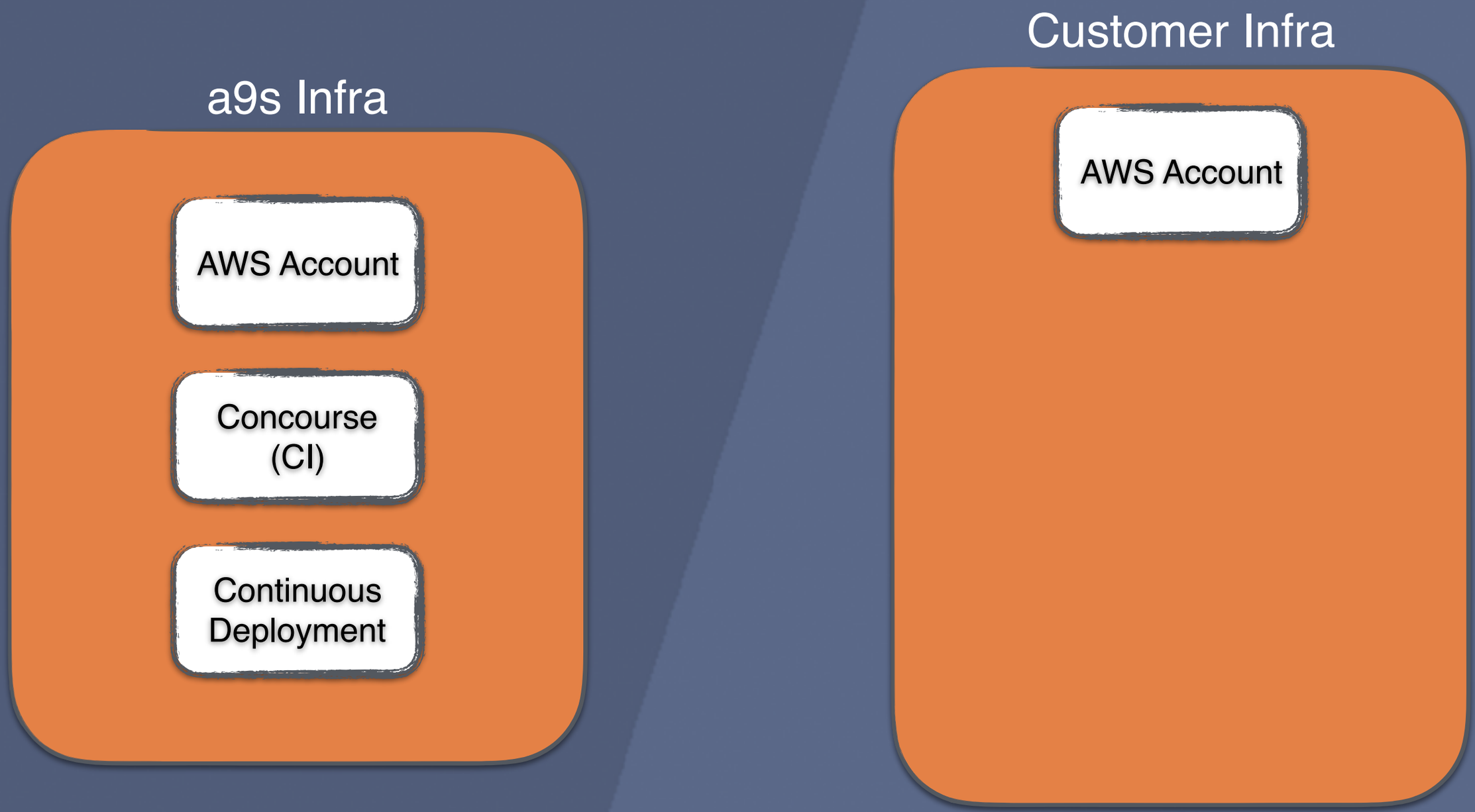
anynines

Infrastructure Divergence



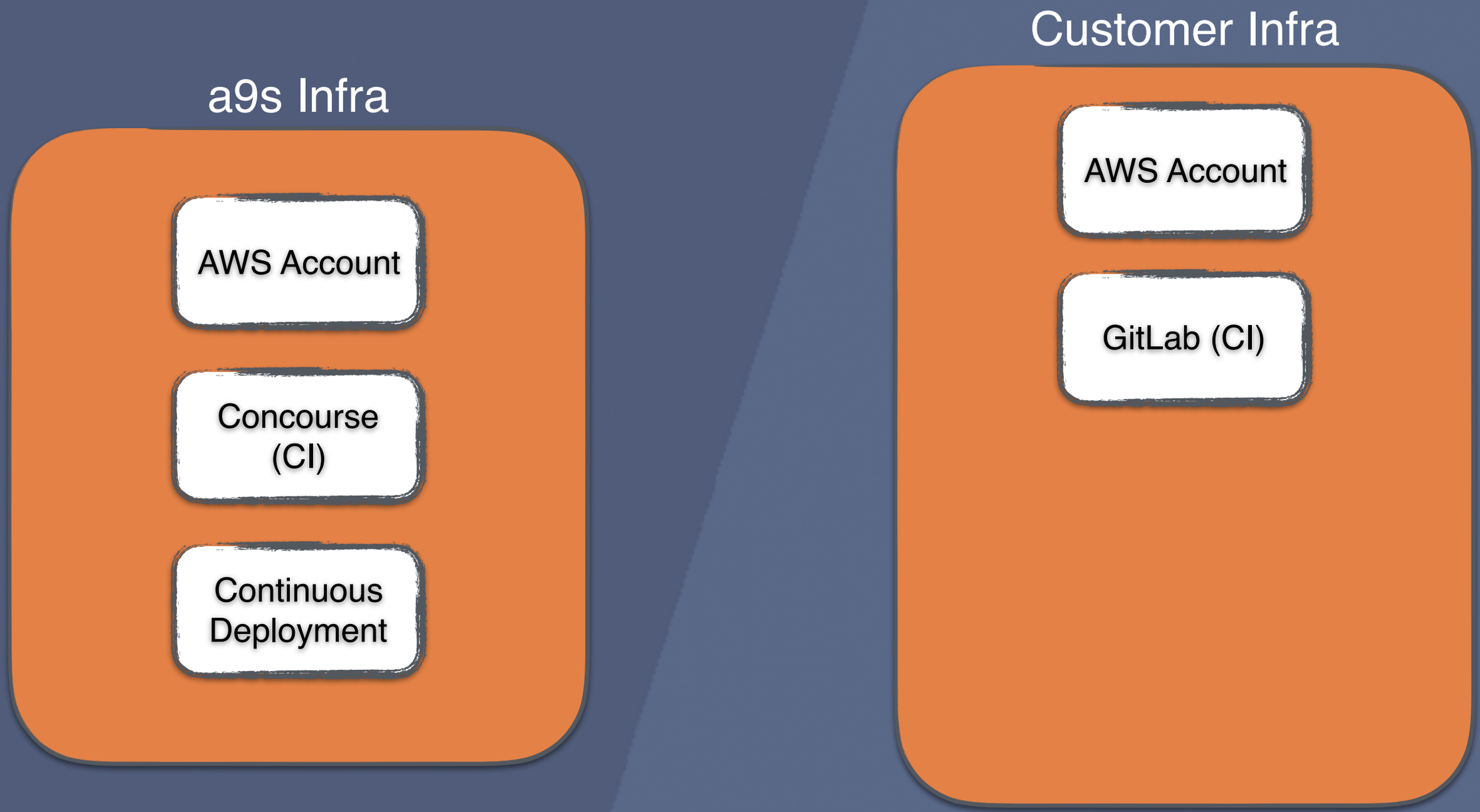
anynines

Infrastructure Divergence



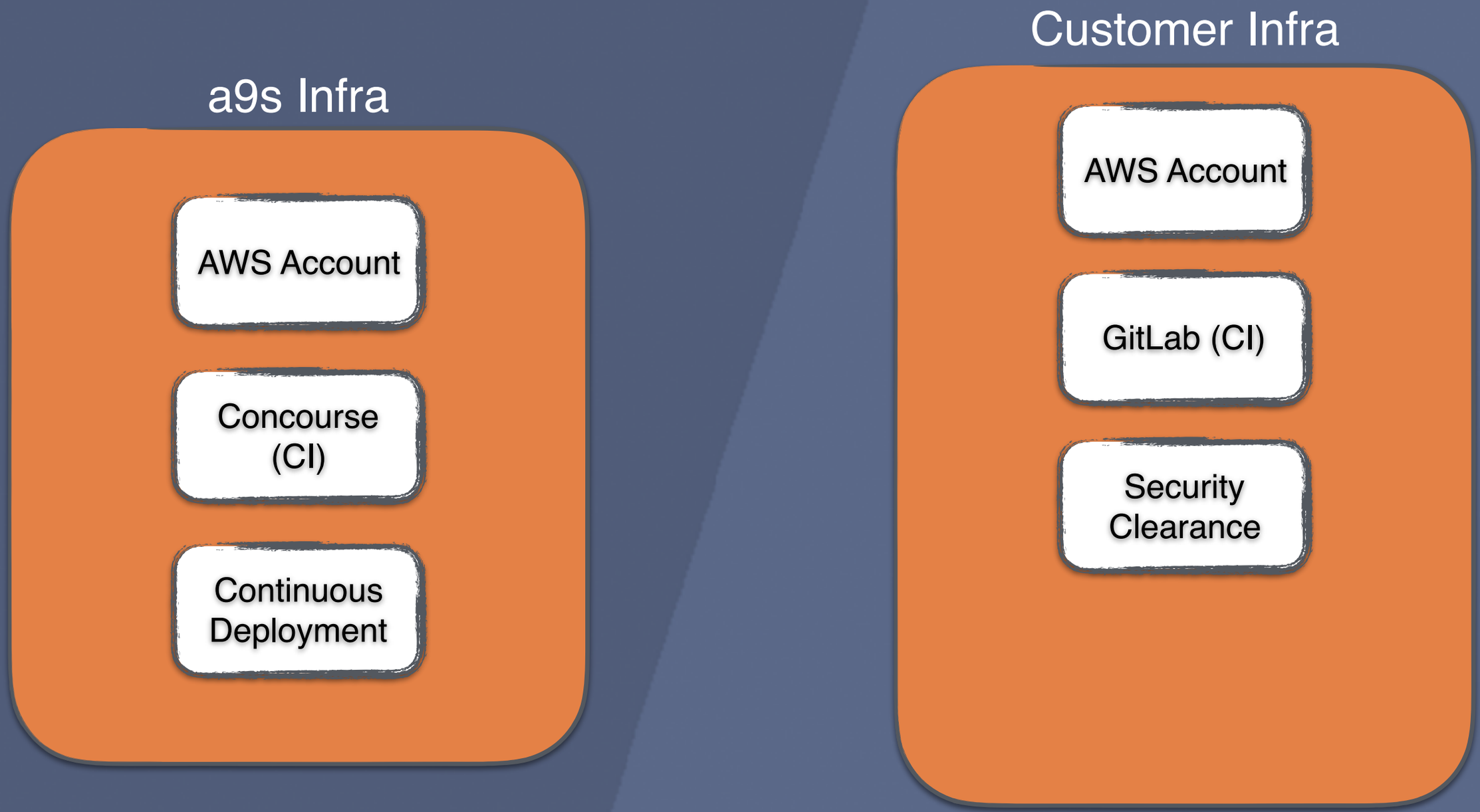
anynines

Infrastructure Divergence



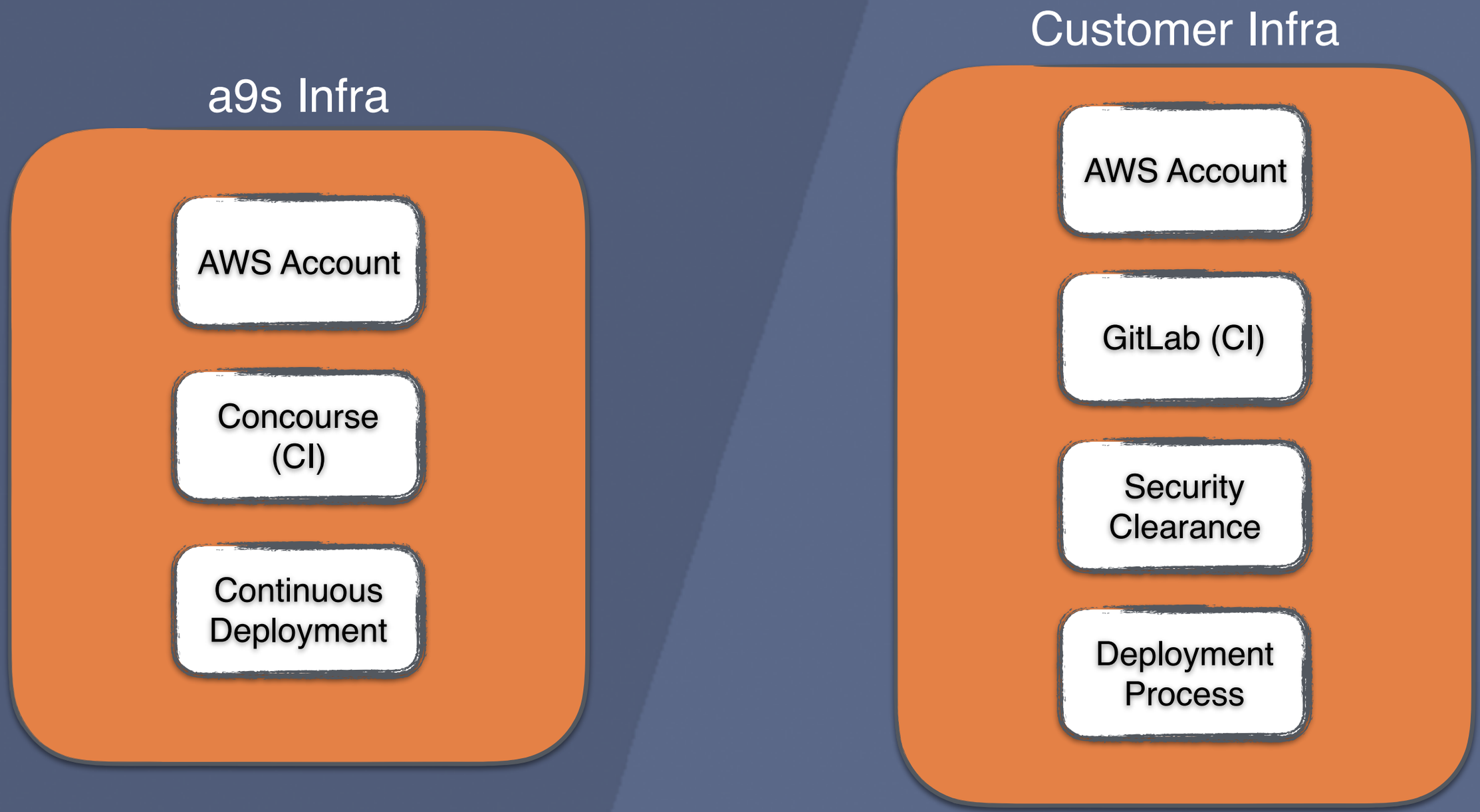
anynines

Infrastructure Divergence



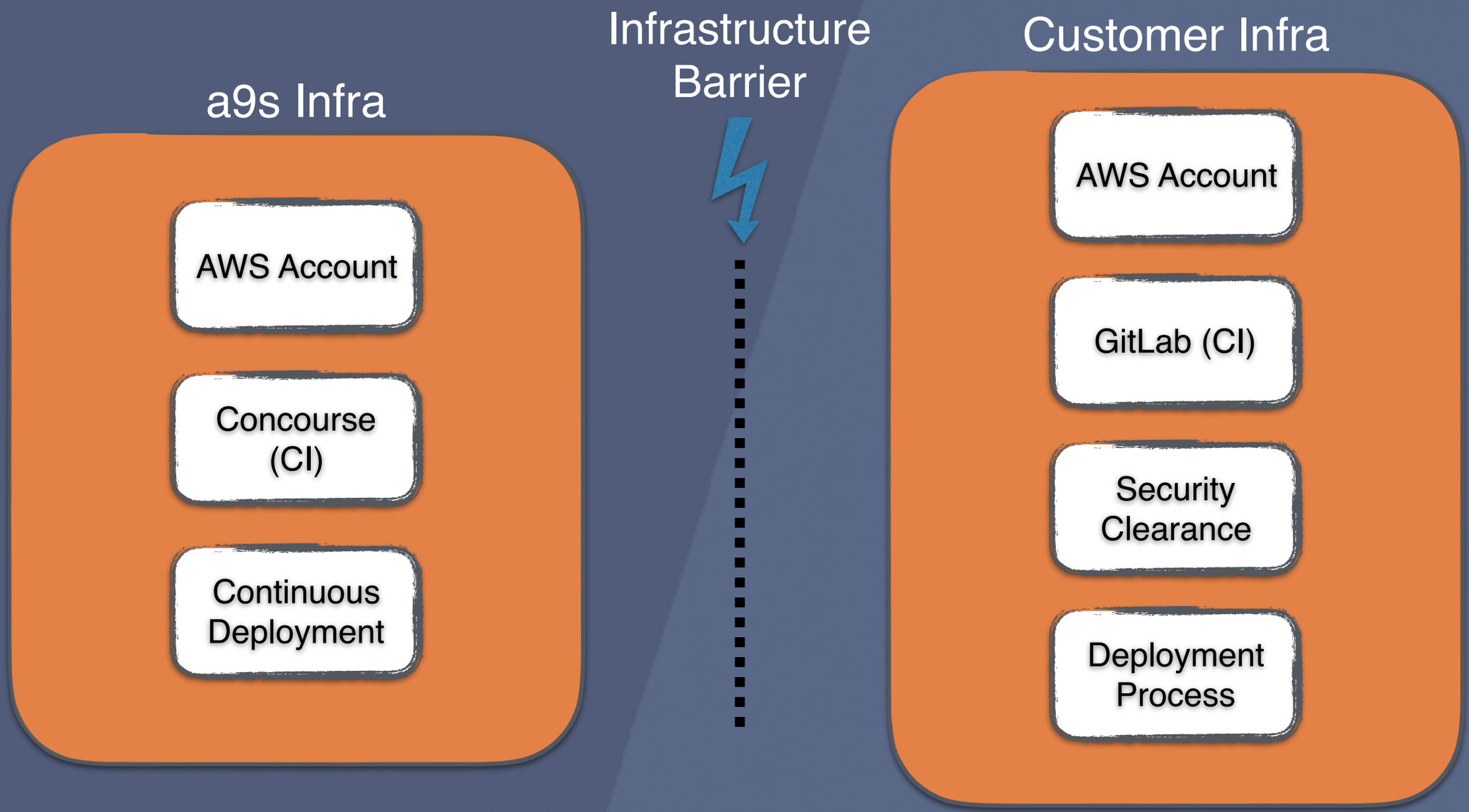
anynines

Infrastructure Divergence



anynines

Infrastructure Divergence



anynines

Remember DevOps Practices

anynines

Remember DevOps Practices

- Break Silos

anynines

Remember DevOps Practices

- Break Silos
- You build it you run it

anynines

Remember DevOps Practices

- Break Silos
- You build it you run it
- Feedback loops

anynines

Remember DevOps Practices

- Break Silos
- You build it you run it
- Feedback loops
- Automation

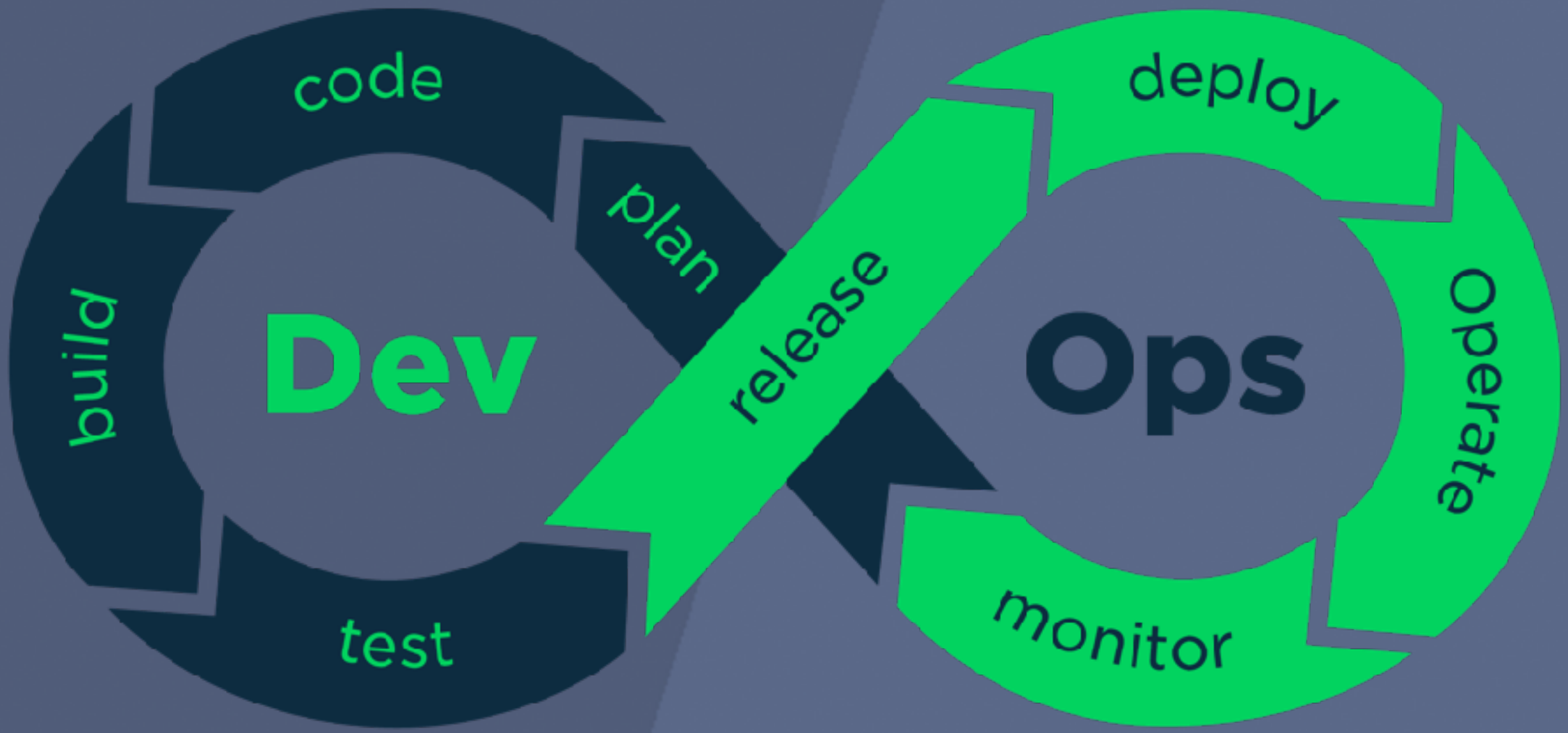
anynines

Remember DevOps Practices

- Break Silos
