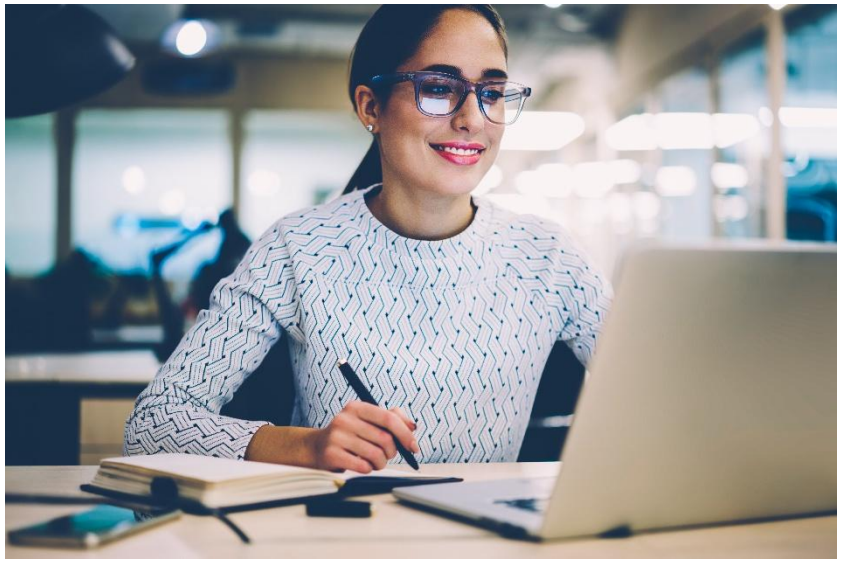
Infrastructure Automation Tools

TEKsystems Global Services

WORKFORCE DEVELOPMENT

CI/CD and Infra Automation

Learning Objectives



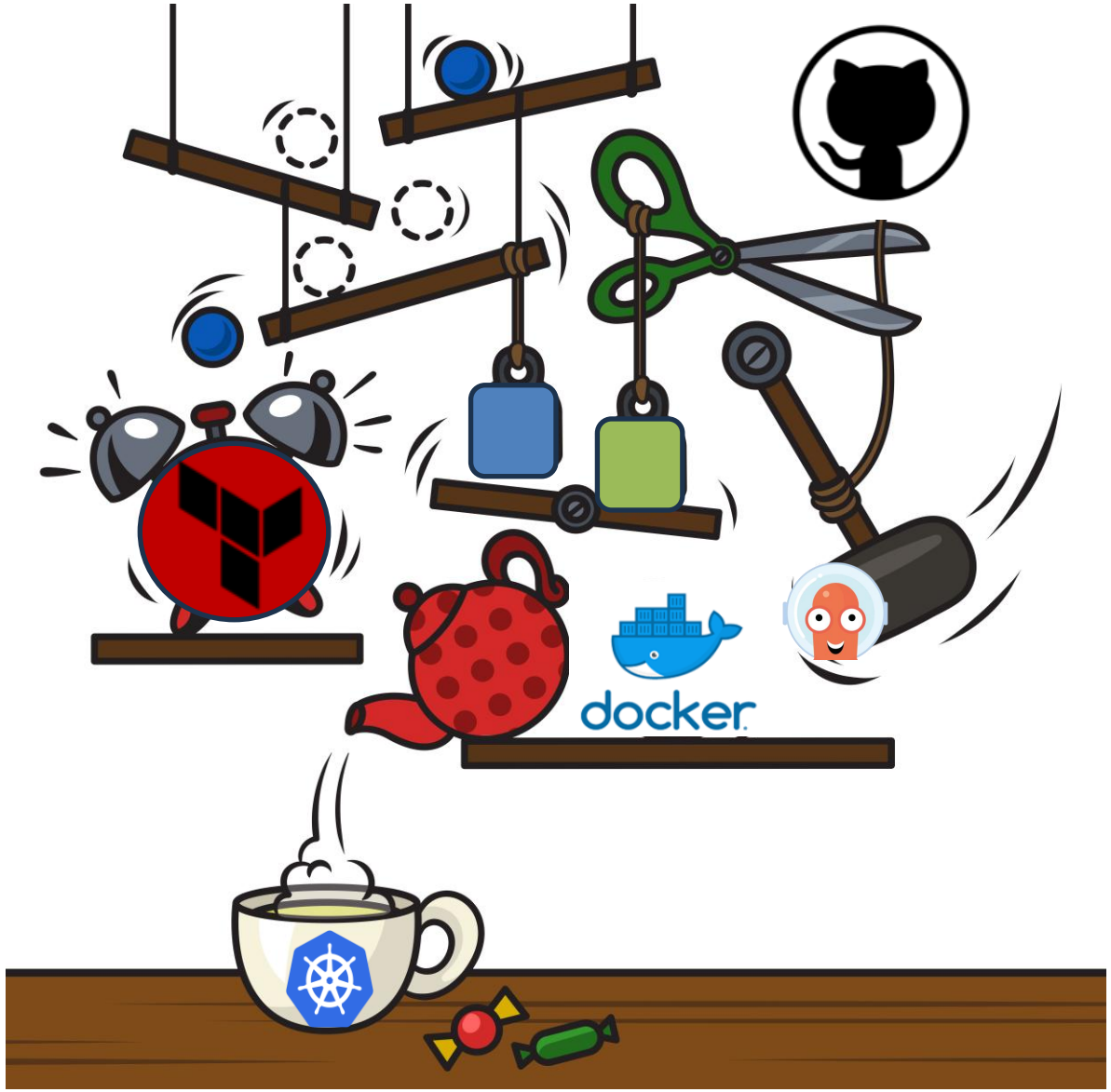
Day 5 — Advanced Deployment and Resiliency

Today you'll dive deeper into **deployment strategies**, with an emphasis on **Blue/Green** and controlled rollouts. You'll also expand into **Kubernetes extensions** and **data center resiliency concepts** used in real production systems.

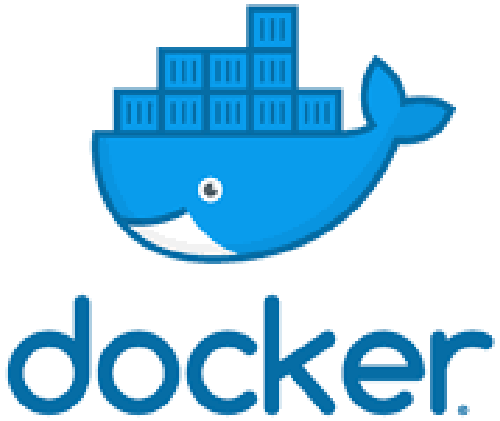
- More on deployment strategies (Blue/Green, Canary, Rollouts)
- Kubernetes CRDs and **Argo Rollouts**
- Data center resiliency concepts: **RTO, RPO, geo-redundancy**
- Data synchronization and Infrastructure as Code patterns

These concepts tie application delivery to real-world reliability and disaster recovery planning.