- You build it you run it
- Feedback loops
- Automation
- Monitoring & Alerting

anynines

DevOps



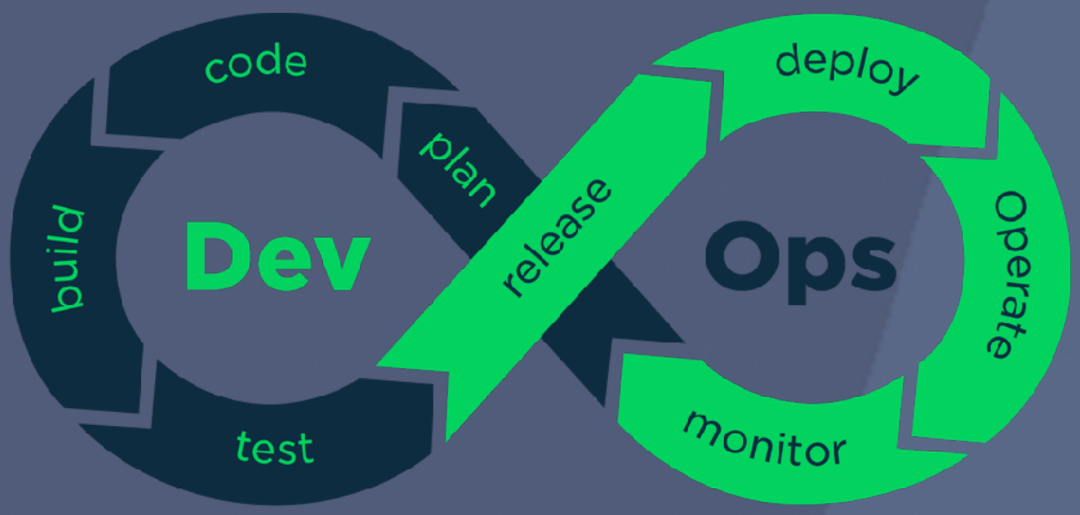
https://medium.com/@neonrocket/devops-is-a-culture-not-a-role-be1bed149b0

anynines

DevOps Barrier

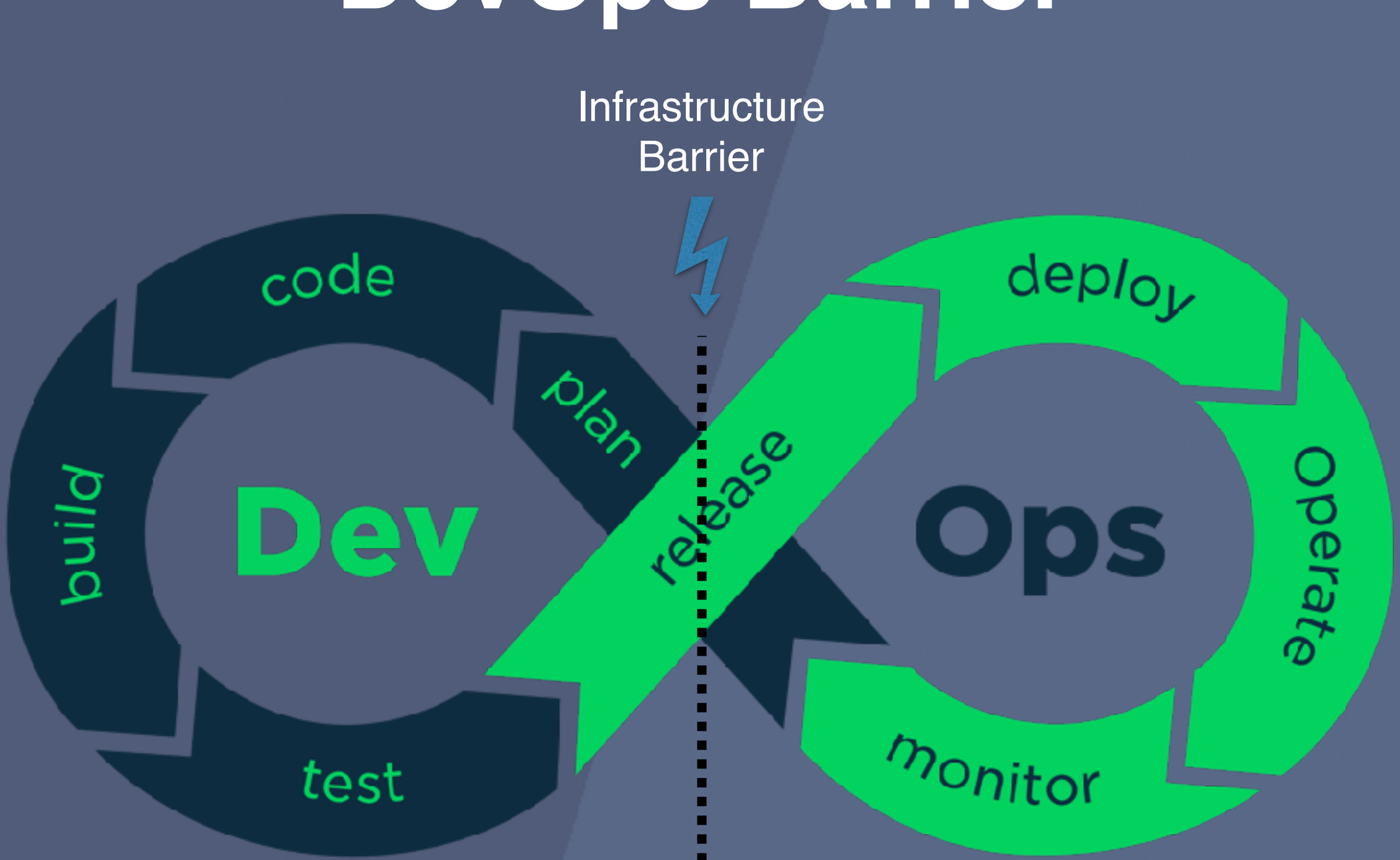
a9s

Customer



anynines

DevOps Barrier



anynines

How should we do it??

anynines

How should we do it??

- We cannot break down the silos completely

anynines

How should we do it??

- We cannot break down the silos completely
- But we can build some bridges

anynines

How should we do it??

- We cannot break down the silos completely
- But we can build some bridges
- Sync as much as possible

anynines

How should we do it??

- We cannot break down the silos completely
- But we can build some bridges
- Sync as much as possible
- Adapt our process according to customers because we won’t change customers mindset

anynines

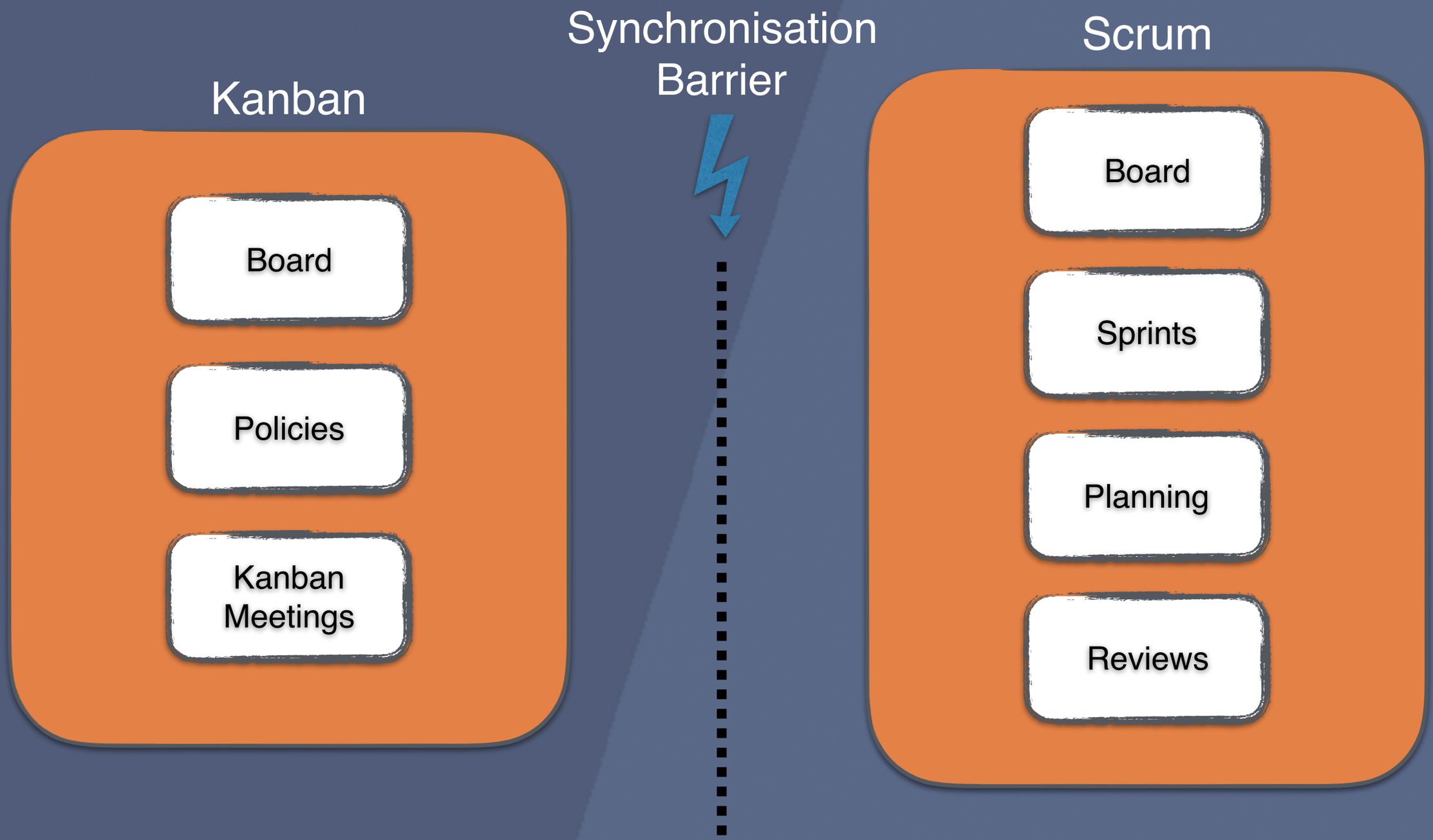
Solutions

anynines

Process Divergence

anynines

Process Divergence



anynines

Process Sync

Kanban

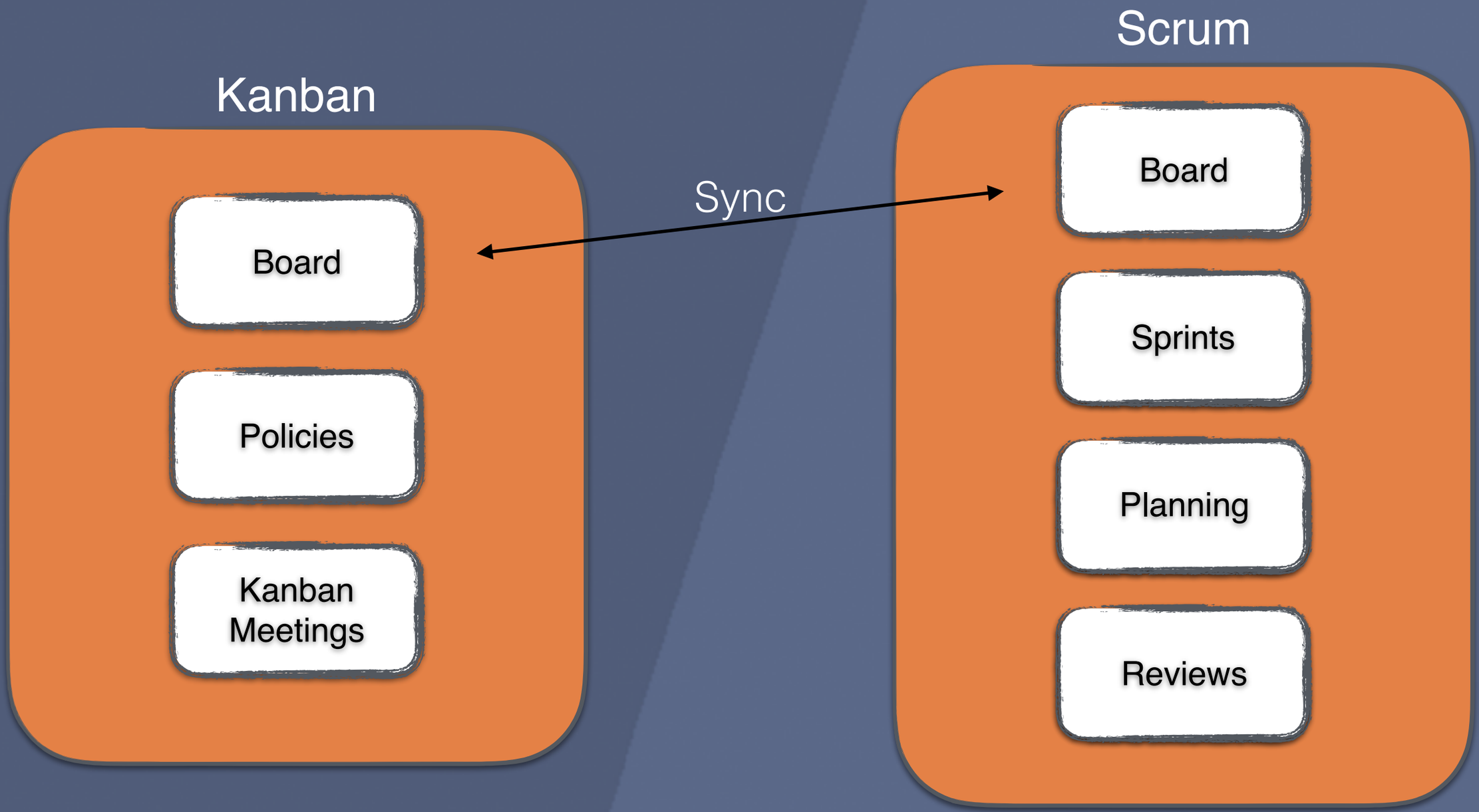
- Board
- Policies
- Kanban Meetings

Scrum

- Board
- Sprints
- Planning
- Reviews

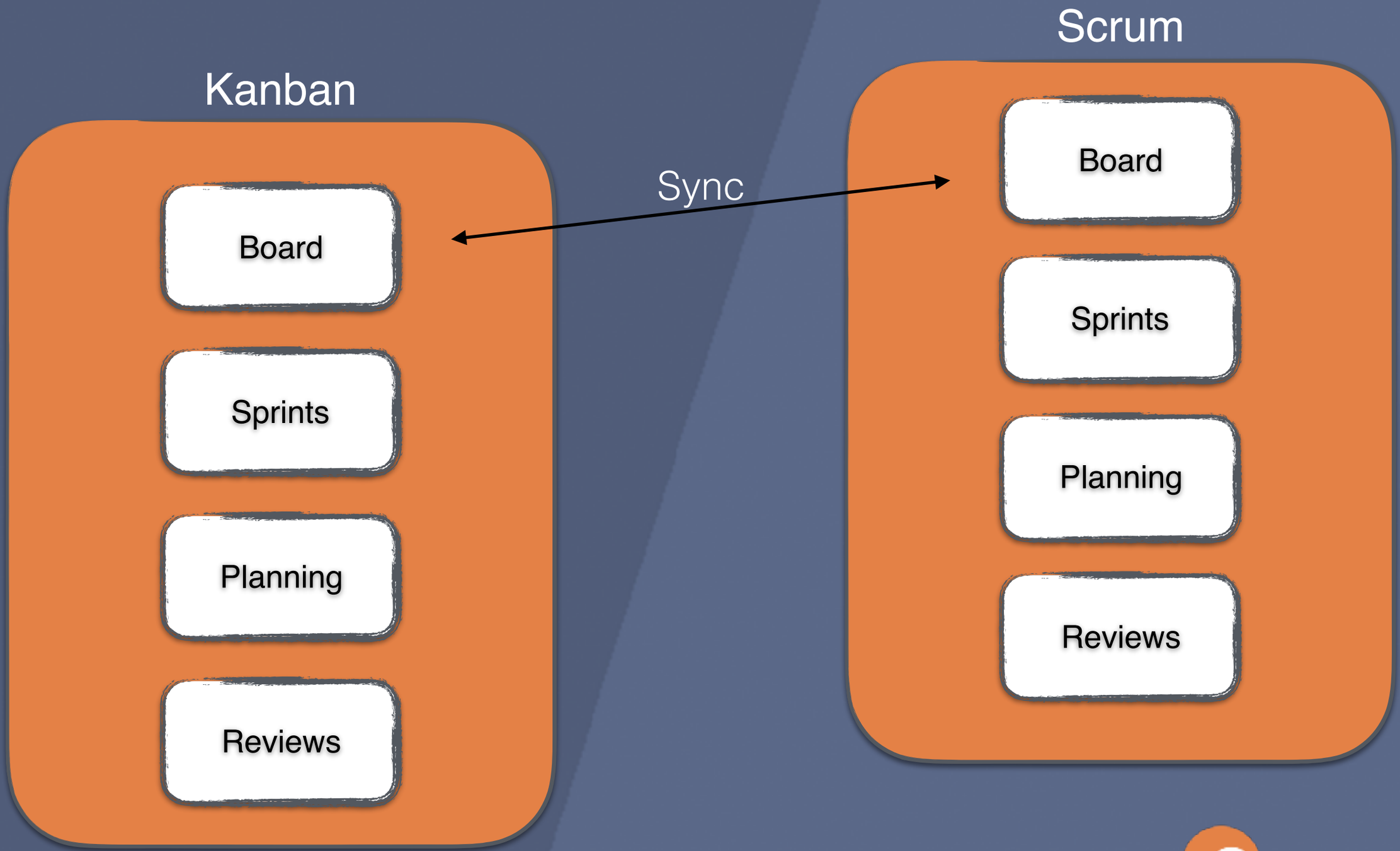
anynines

Process Sync



anynines

Process Sync

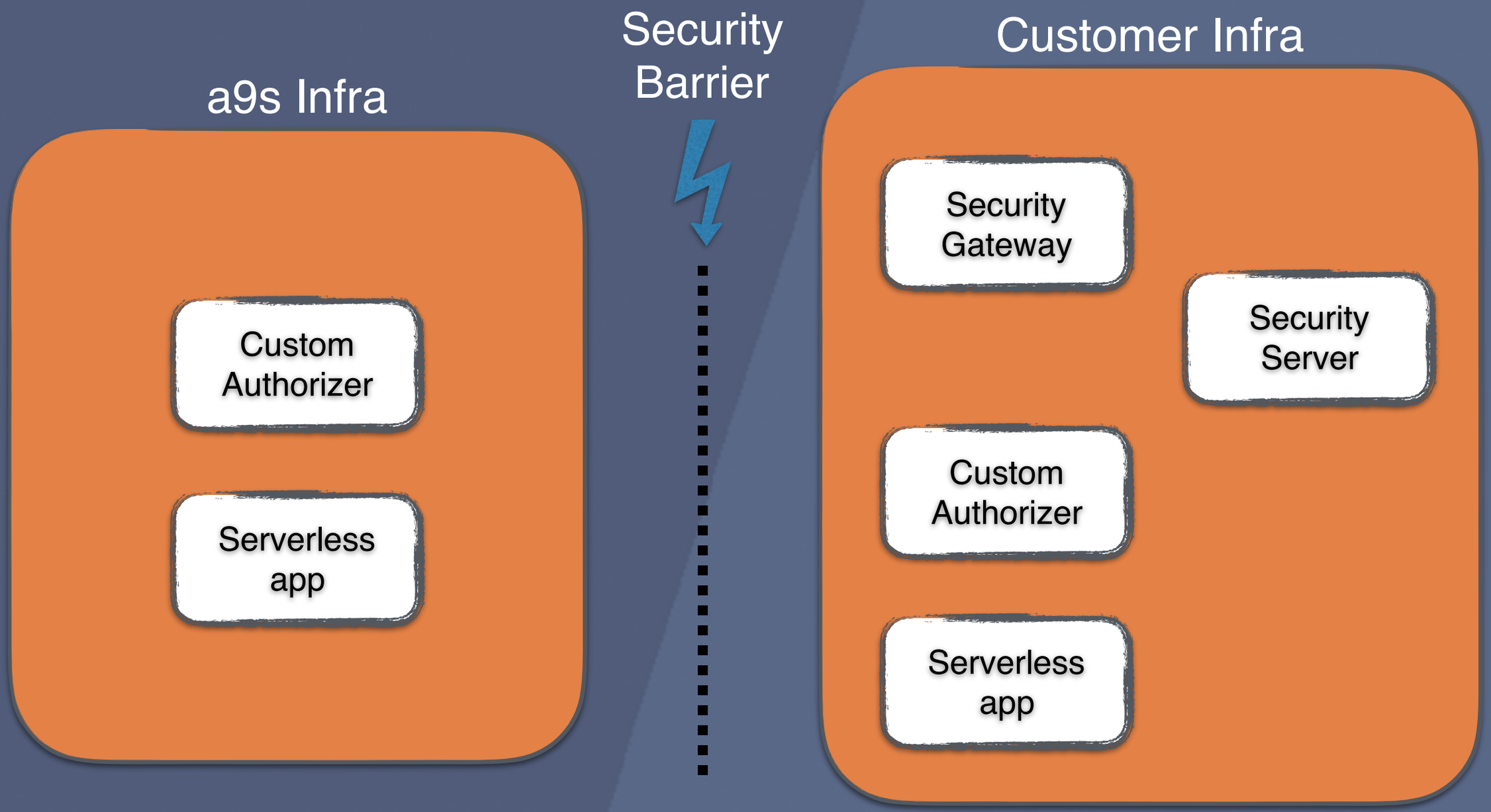


anynines

Context/Security Divergence

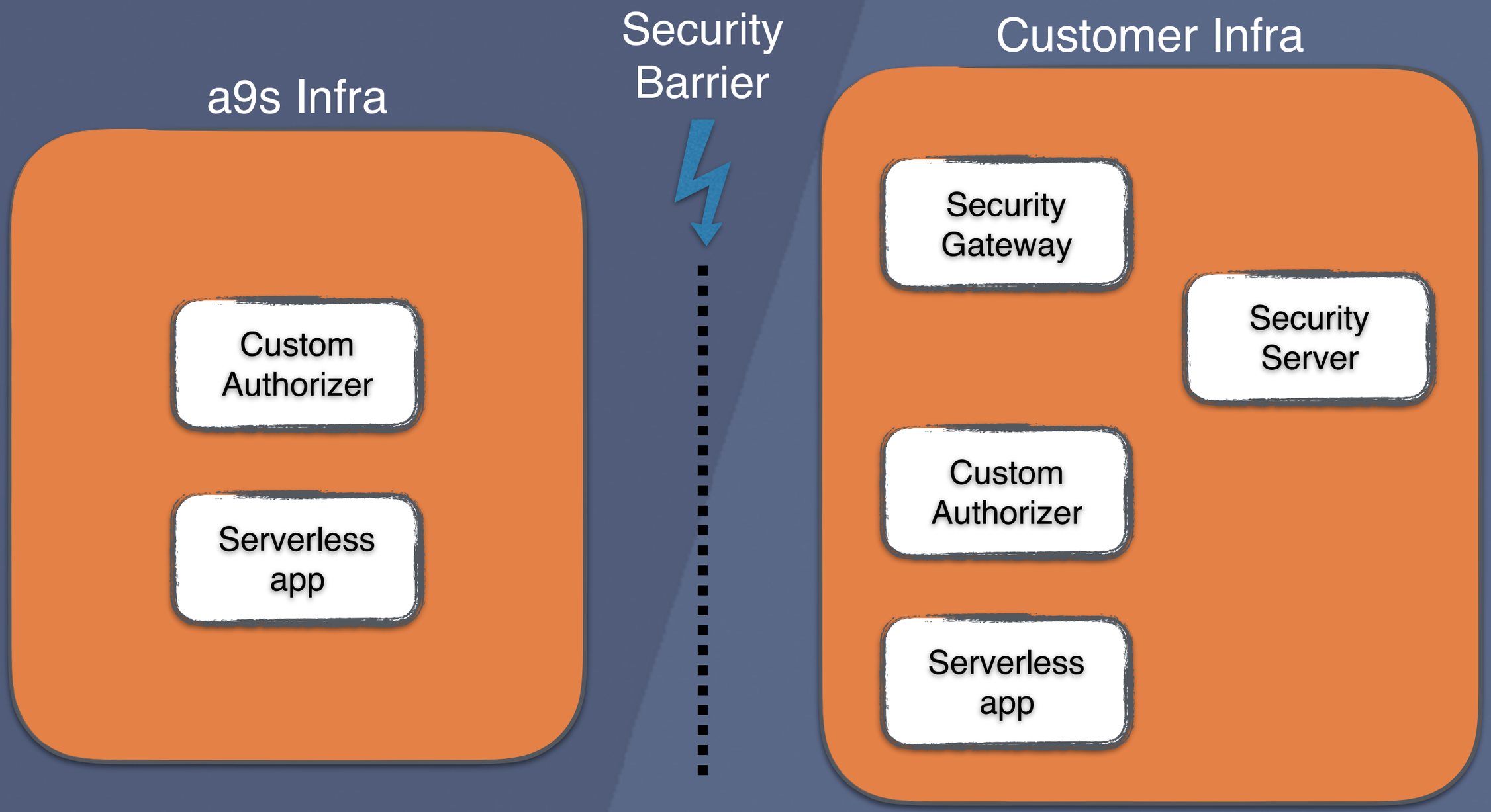
anynines

Context/Security Divergence



anynines

Context/Security Divergence

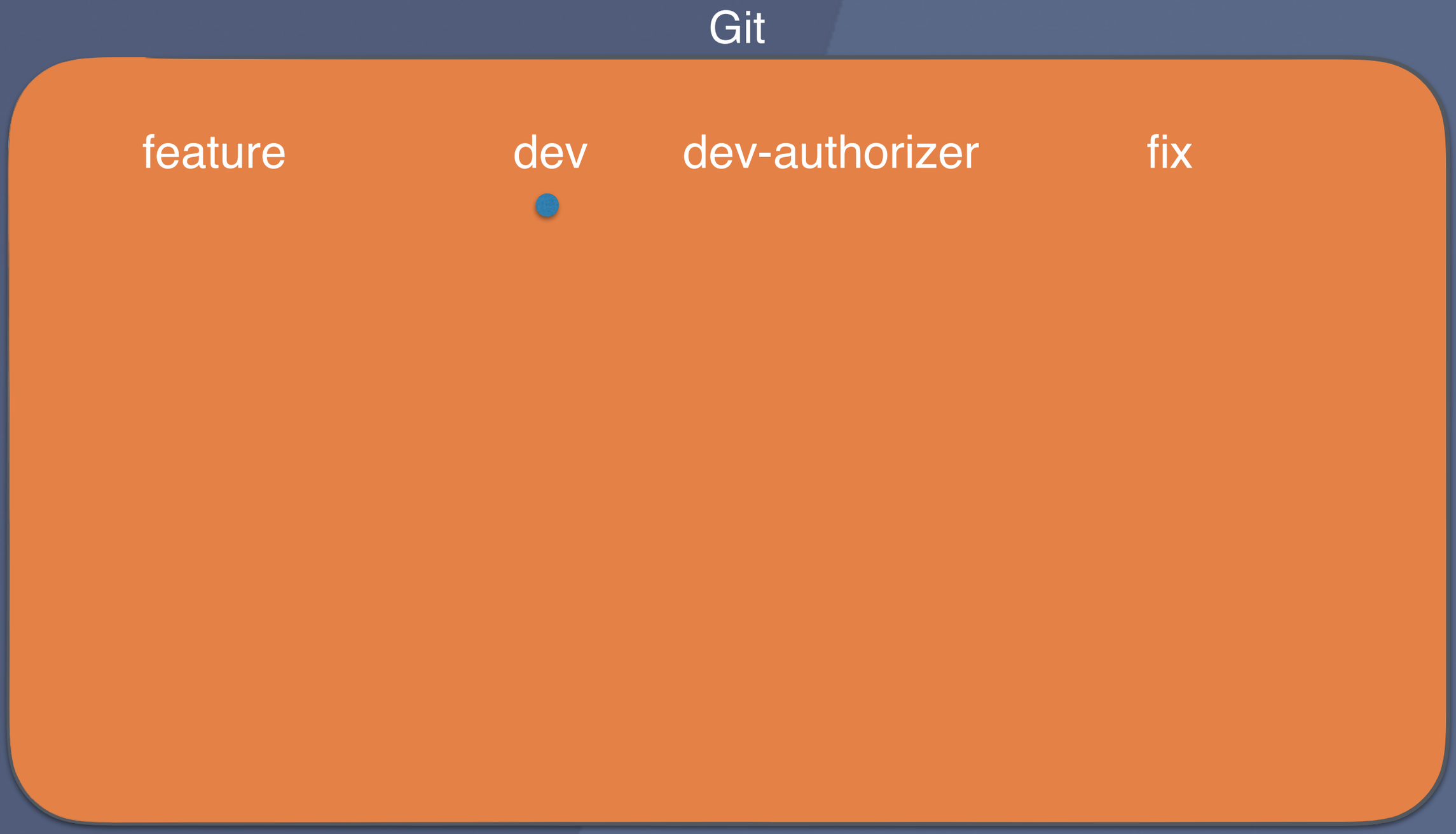


anynines

Context/Security Sync (1. Approach)

anynines

Context/Security Sync (1. Approach)



anynines

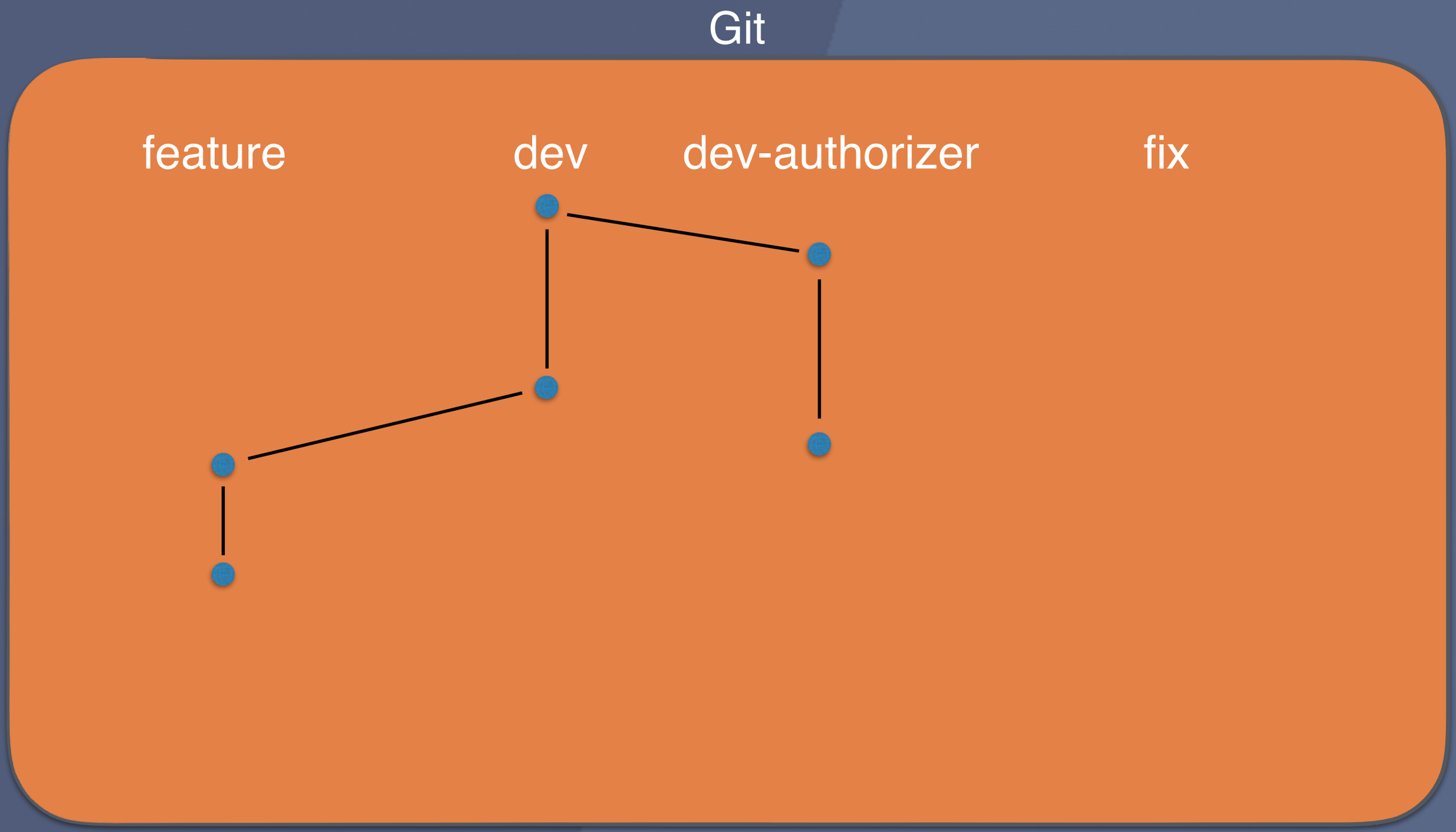
Context/Security Sync (1. Approach)

anynines

Context/Security Sync (1. Approach)