CI CD



Automation tools



kubernetes

Resiliency

Release Strategies

Release Strategies

Strategy	Description	Benefits	Downsides	Rollback Process / Risk
Recreate	Old version is stopped before the new one starts	Very simple, no version overlap	Downtime, high risk in prod	**High risk** – rollback requires redeploying and causes more downtime
Rolling Update	Gradually replaces old instances with new ones	No full downtime, efficient resource use	Mixed versions running simultaneously	**Moderate risk** – rollback requires another rollout
Blue/Green	Two identical environments; traffic switches when ready	Instant rollback, very safe	Higher infrastructure cost	**Low risk** – switch traffic back immediately
Canary	New version released to a small % of users first	Low blast radius, early issue detection	Requires advanced monitoring	**Low risk** – stop or reduce canary traffic
A/B Testing	Traffic split intentionally to compare versions	Data-driven decisions, UX insights	Operational complexity	**Medium risk** – depends on routing and experiment setup

Release Plan Checklist

A **release plan** ensures changes are deployed safely and can be recovered if needed.

Checklist:

- Version tagged and tested
- Database changes reviewed and backed up
- Release strategy chosen
- Monitoring and alerts active
- Rollback plan confirmed
- Stakeholders notified / CAB approved
- Deployment scheduled outside blackout periods

Good release plans reduce risk by making deployments deliberate and controlled.

Major Database Updates

Some upgrades change the database schema in ways older versions can't understand, so rolling back the application alone is not enough.

- Major upgrades often run migrations that make the DB **incompatible** with the old version
- Rollback typically requires a **backup/restore** of the database, not just redeploying old code
- To avoid data corruption, teams often use a **maintenance window** or read-only mode
- Companies try to avoid releasing updates that are not backwards compatible to avoid this risks

If rollback is required, the database is restored to the pre-upgrade backup, and any data written during the upgrade window may need reconciliation or may be lost depending on the recovery plan.

Rolling Update (K8s Deployment)

Kubernetes Deployments support **basic release strategies** out of the box, without requiring any extensions or CRDs.

- A **Deployment** supports:
  - **Recreate** strategy
  - **RollingUpdate** strategy (default)
- Rolling updates gradually replace Pods based on configured limits
- No native support for Blue/Green or Canary without additional tooling

Apply or Update a Deployment

Apply or update a Deployment:
kubectl apply -f deployment.yaml

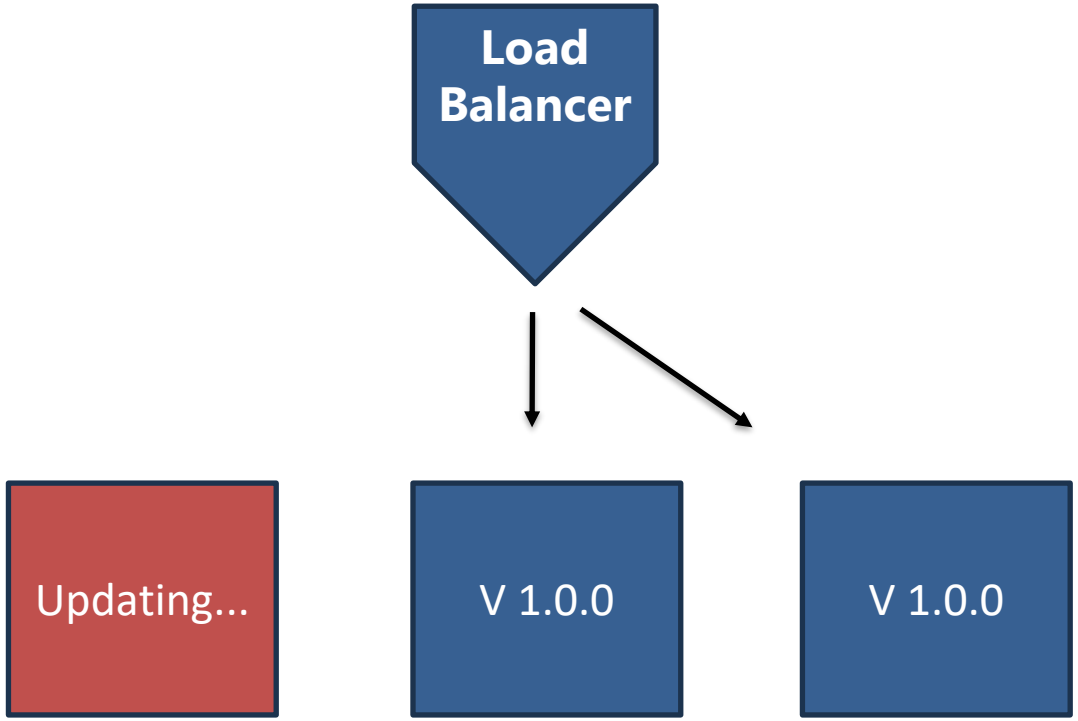
View rollout status:
kubectl rollout status deployment/my-app

View rollout history:
kubectl rollout history deployment/my-app

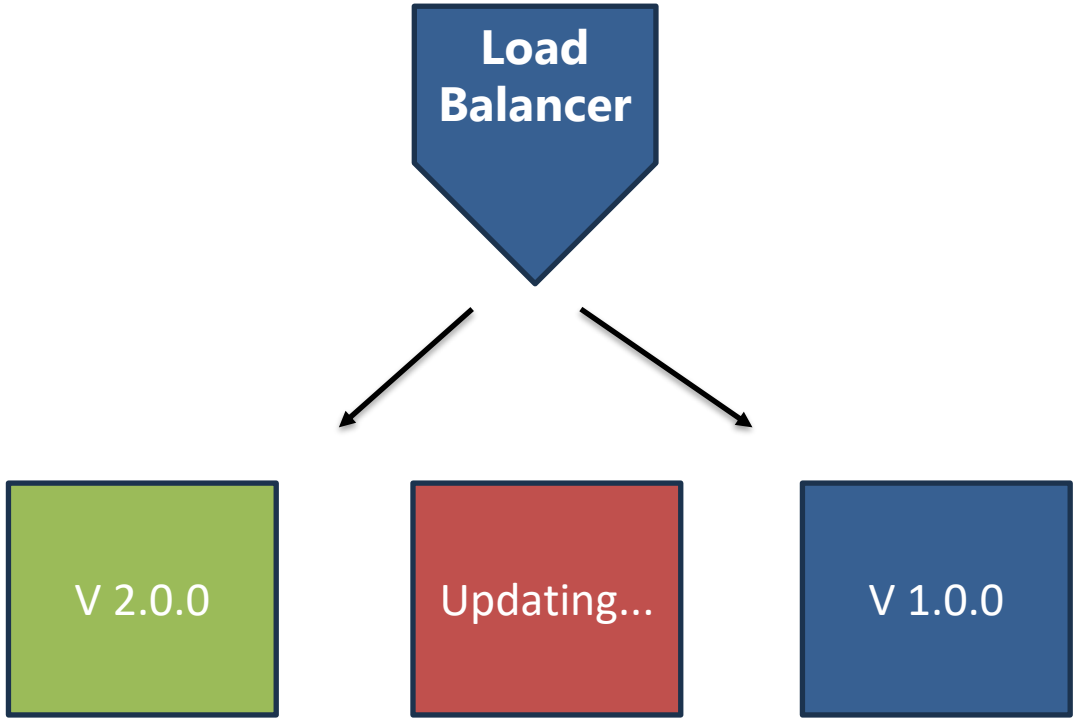
Roll back to previous version:
kubectl rollout undo deployment/my-app