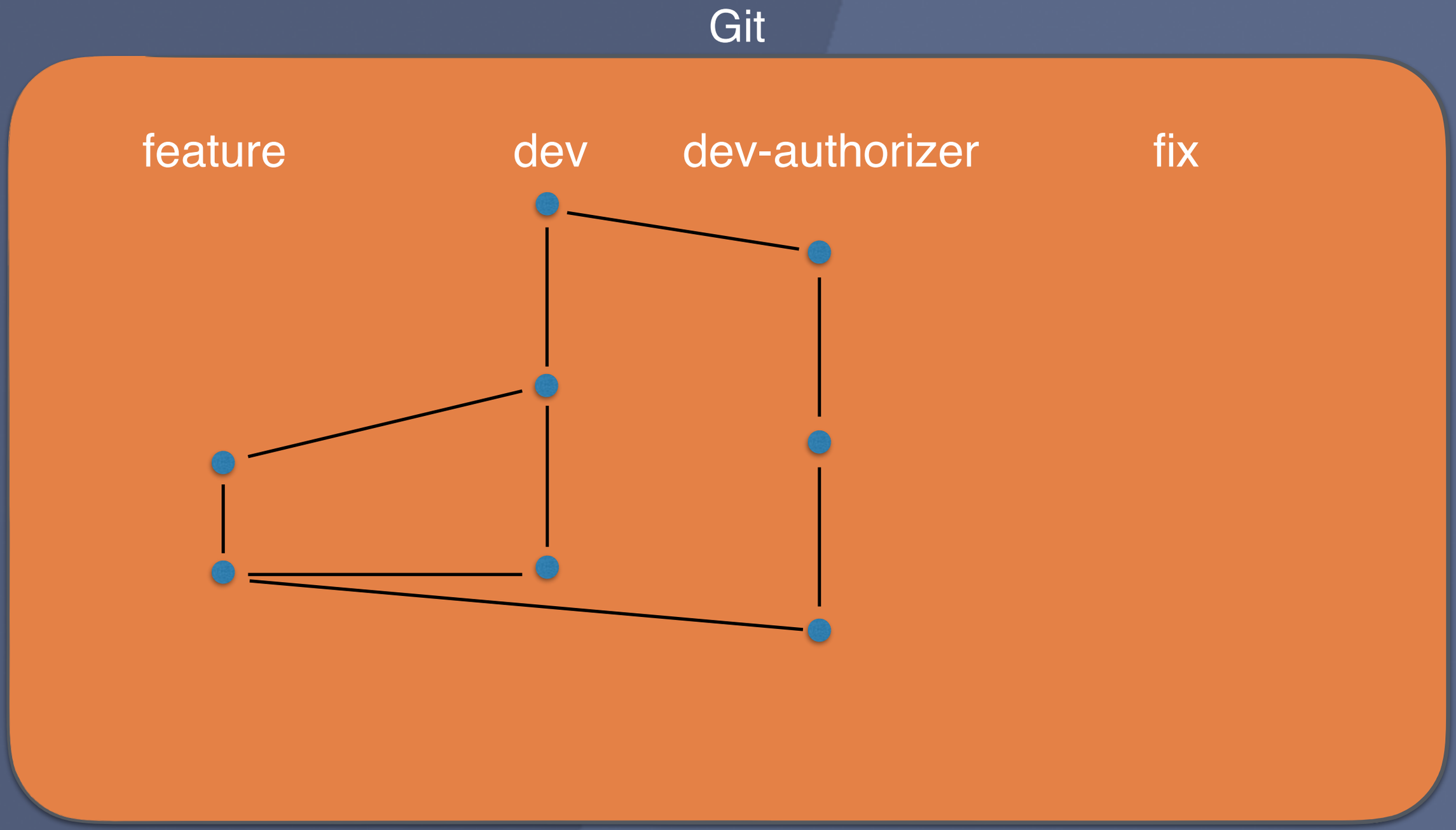
anynines

Context/Security Sync (1. Approach)



anynines

Context/Security Sync (1. Approach)

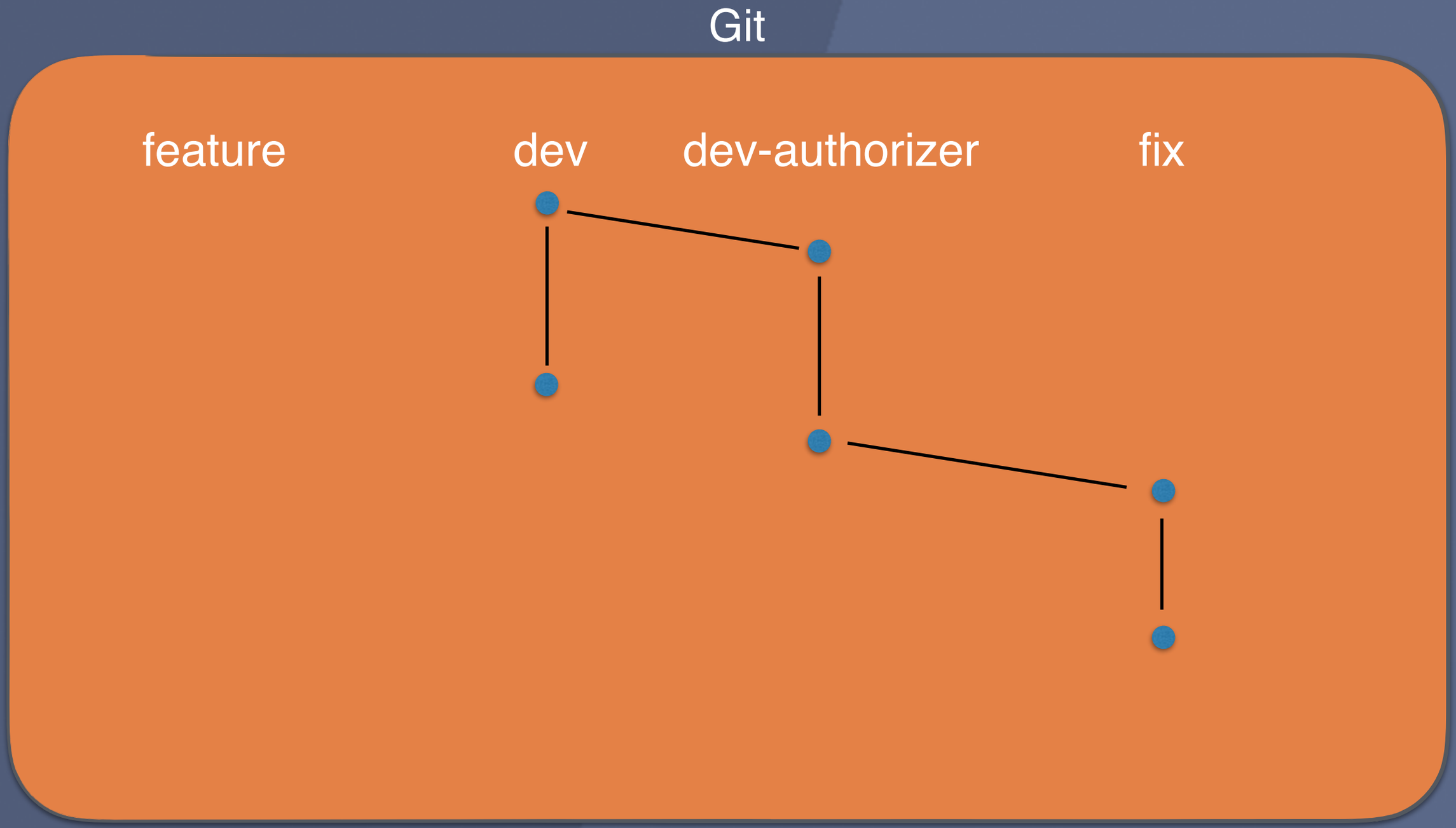


anynines

Context/Security Sync (1. Approach)

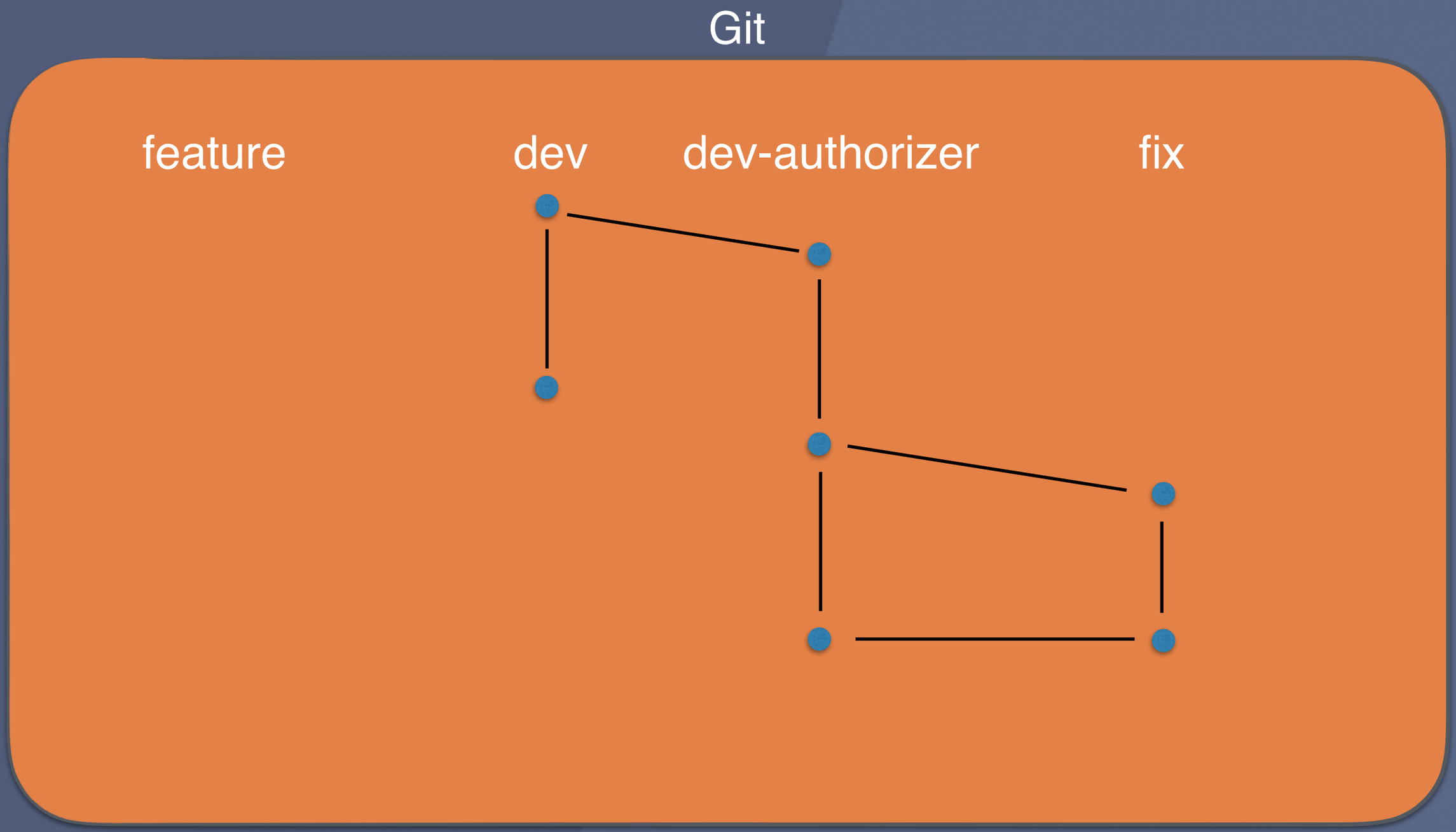
anynines

Context/Security Sync (1. Approach)



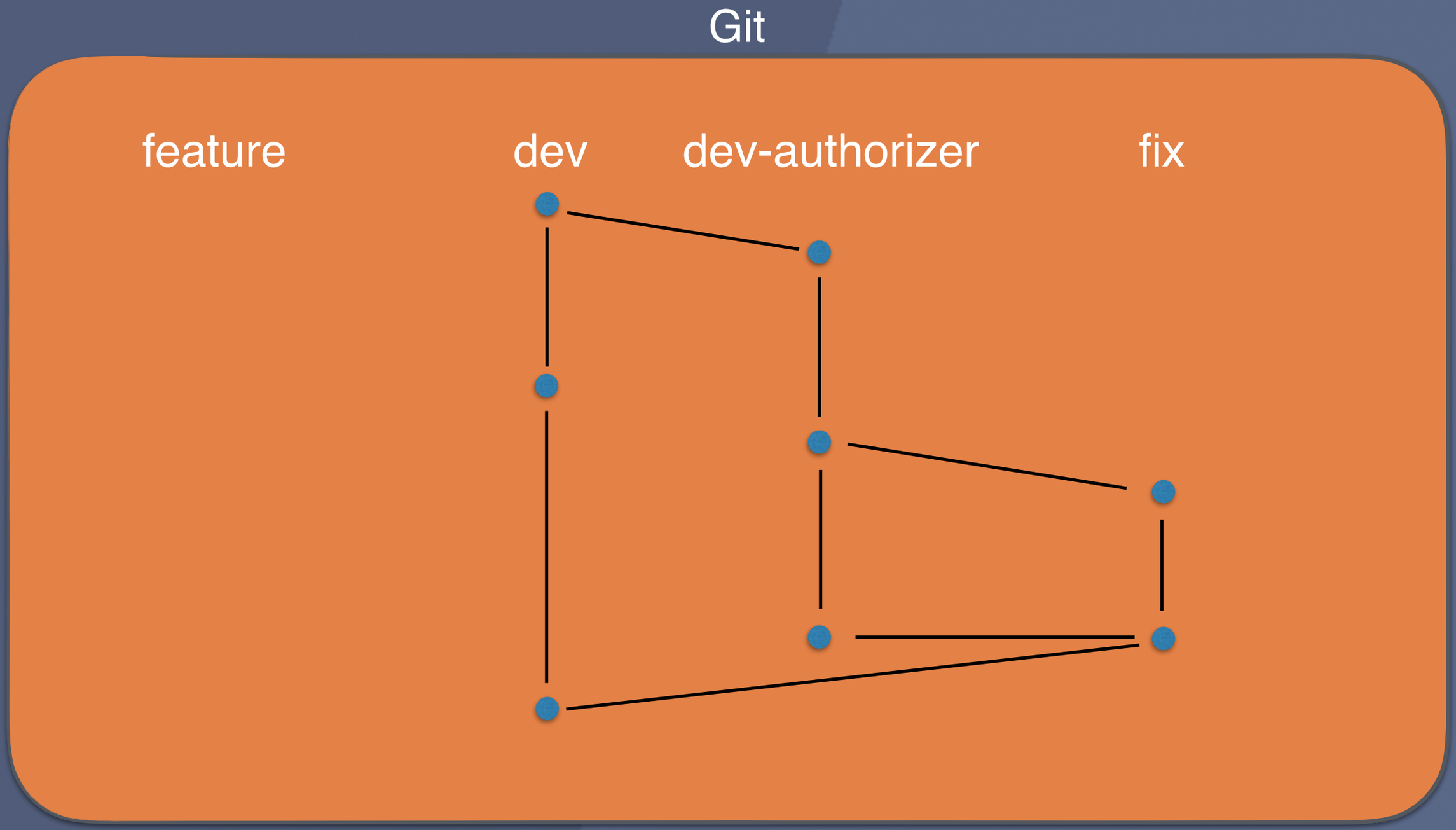
anynines

Context/Security Sync (1. Approach)



anynines

Context/Security Sync (1. Approach)

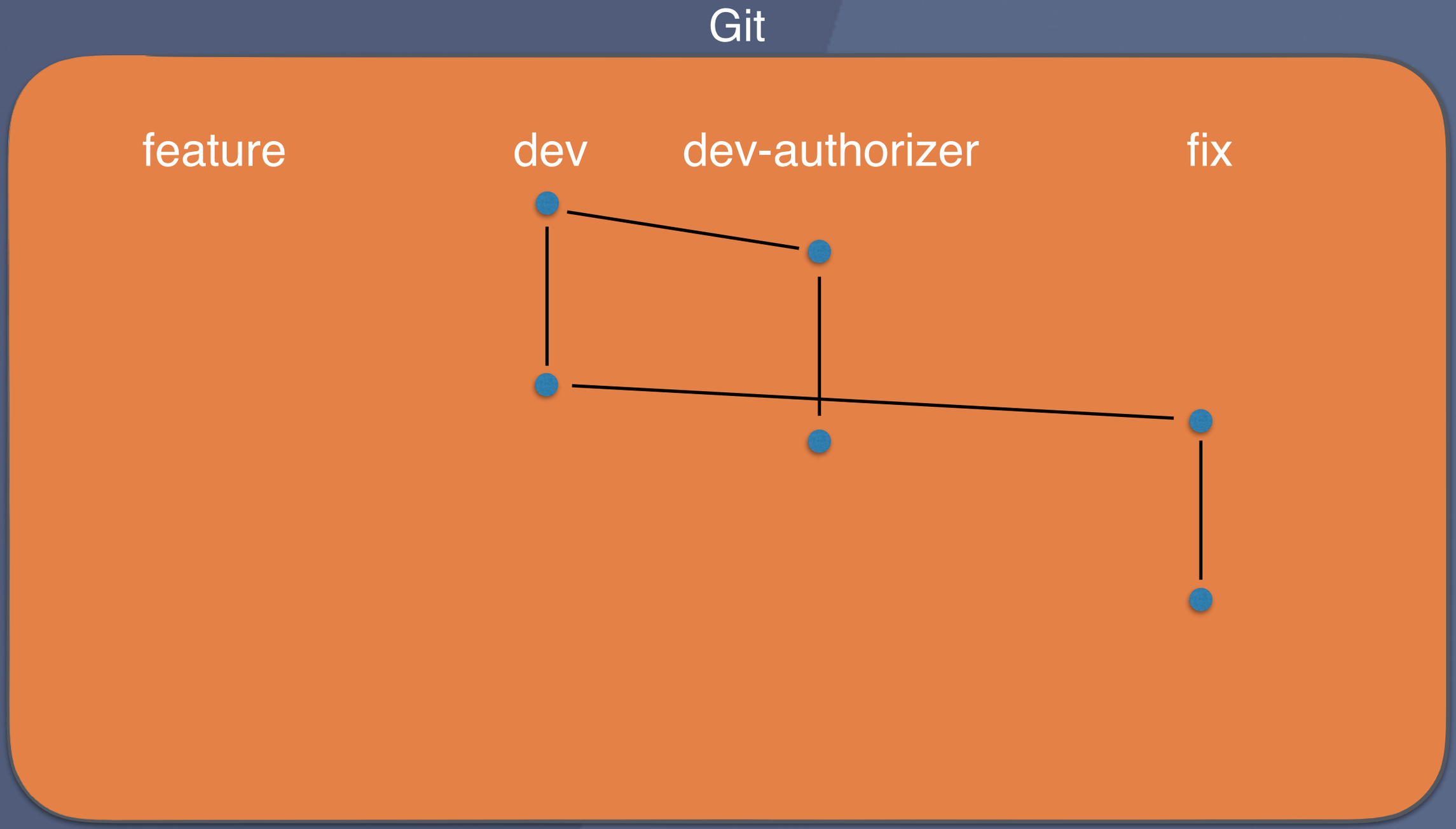


anynines

Context/Security Sync (1. Approach)

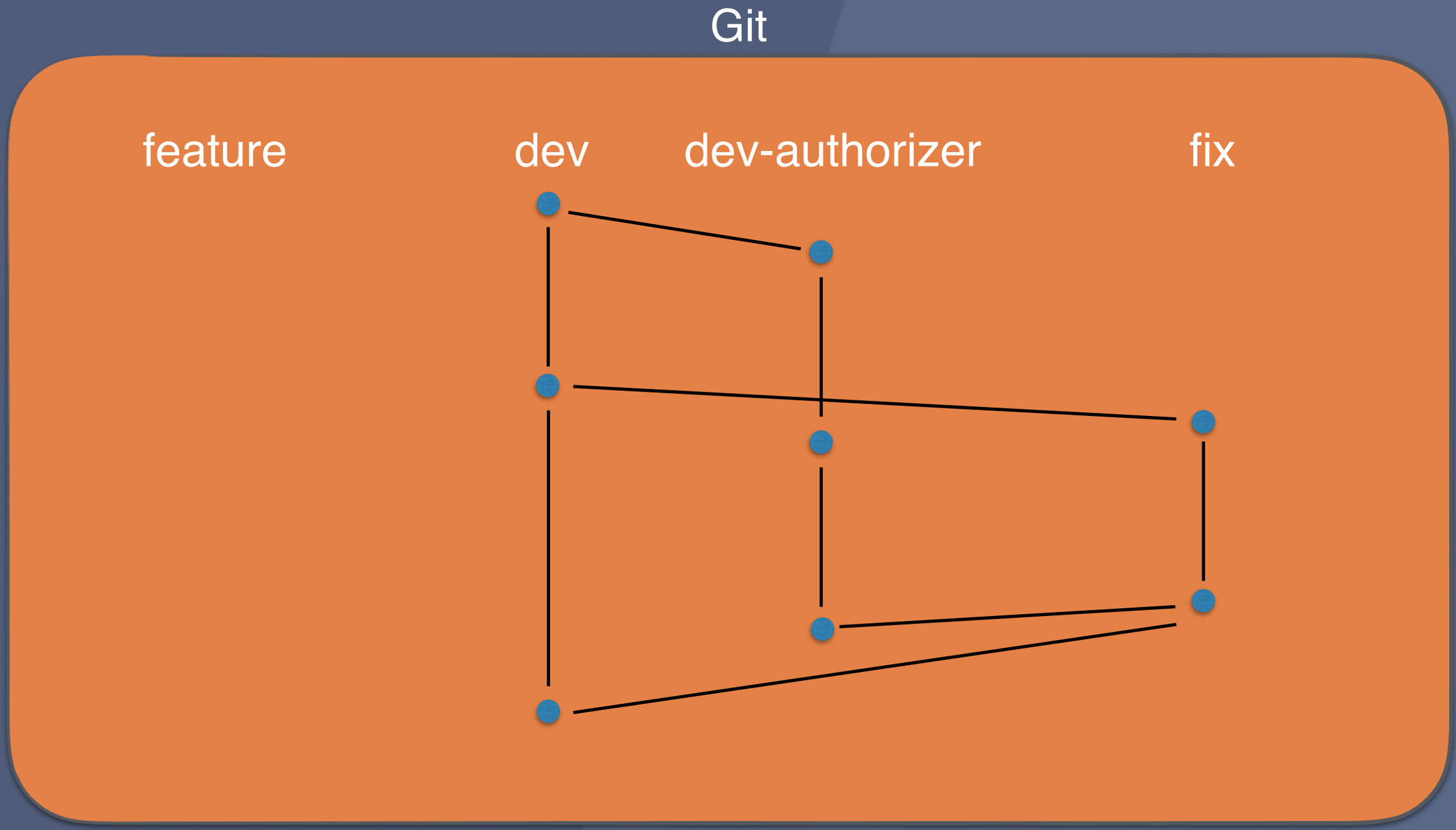
anynines

Context/Security Sync (1. Approach)



anynines

Context/Security Sync (1. Approach)

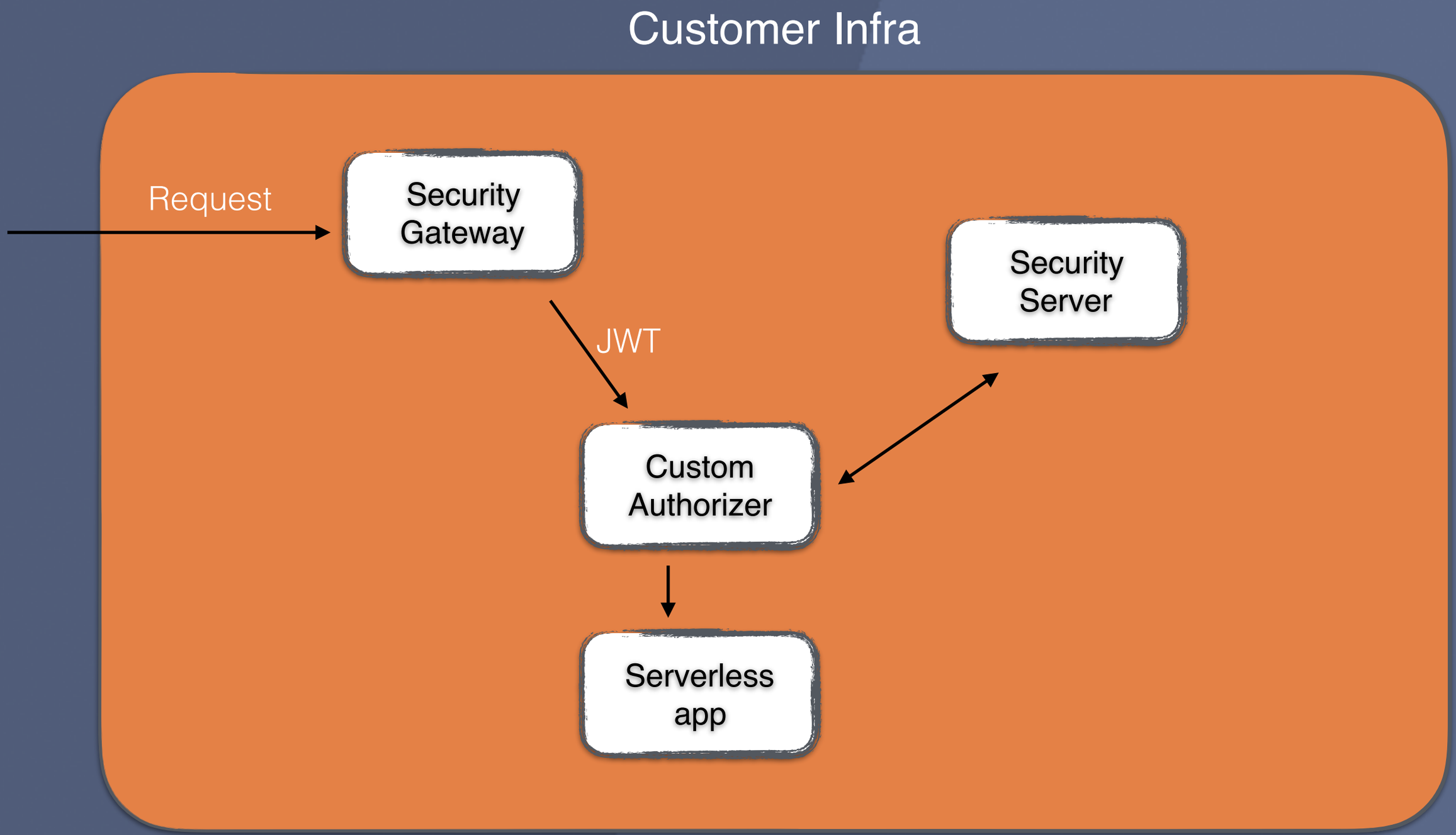


anynines

Context/Security Sync (1. Approach)