Native Deployments are simple and reliable, but more advanced release strategies require tools like **Argo Rollouts**.

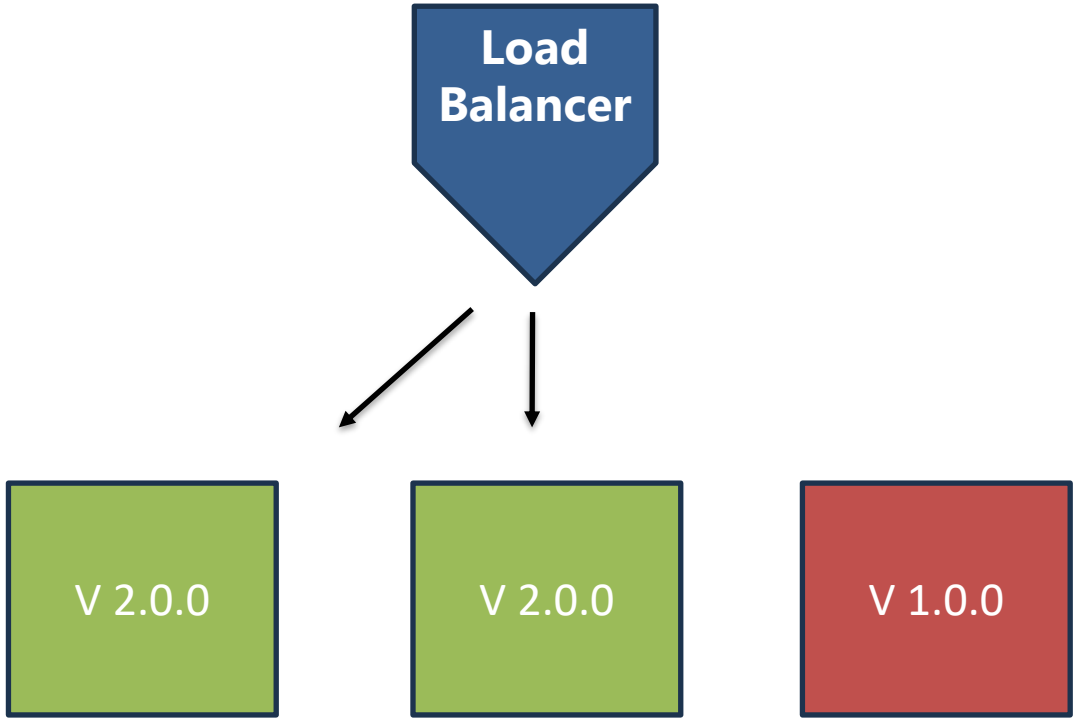
Rolling Update



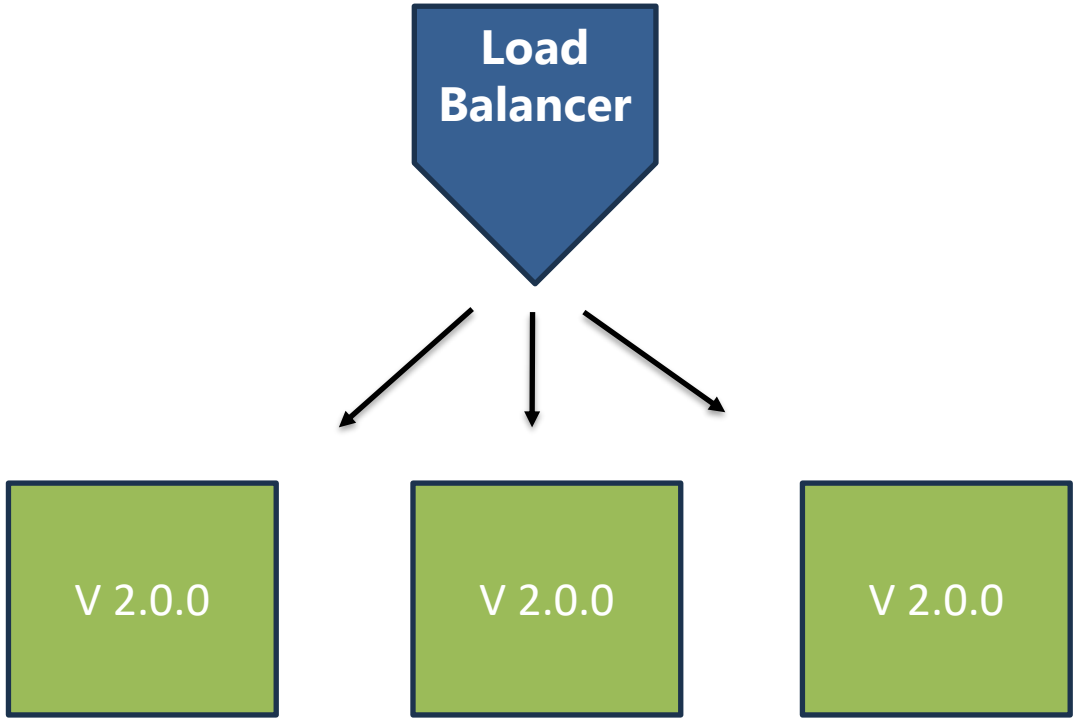
Rolling Update



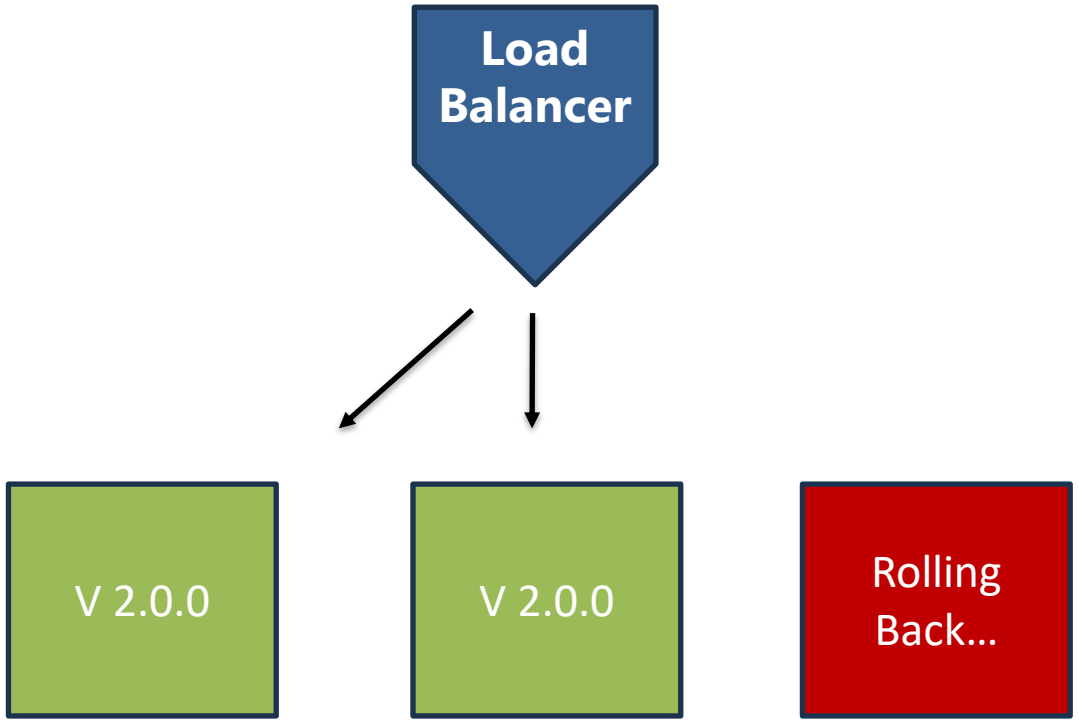
Rolling Update



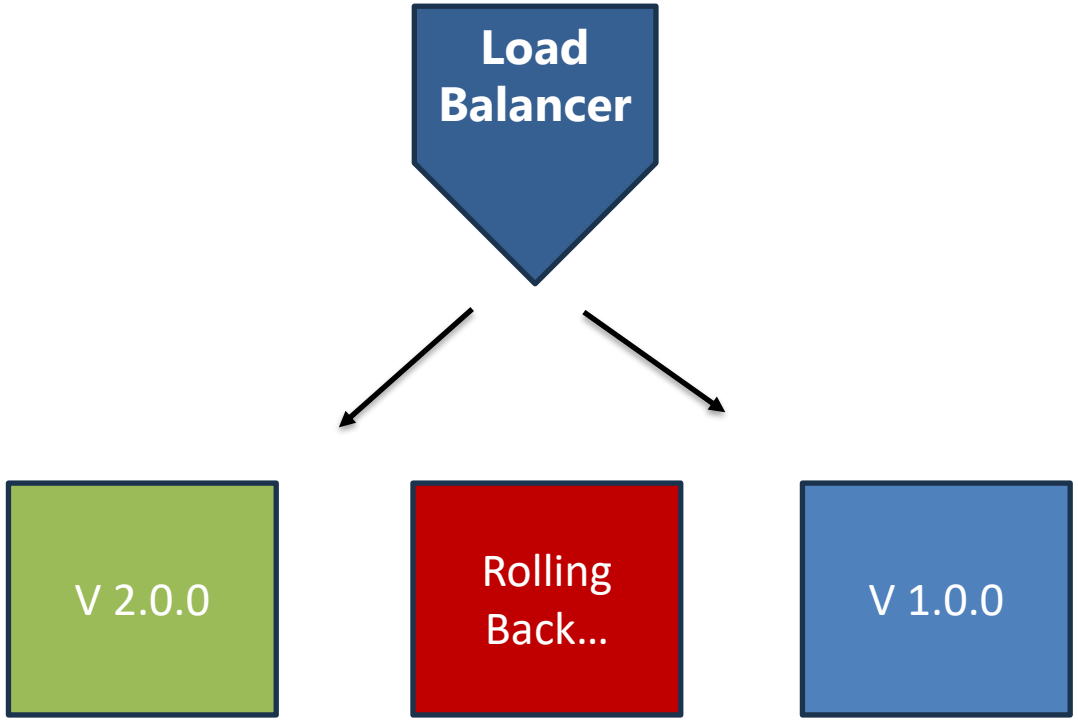
Rolling Update



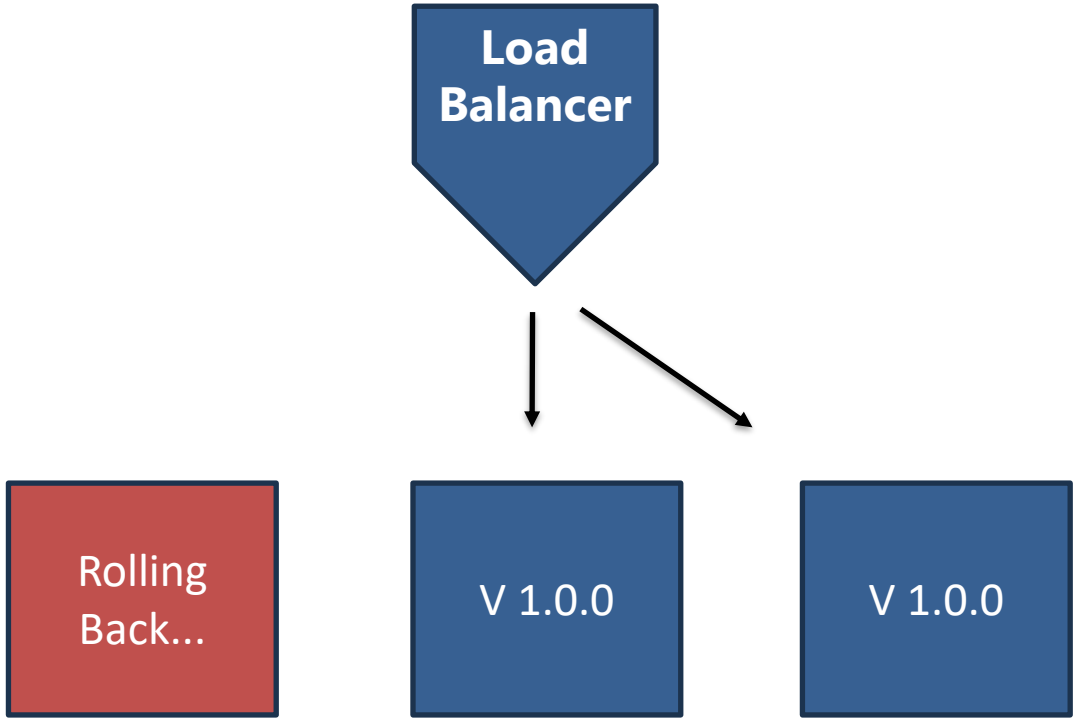
Rolling Rollback



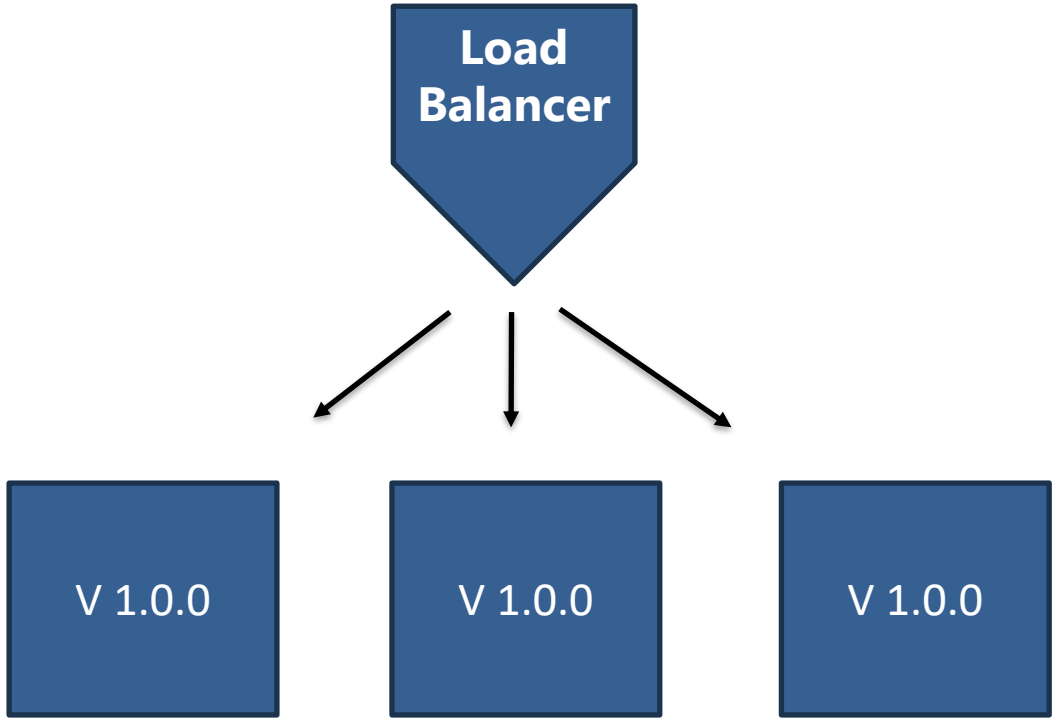
Rolling Rollback



Rolling Rollback



Rolling Rollback



Testing Rolling Update vs Blue Green

Rolling updates are harder to test because **live traffic reaches the new version immediately** as it is deployed.

- **Rolling Update:** New version receives traffic as Pods are replaced
- Issues may impact users before they are detected
- Testing is limited to live monitoring and alerts

With **Blue/Green**, the Green environment can be fully deployed and tested **with zero live traffic**. Only after validation does traffic switch, making Blue/Green safer and easier to verify before users are affected.

Blue Green Deployment With Argo CD

Today we will be performing a Blue Green deployment using Argo CD

Argo Rollout CRD

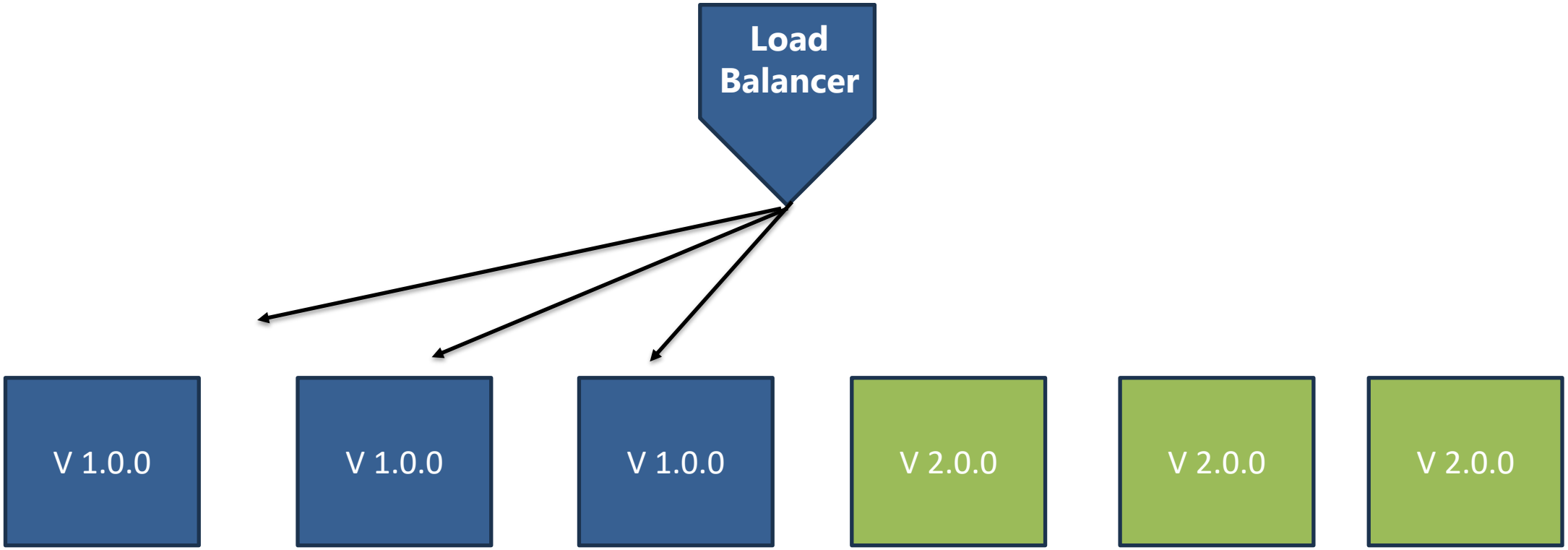
```yaml
apiVersion: argoproj.io/v1alpha1
kind: Rollout
metadata:
  name: example-application
spec:
  replicas: 2
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      app: example-application
  template:
    metadata:
      labels:
        app: example-application
    spec:
      containers:
      - name: example-application
        image: docker.io/YOUR_USERNAME/
        example-application:v1.0.0
        ports:
        - containerPort: 8080
```

This manifest defines a **Rollout**, which is **not a built-in Kubernetes object** and must be installed as **a Custom Resource Definition (CRD)**.