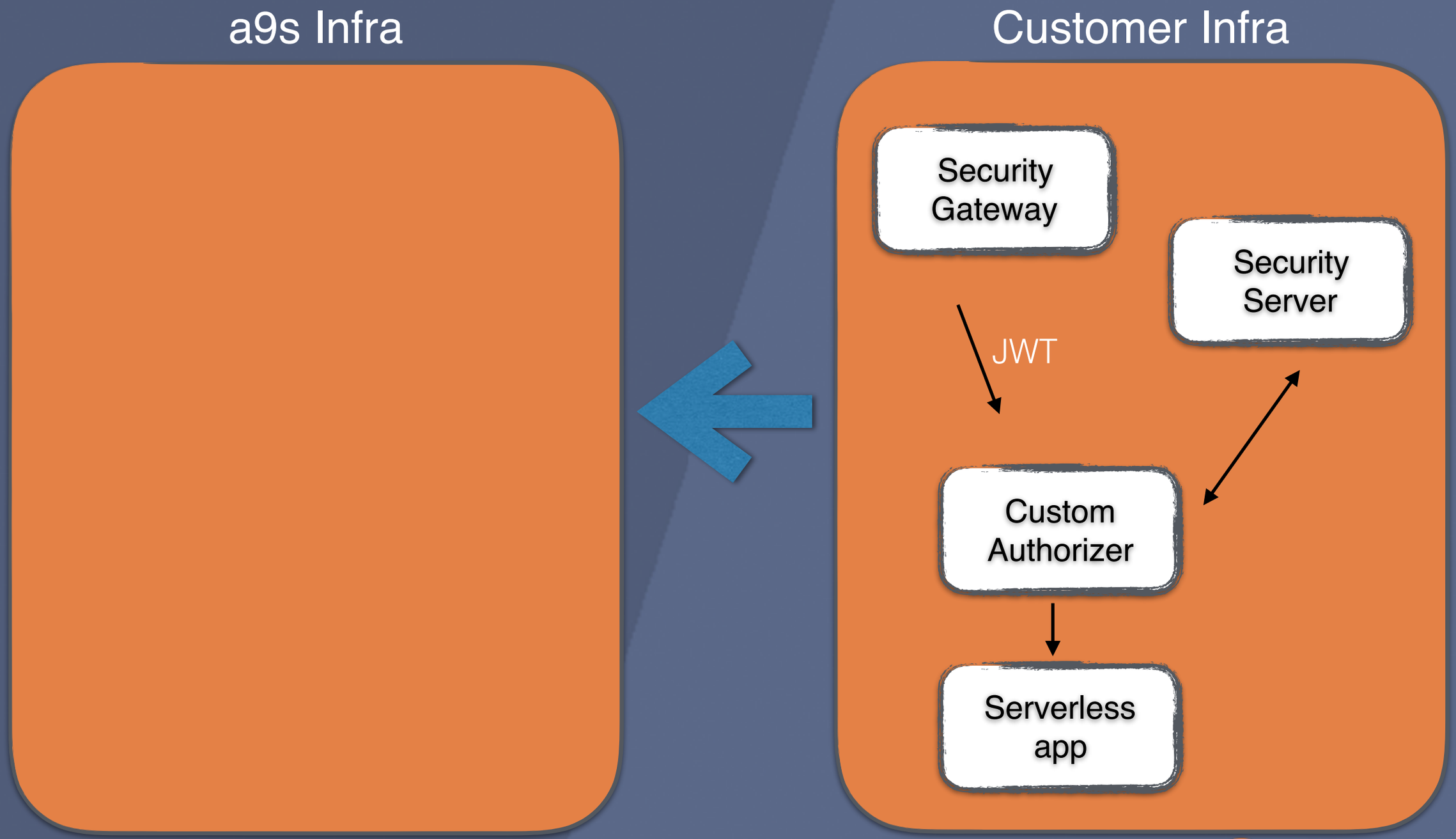
anynines

Context/Security Sync



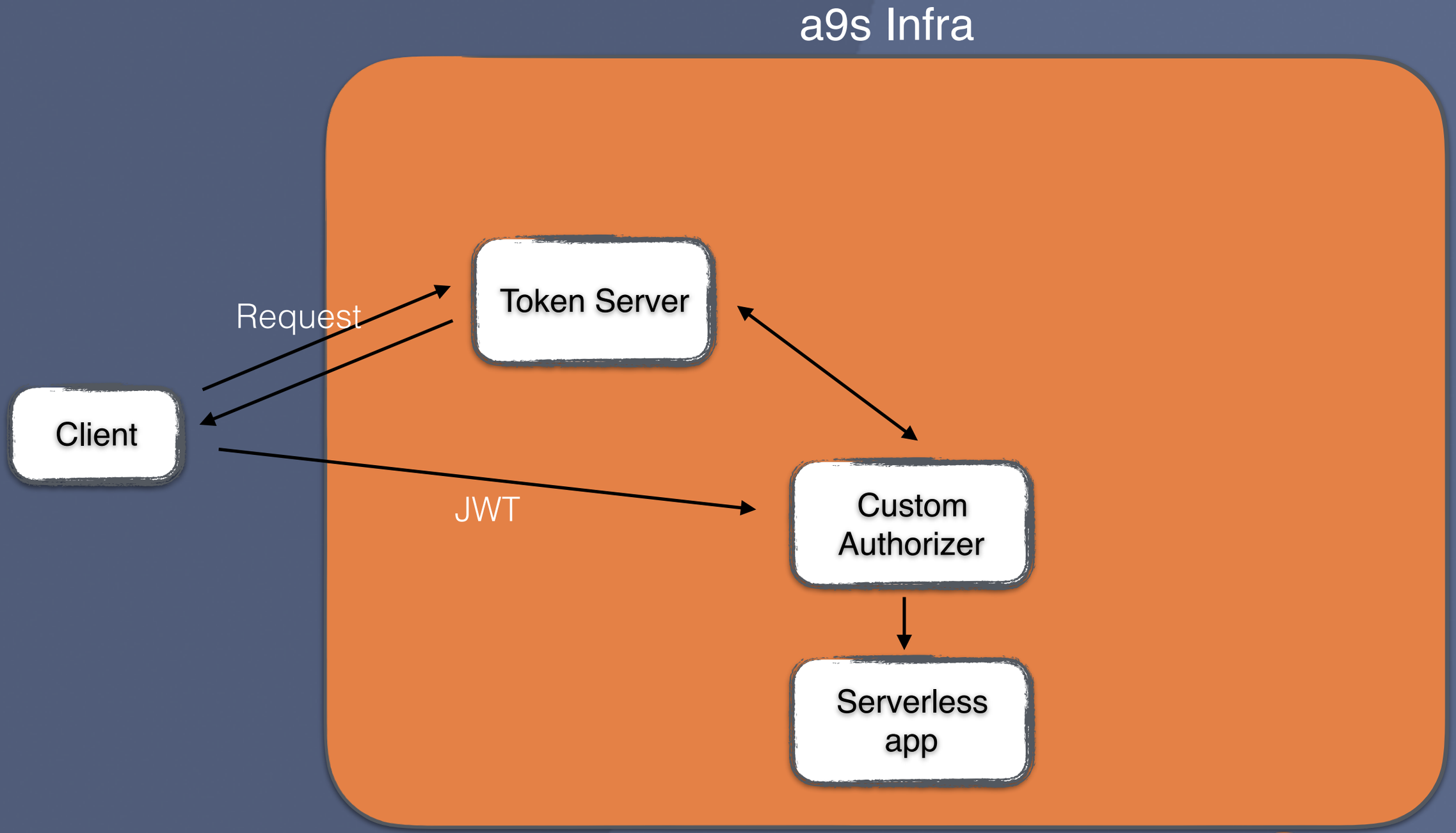
anynines

Context/Security Sync



anynines

Context/Security Sync

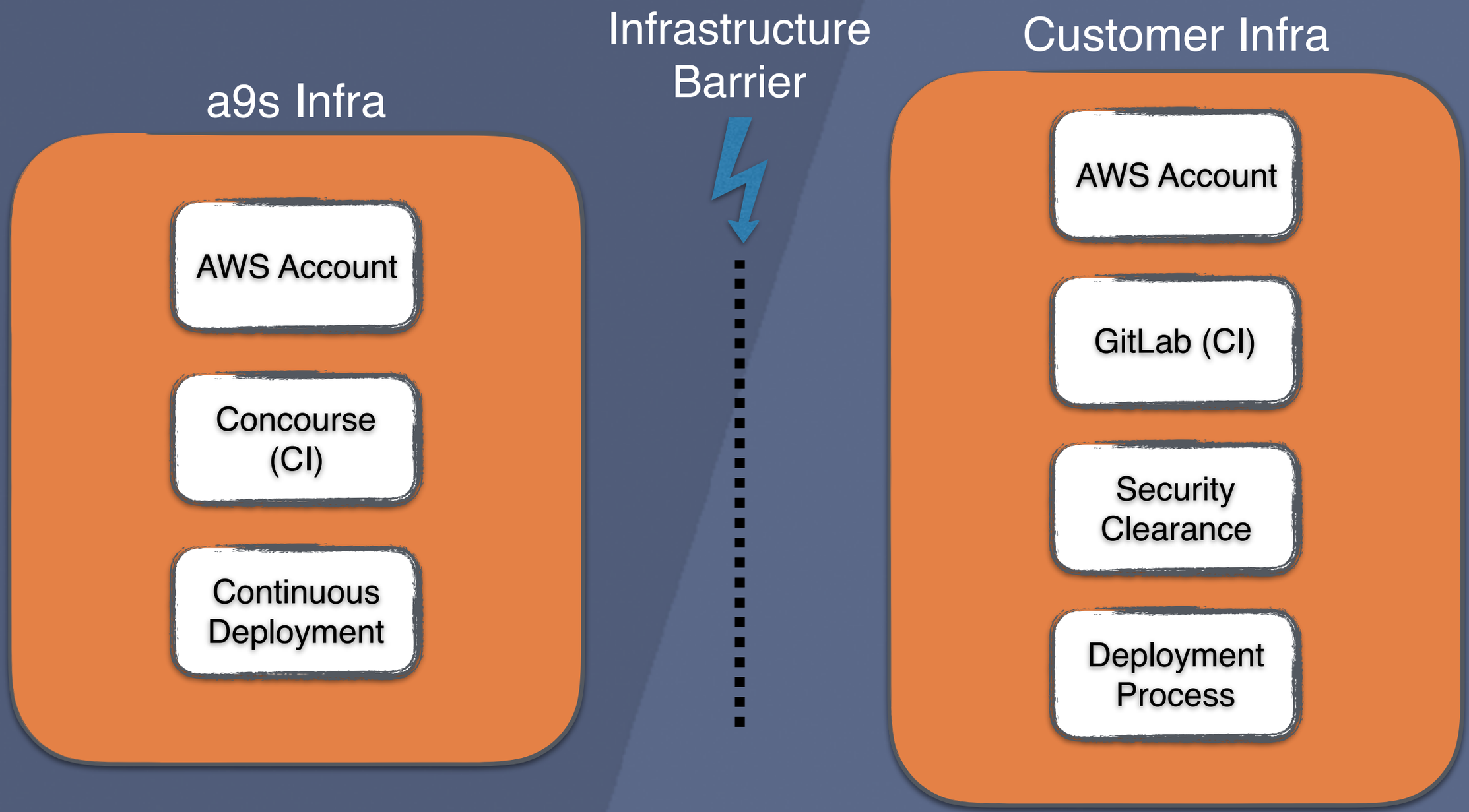


anynines

Infrastructure Divergence

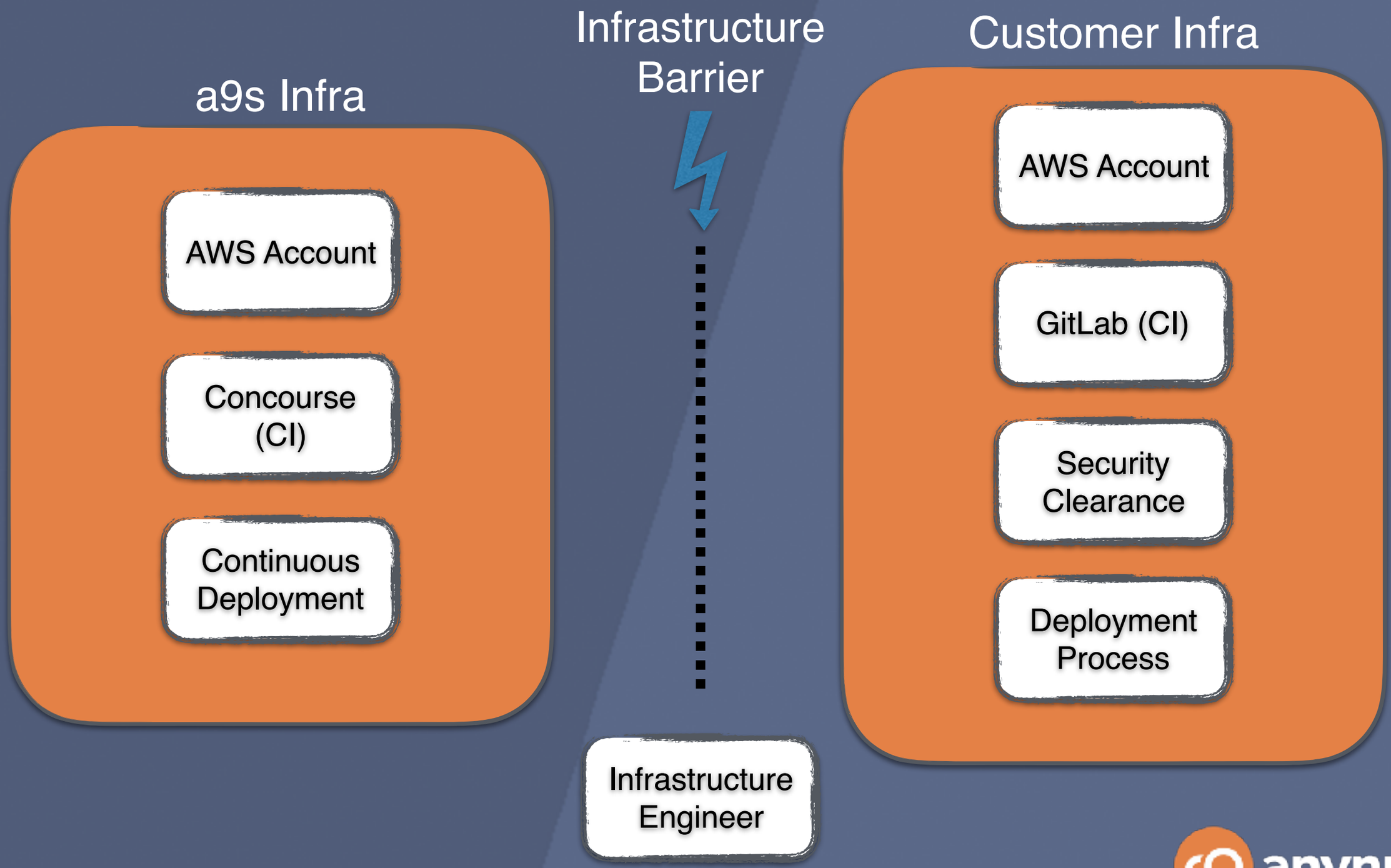
anynines

Infrastructure Divergence



anynines

Infrastructure Divergence

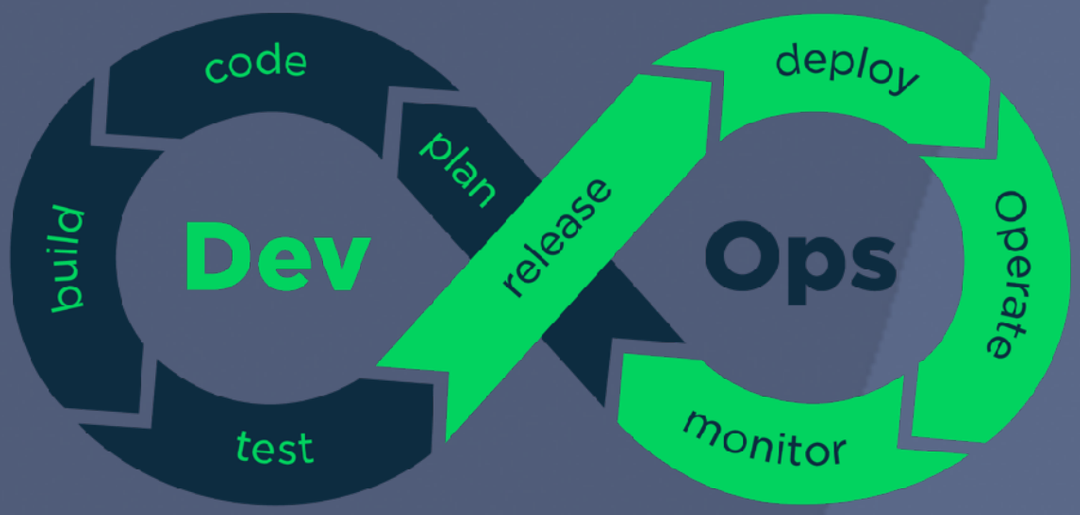


anynines

DevOps Sync

a9s

Customer



anynines

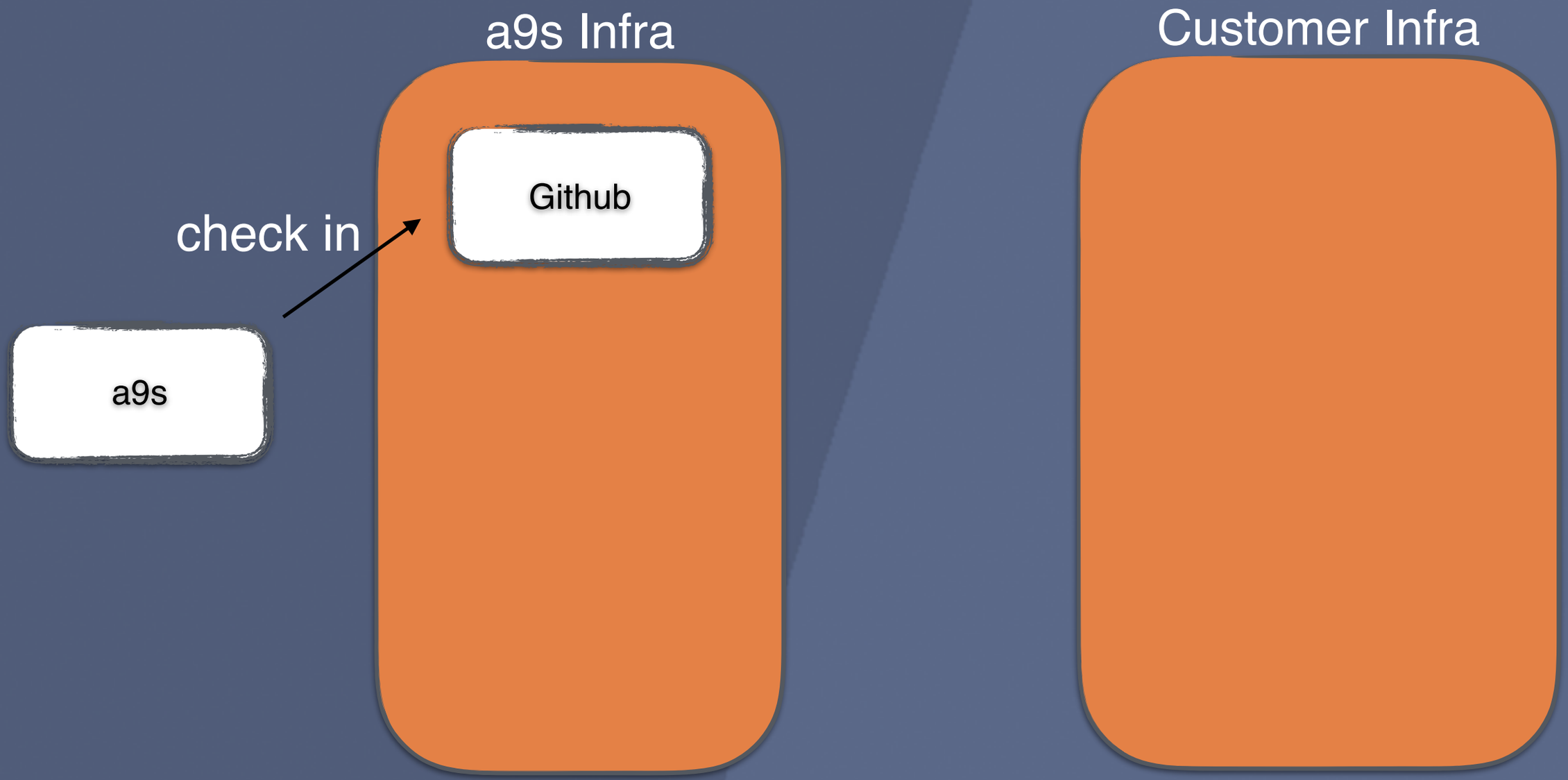
DevOps Sync

a9s Infra

Customer Infra

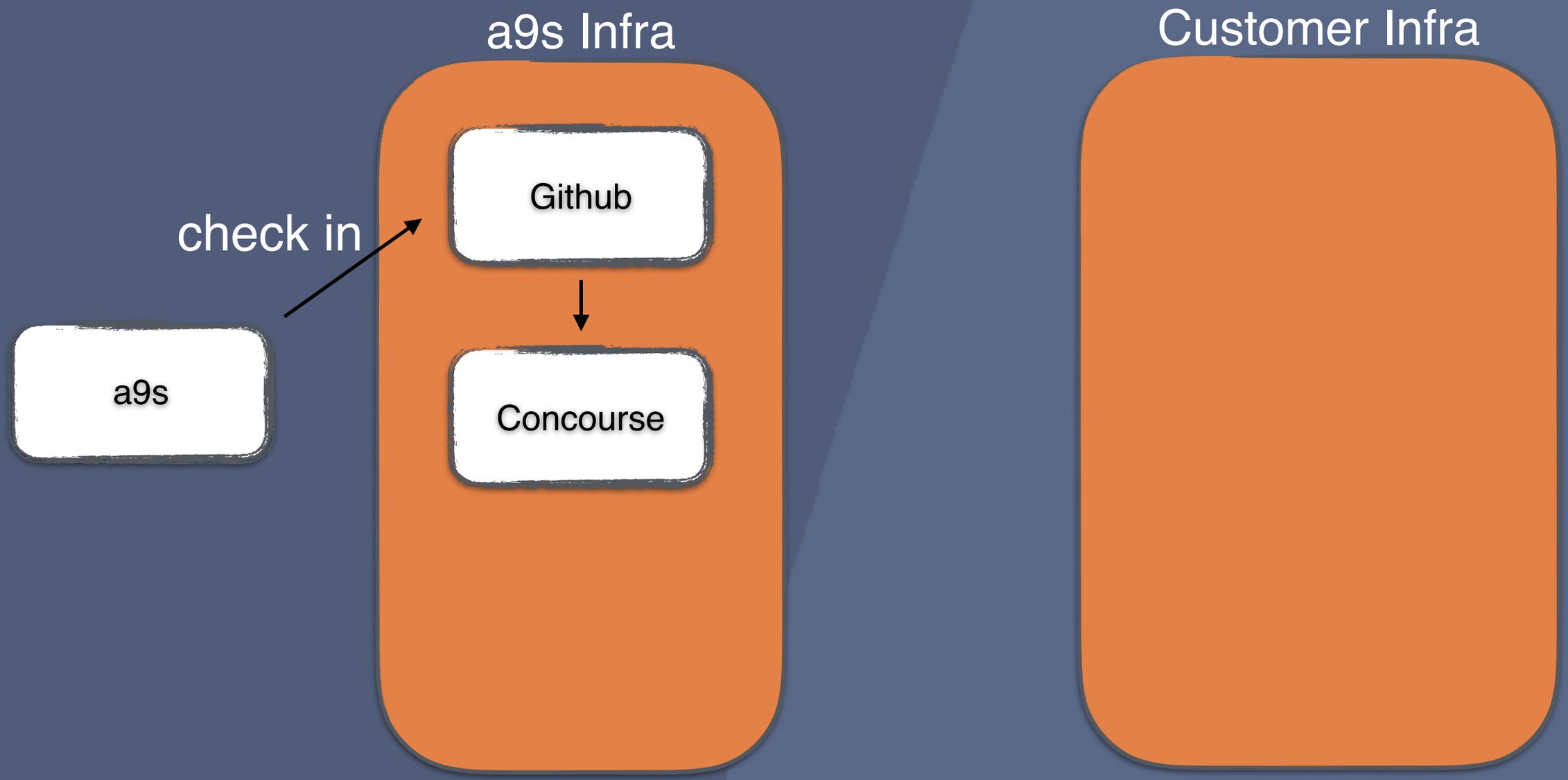
anynines

DevOps Sync



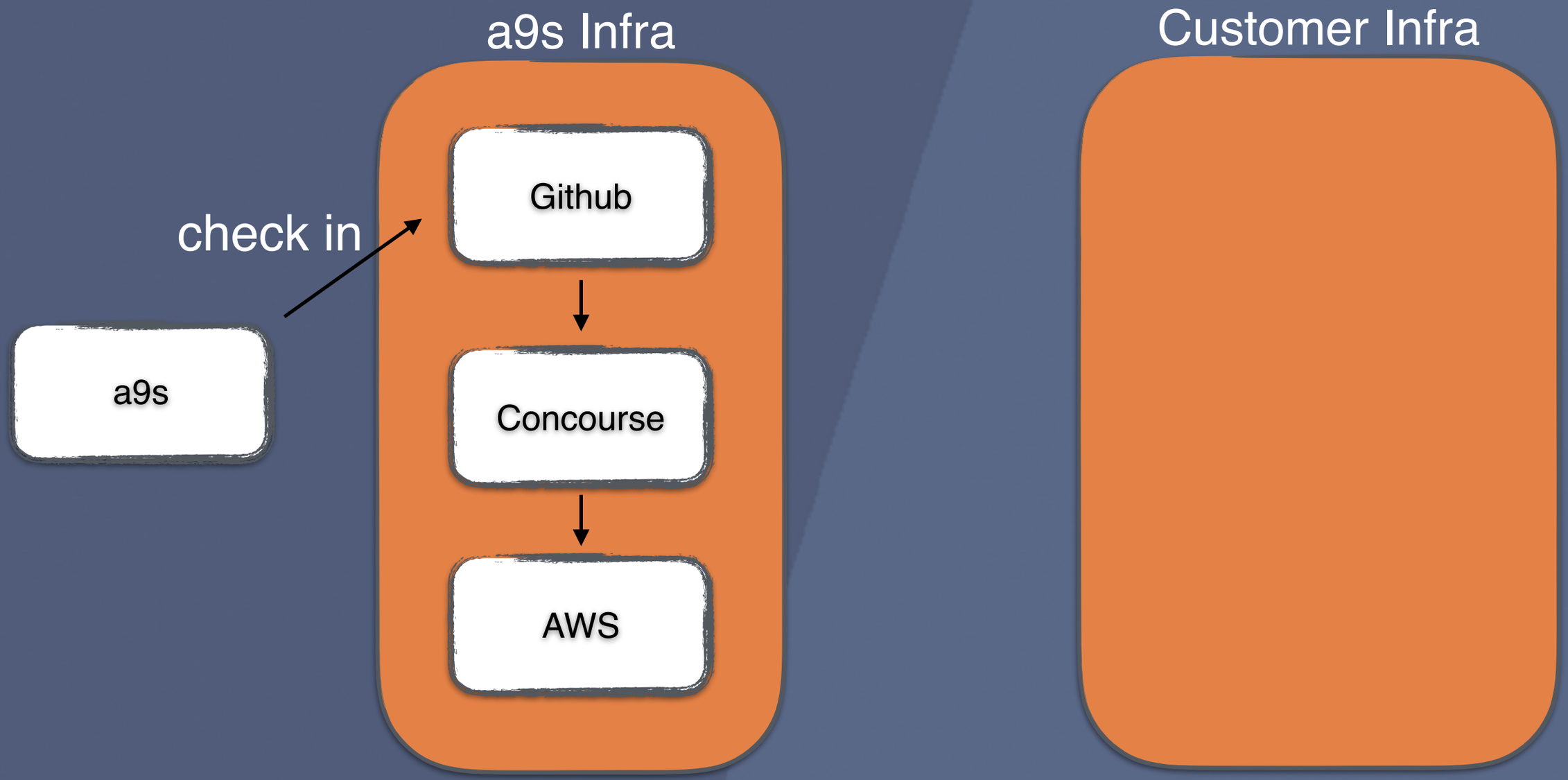
anynines

DevOps Sync



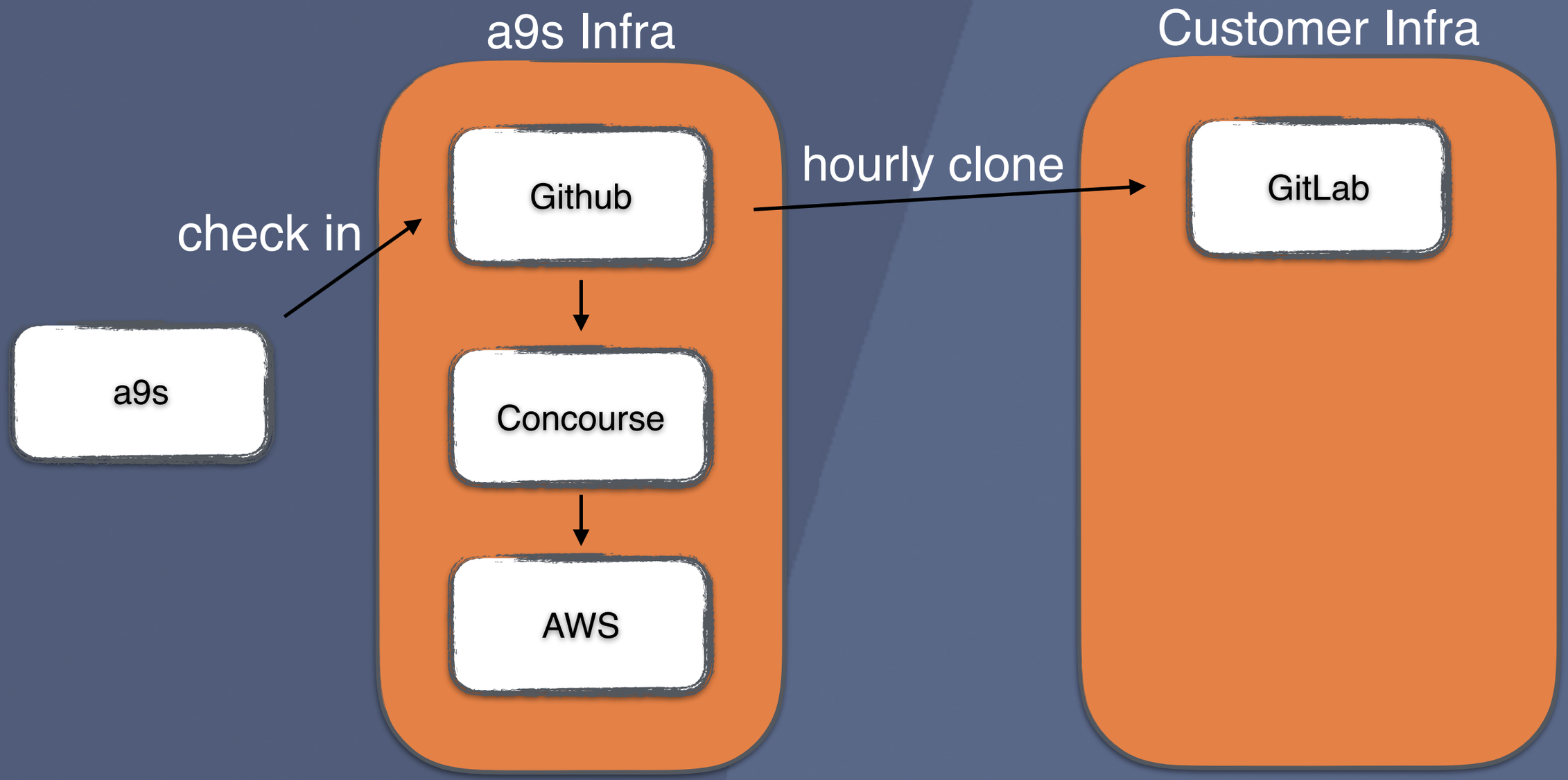
anynines

DevOps Sync



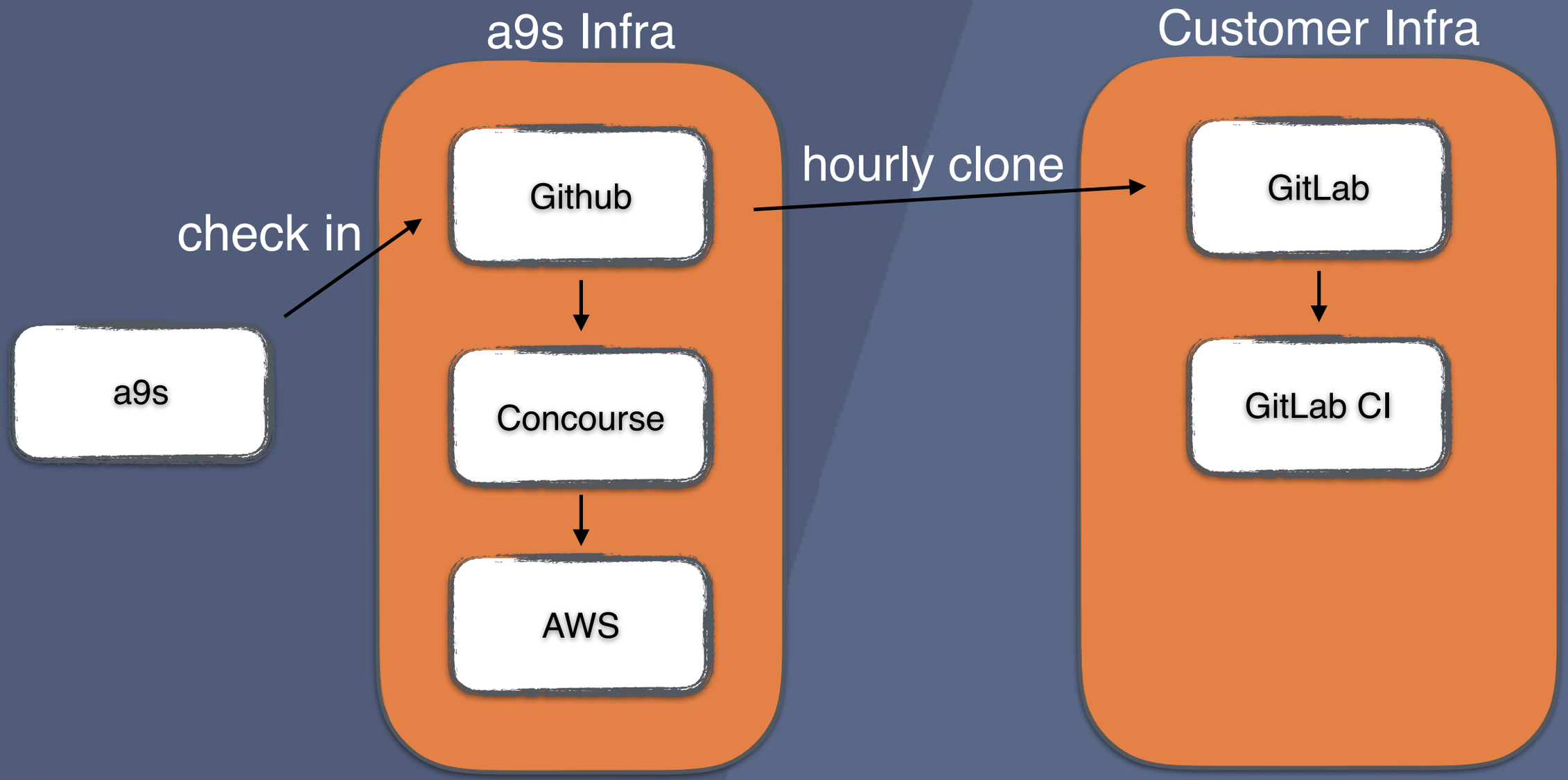
anynines

DevOps Sync



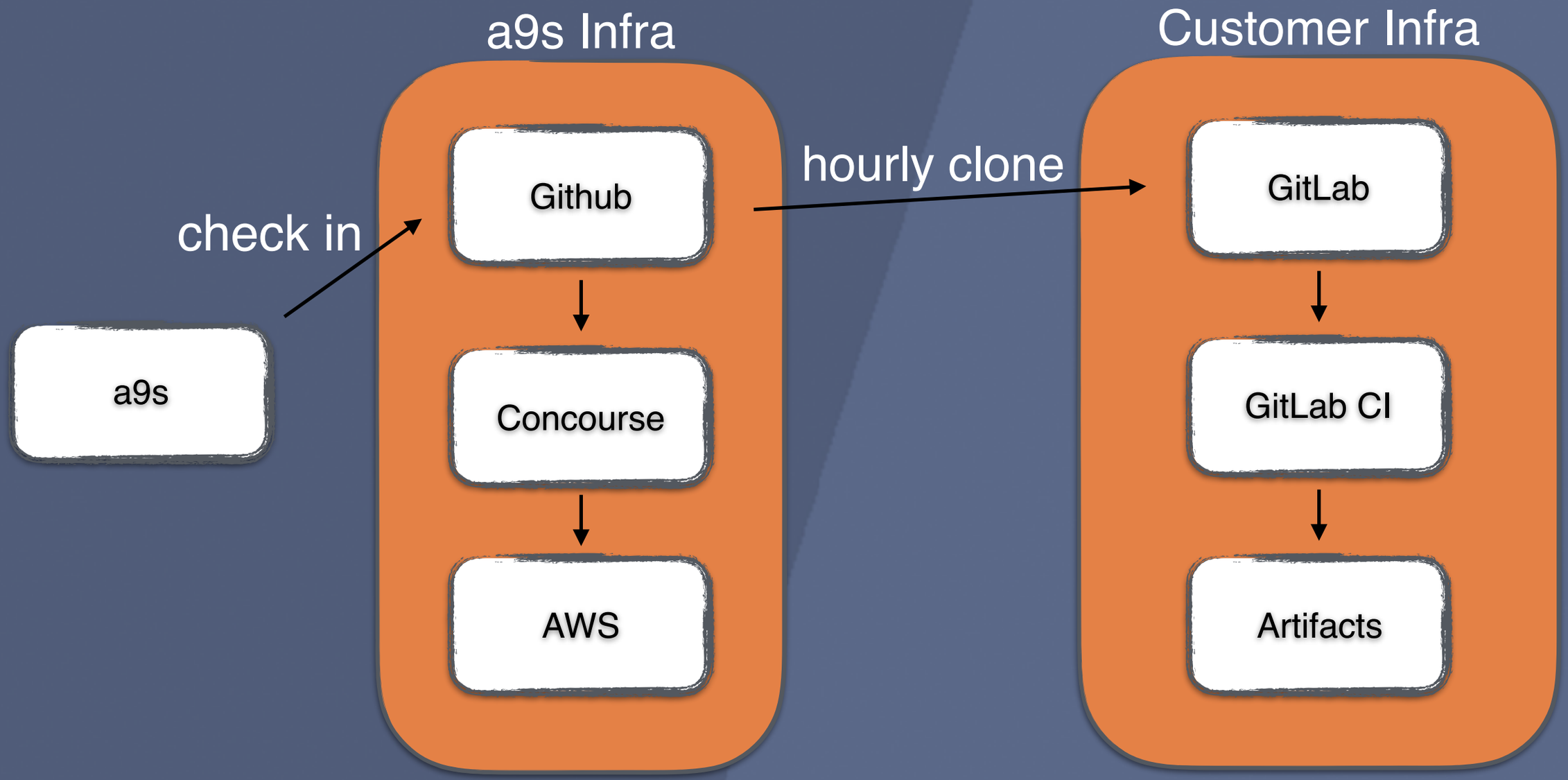
anynines

DevOps Sync



anynines

DevOps Sync



anynines

DevOps Sync

Customer Infra

Dev

Artifacts

Integration

GitLab

anynines

DevOps Sync

Customer Infra

Dev

Integration

Artifacts

GitLab

anynines

DevOps Sync

Customer Infra

Integration

Artifacts

Deployment Train

Production

Infrastructure Engineer

anynines

DevOps Sync

Customer Infra

Integration