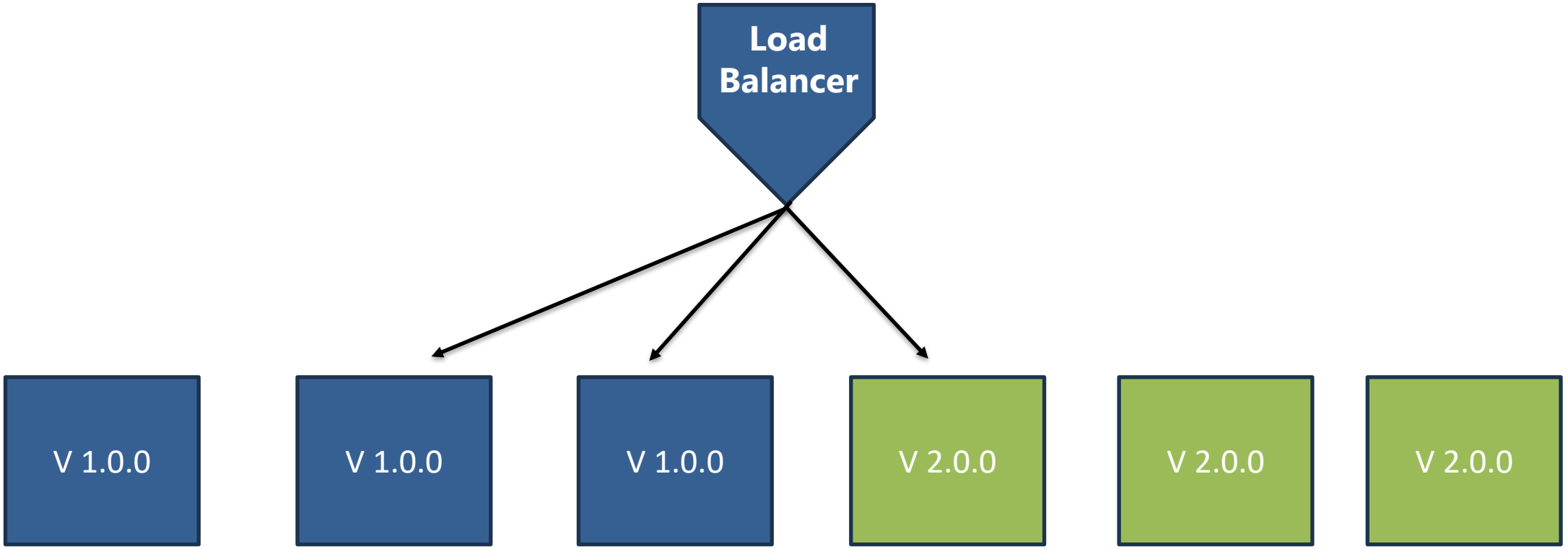
- **kind: Rollout** is not part of standard Kubernetes
- It exists only after installing the **Argo Rollouts CRD**
- **apiVersion** points to the API group introduced by that CRD
- The Rollout replaces a standard Deployment for release control

A Rollout still manages **replicas of Pods** (via ReplicaSets), but it adds advanced release logic such as Blue/Green, Canary, pauses, and analysis steps. Kubernetes handles the Pods; the Rollout controller handles *how* they are released.