Production

Artifacts

Deployment Train

Infrastructure Engineer

anynines

Operational Knowledge Diverged

anynines

Operational Knowledge Diverged

- There was a dedicated repo on customer side for operation scripts

anynines

Operational Knowledge Diverged

- There was a dedicated repo on customer side for operation scripts
- Blue Green Deployment was very complex

anynines

Operational Knowledge Diverged

- There was a dedicated repo on customer side for operation scripts
- Blue Green Deployment was very complex
- At the end of the project knowledge sharing didn’t go very well (deadlines)

anynines

Operational Knowledge Diverged

- There was a dedicated repo on customer side for operation scripts
- Blue Green Deployment was very complex
- At the end of the project knowledge sharing didn’t go very well (deadlines)
  - -> Standups with Infrastructure Engineer

anynines

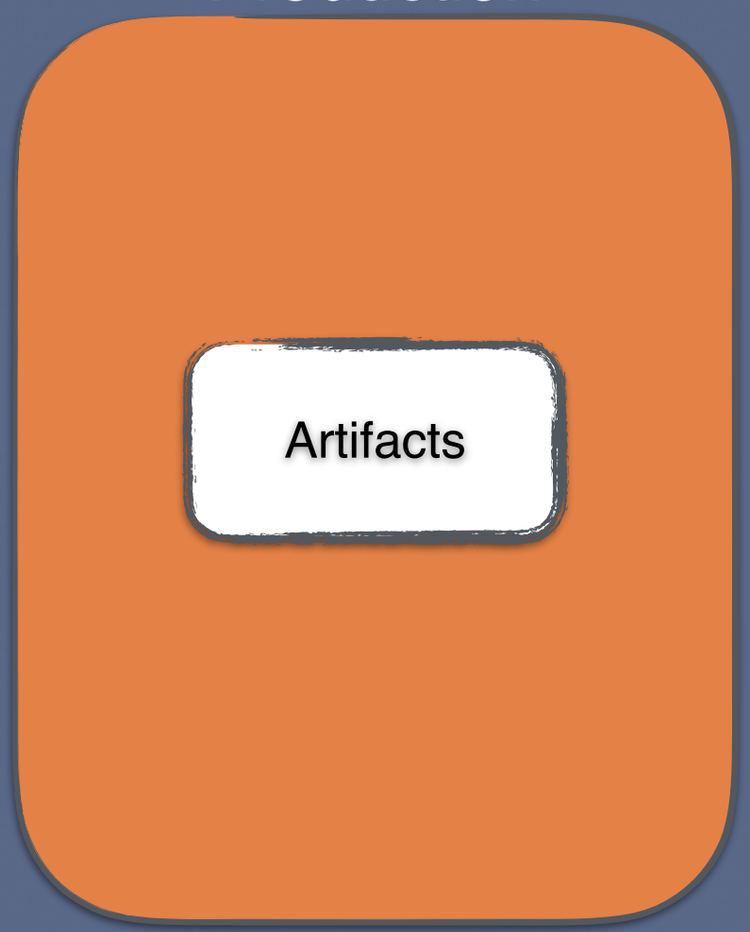
Operational Knowledge Diverged

- There was a dedicated repo on customer side for operation scripts
- Blue Green Deployment was very complex
- At the end of the project knowledge sharing didn’t go very well (deadlines)
  - -> Standups with Infrastructure Engineer
  - -> Ops Showcase to sync ops knowledge