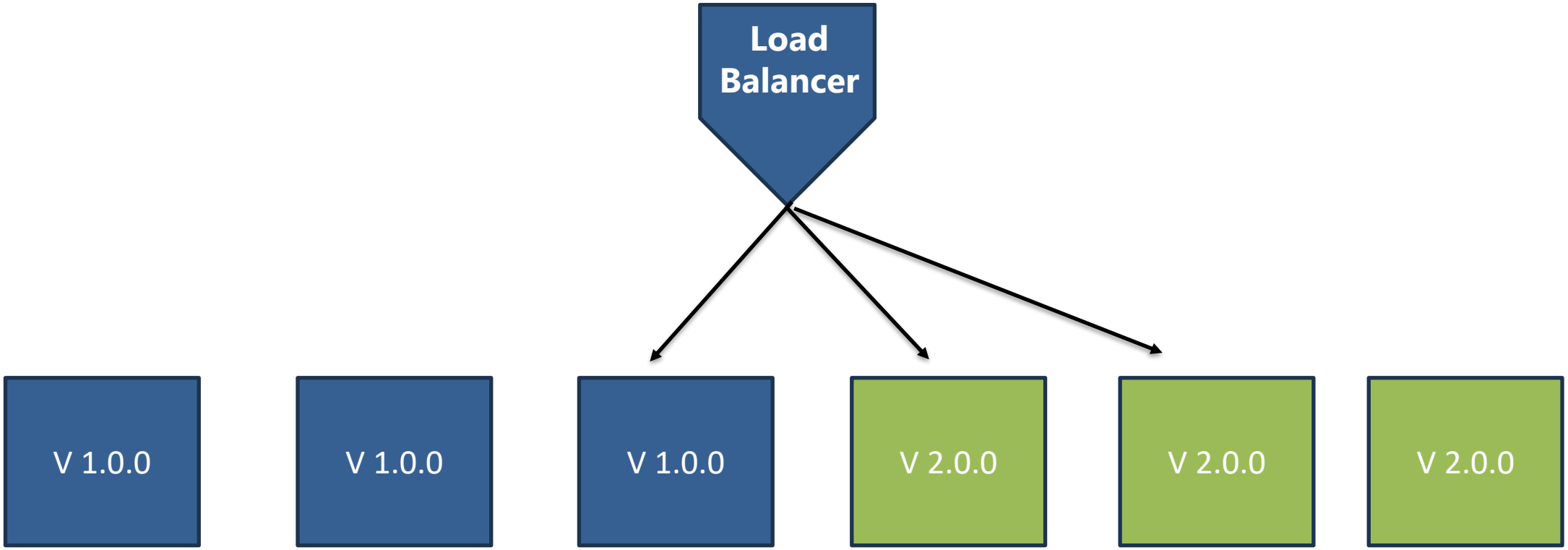
Blue/Green Deployment



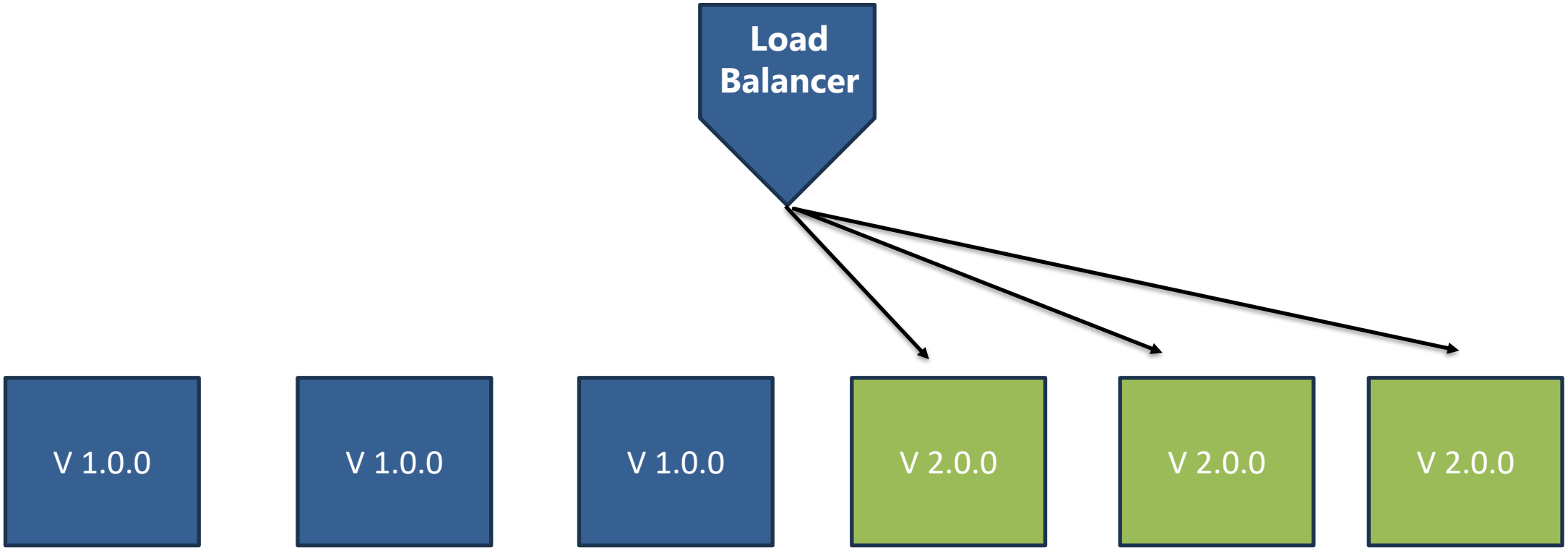
Blue/Green Deployment



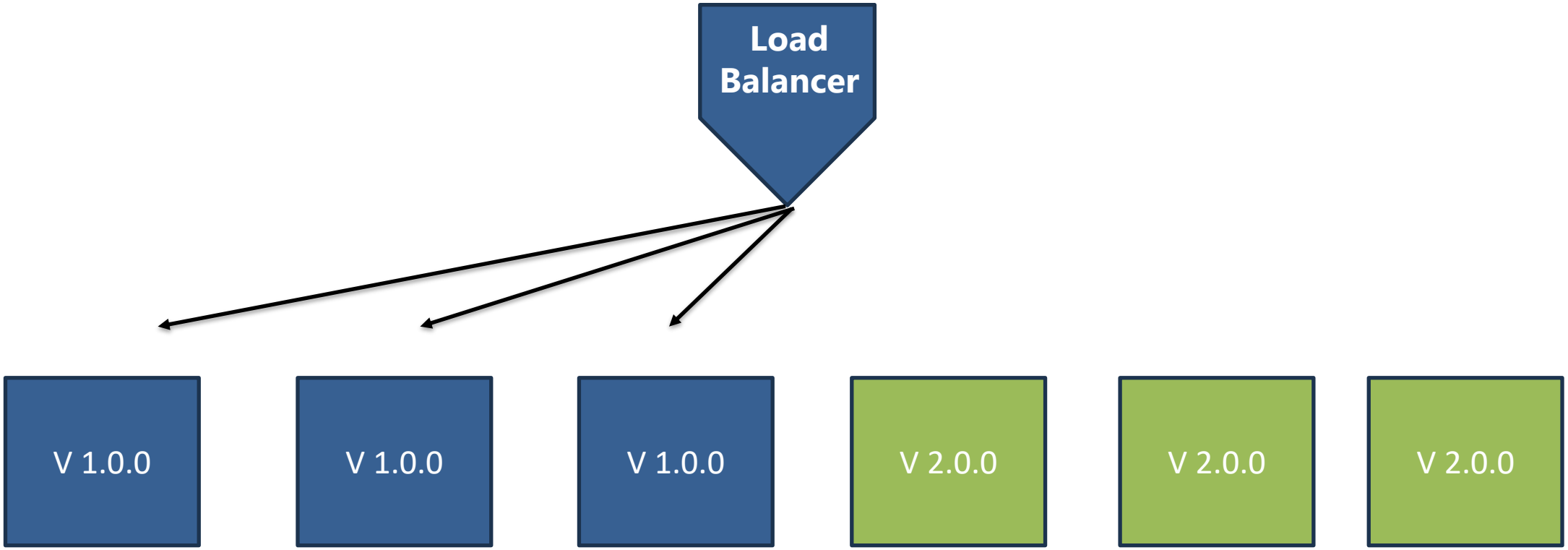
Blue/Green Deployment