anynines

DevOps Sync - Monitoring

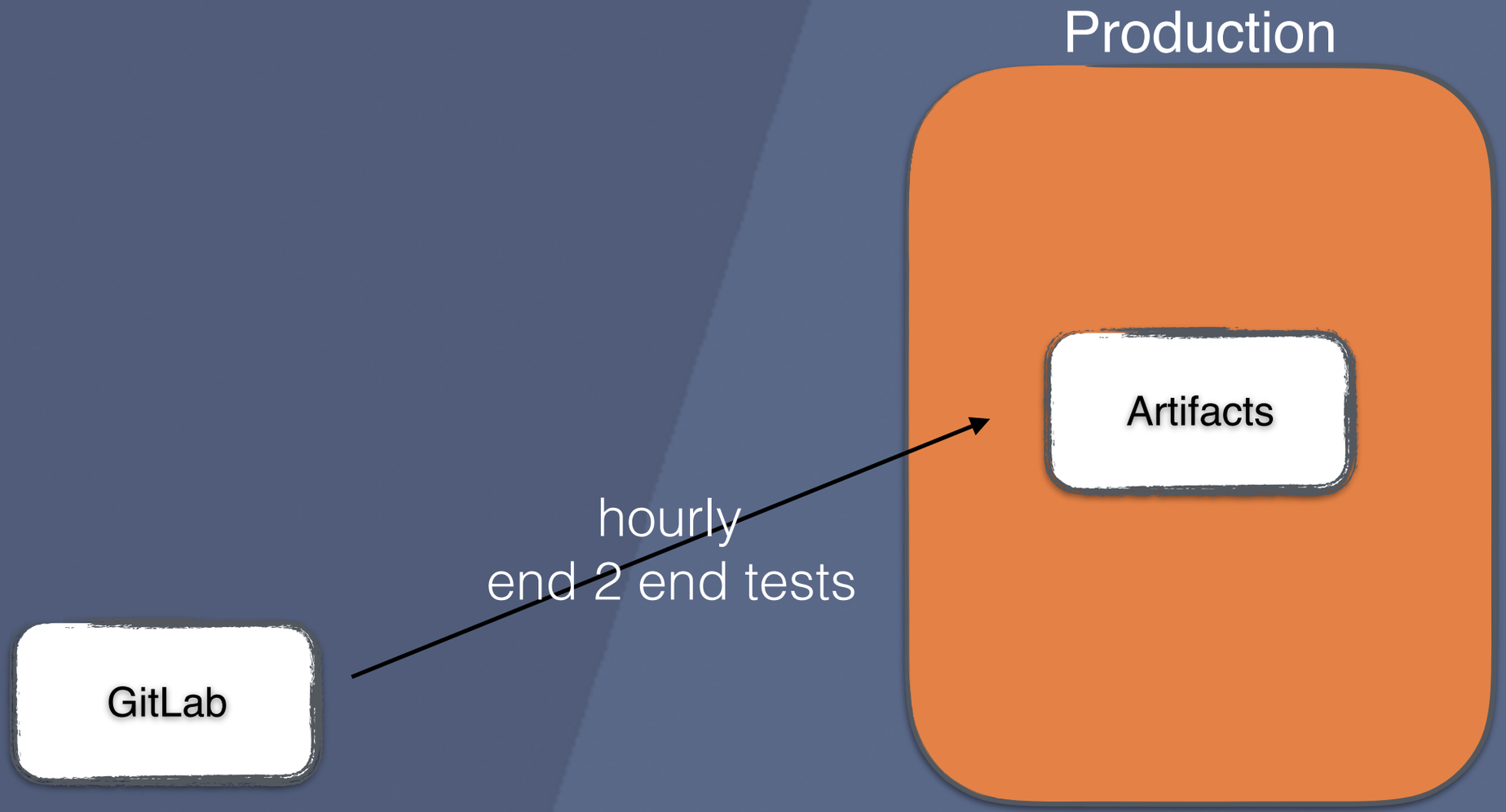
Customer Infra



anynines

DevOps Sync - Monitoring

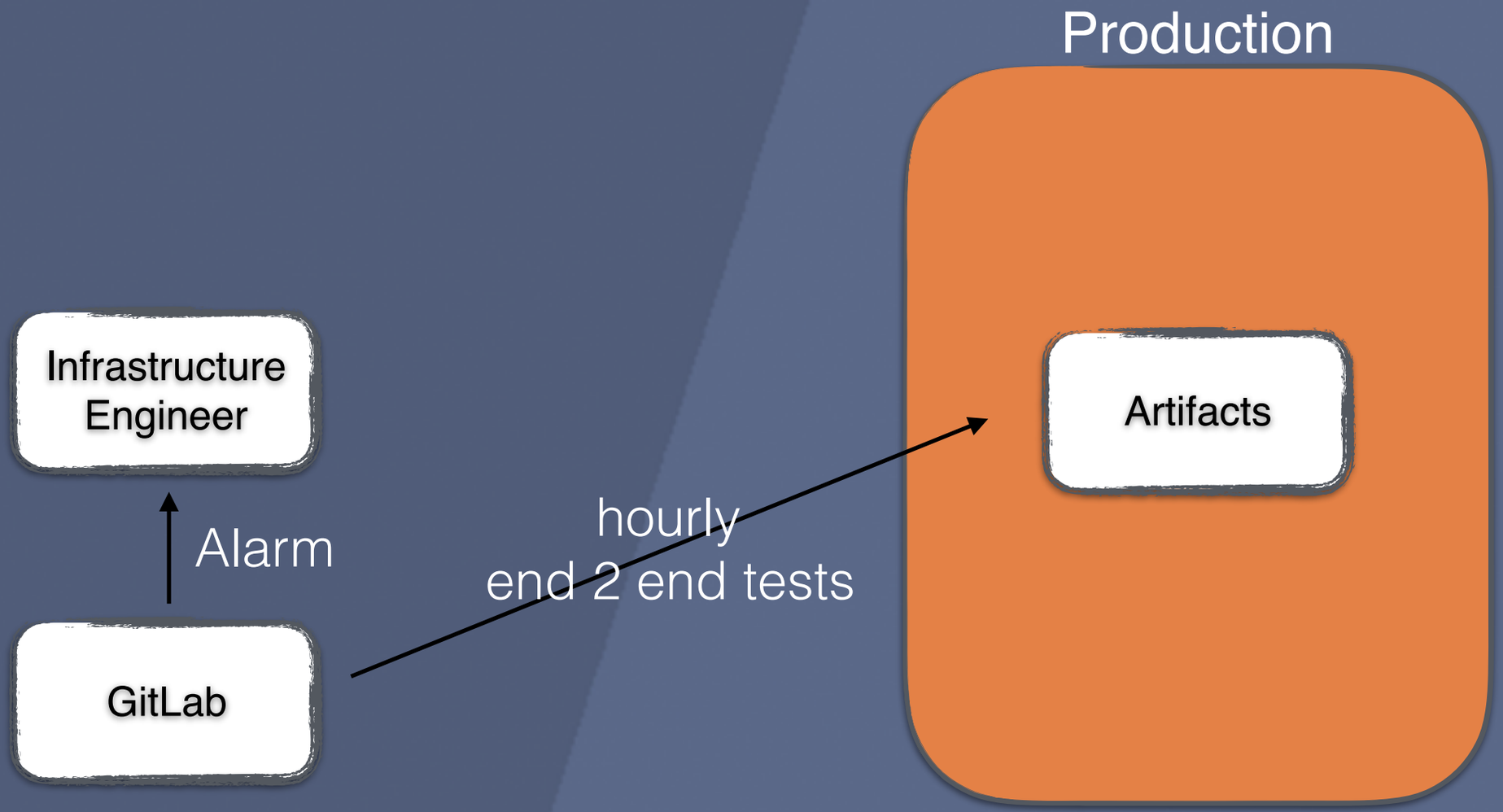
Customer Infra



anynines

DevOps Sync - Monitoring

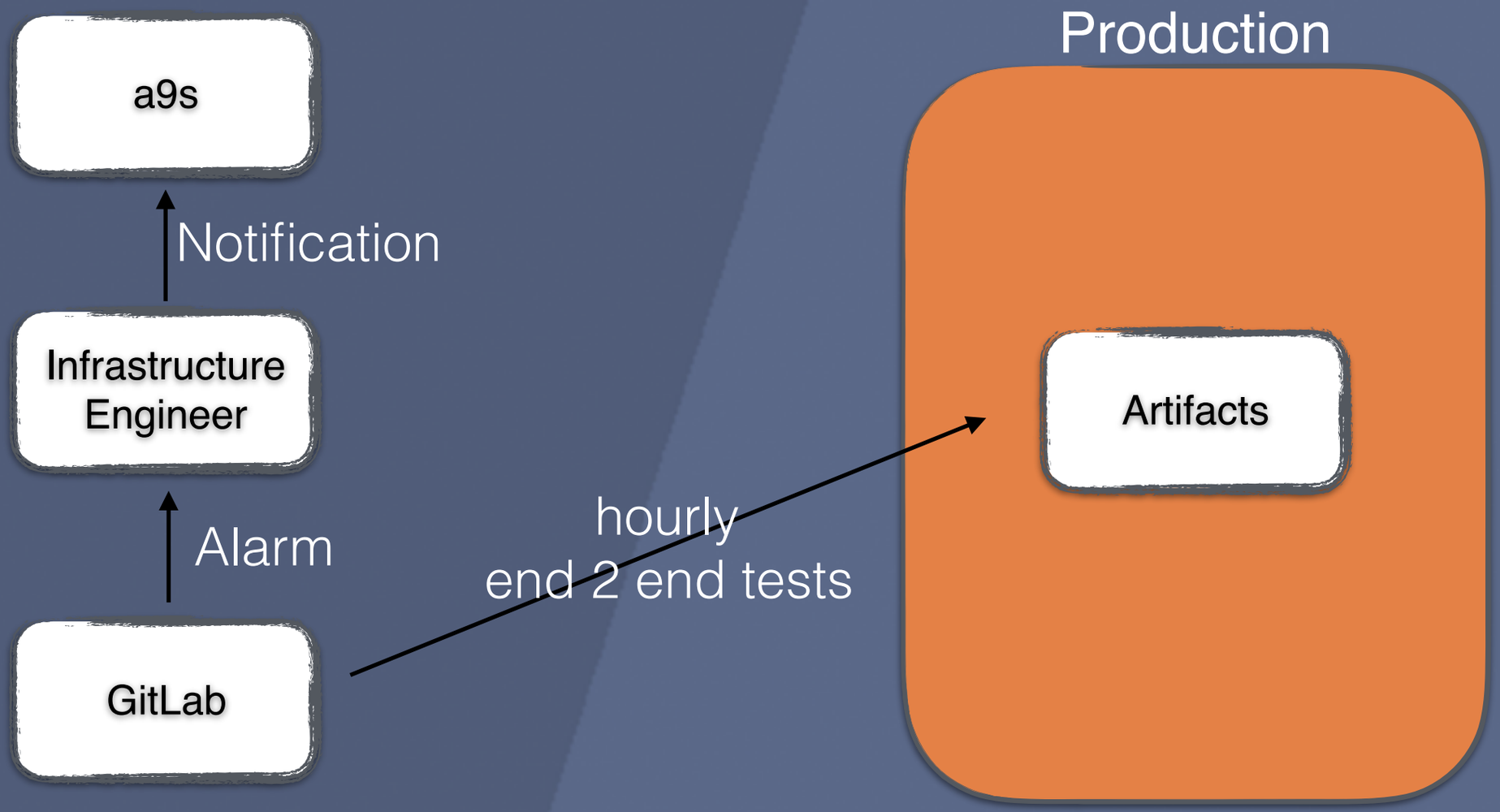
Customer Infra



anynines

DevOps Sync - Monitoring

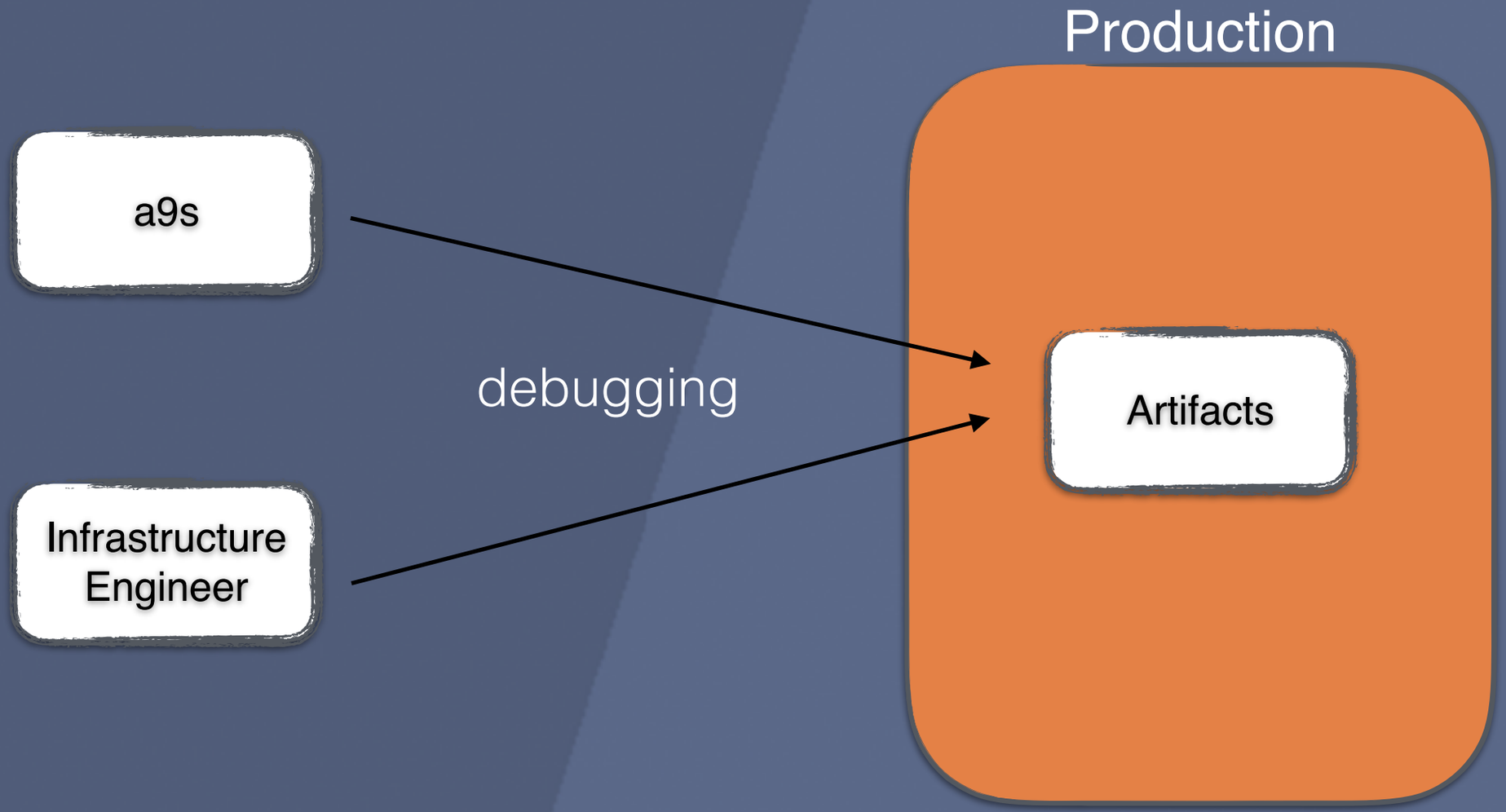
Customer Infra



anynines

DevOps Sync - Monitoring

Customer Infra



anynines

“

”

anynines

“Responding to change over following a plan”

anynines

Recap

anynines

Recap

- Complete transformation won’t happen

anynines

Recap

- Complete transformation won’t happen
- Still a good mindset base

anynines

Recap

- Complete transformation won’t happen
- Still a good mindset base
- We knew, we won’t break down silo completely

anynines

Recap

- Complete transformation won’t happen
- Still a good mindset base
- We knew, we won’t break down silo completely
- But we build some good bridges

anynines

Lessons Learned / Advices

Lessons Learned / Advices

- Don’t underestimate Conway’s Law

anynines

Lessons Learned / Advices

- Don’t underestimate Conway’s Law
- C level support for DevOps transformation is a must

anynines

Lessons Learned / Advices

- Don’t underestimate Conway’s Law
- C level support for DevOps transformation is a must
- Agile software development supports DevOps culture

anynines

Lessons Learned / Advices

- Don’t underestimate Conway’s Law
- C level support for DevOps transformation is a must
- Agile software development supports DevOps culture
- DevOps is a mindset not a job title

anynines

Lessons Learned / Advices

- Don’t underestimate Conway’s Law
- C level support for DevOps transformation is a must
- Agile software development supports DevOps culture
- DevOps is a mindset not a job title
- Don’t rush the DevOps transformation

anynines

Lessons Learned / Advices

- Don’t underestimate Conway’s Law
- C level support for DevOps transformation is a must
- Agile software development supports DevOps culture
- DevOps is a mindset not a job title
- Don’t rush the DevOps transformation
- Don’t force DevOps on your team, instead try to establish a mindset and a transformation from inside out, not from outside in

anynines

Lessons Learned / Advices

- Don’t underestimate Conway’s Law
- C level support for DevOps transformation is a must
- Agile software development supports DevOps culture
- DevOps is a mindset not a job title
- Don’t rush the DevOps transformation
- Don’t force DevOps on your team, instead try to establish a mindset and a transformation from inside out, not from outside in
- Big Bang transformation won’t gonna happen

anynines

Lessons Learned/Advices

Lessons Learned/Advices

- If you are constraint sync as much as possible

anynines

Lessons Learned/Advices

- If you are constraint sync as much as possible
- Knowledge transfer

anynines

Lessons Learned/Advices

- If you are constraint sync as much as possible
- Knowledge transfer
- Try to transform yourself to adapt to customer instead of trying to transform the customers mindset and constraint

anynines

Lessons Learned/Advices

- If you are constraint sync as much as possible
- Knowledge transfer
- Try to transform yourself to adapt to customer instead of trying to transform the customers mindset and constraint
- If you cannot break down silos, build bridges

anynines

Lessons Learned/Advices

- If you are constraint sync as much as possible
- Knowledge transfer
- Try to transform yourself to adapt to customer instead of trying to transform the customers mindset and constraint
- If you cannot break down silos, build bridges
- If you have to pass the software to ops, make sure that you provide as much as possible

anynines

Lessons Learned/Advices

- If you are constraint sync as much as possible
- Knowledge transfer
- Try to transform yourself to adapt to customer instead of trying to transform the customers mindset and constraint
- If you cannot break down silos, build bridges
- If you have to pass the software to ops, make sure that you provide as much as possible
- Try to have dev/prod parity

anynines

Lessons Learned/Advices

- If you are constraint sync as much as possible
- Knowledge transfer
- Try to transform yourself to adapt to customer instead of trying to transform the customers mindset and constraint
- If you cannot break down silos, build bridges
- If you have to pass the software to ops, make sure that you provide as much as possible
- Try to have dev/prod parity
- Try to get System Access as soon as possible

anynines

We are humans

anynines

We are humans

- Ego

anynines

We are humans

- Ego
- Comfort zone

anynines

We are humans

- Ego
- Comfort zone
- Fear of Change

anynines

We are humans

- Ego
- Comfort zone
- Fear of Change
- Fear of loosing things

anynines

We are humans

- Ego
- Comfort zone
- Fear of Change
- Fear of loosing things
- Feeling offended

anynines

We are humans

- Ego
- Comfort zone
- Fear of Change
- Fear of loosing things
- Feeling offended
- …

anynines

“

”

anynines

“Don’t hire people for what they can do for you today, hire them for what they can do for you tomorrow”

anynines

“

”

anynines

“No innovation without diversity. No improvement without change.”

Thank You

anynines

Questions

anynines