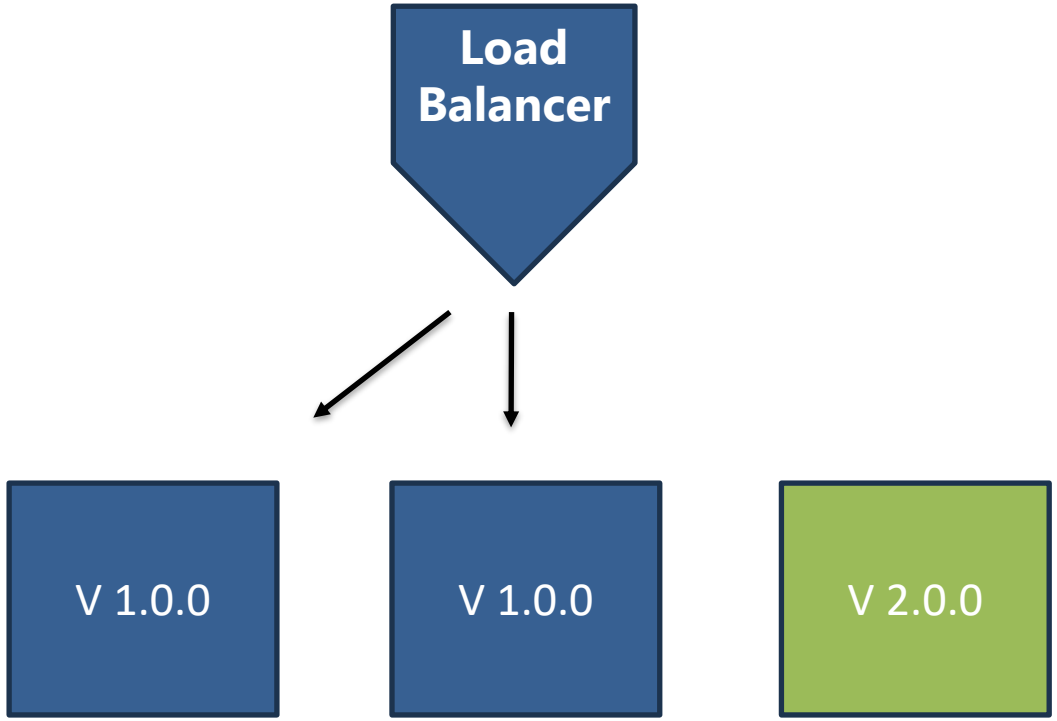
Blue/Green Deployment



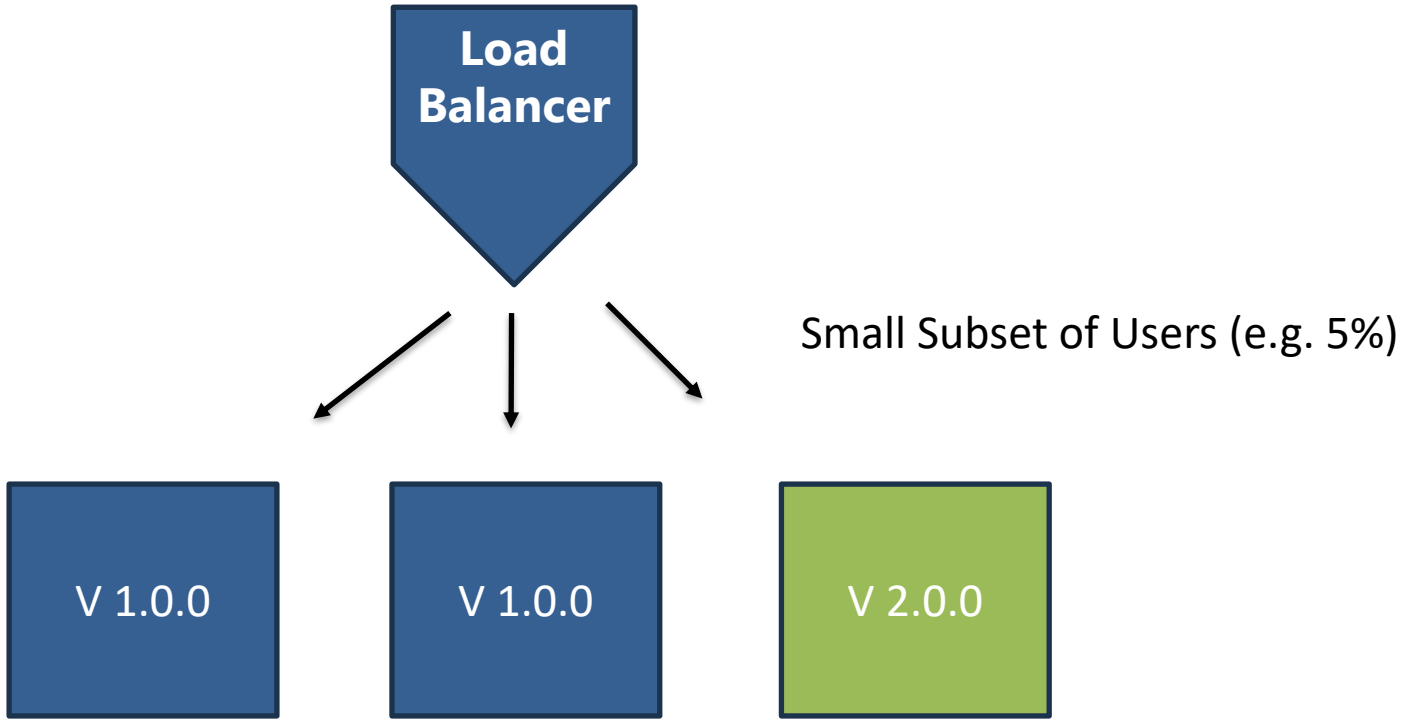
Blue/Green Rollback



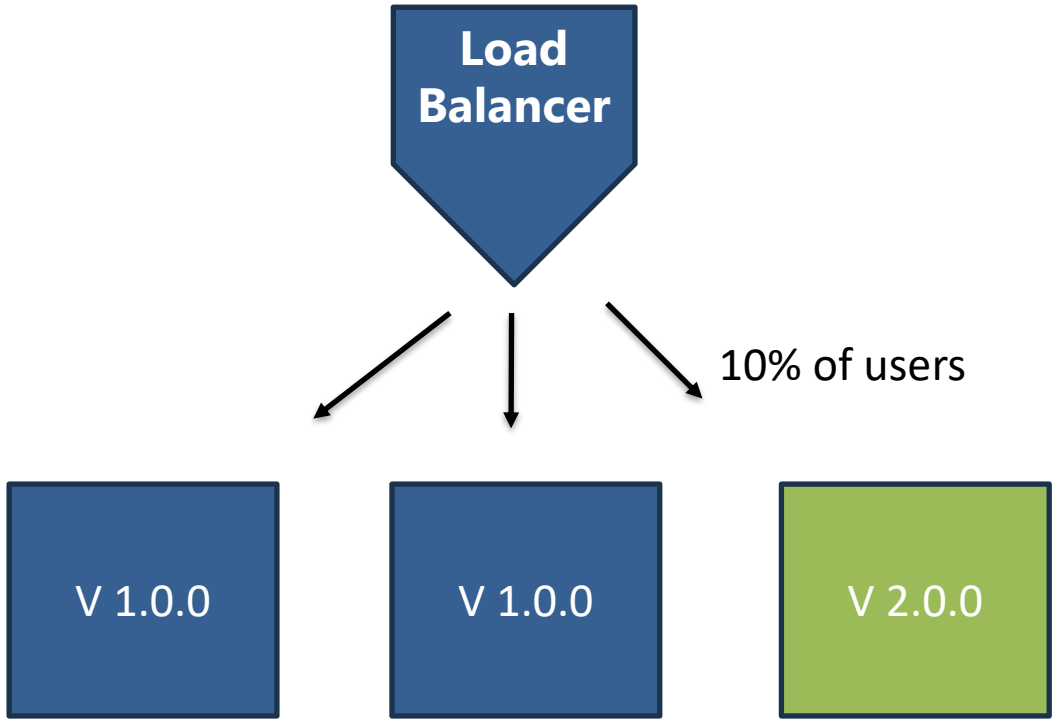
Canary



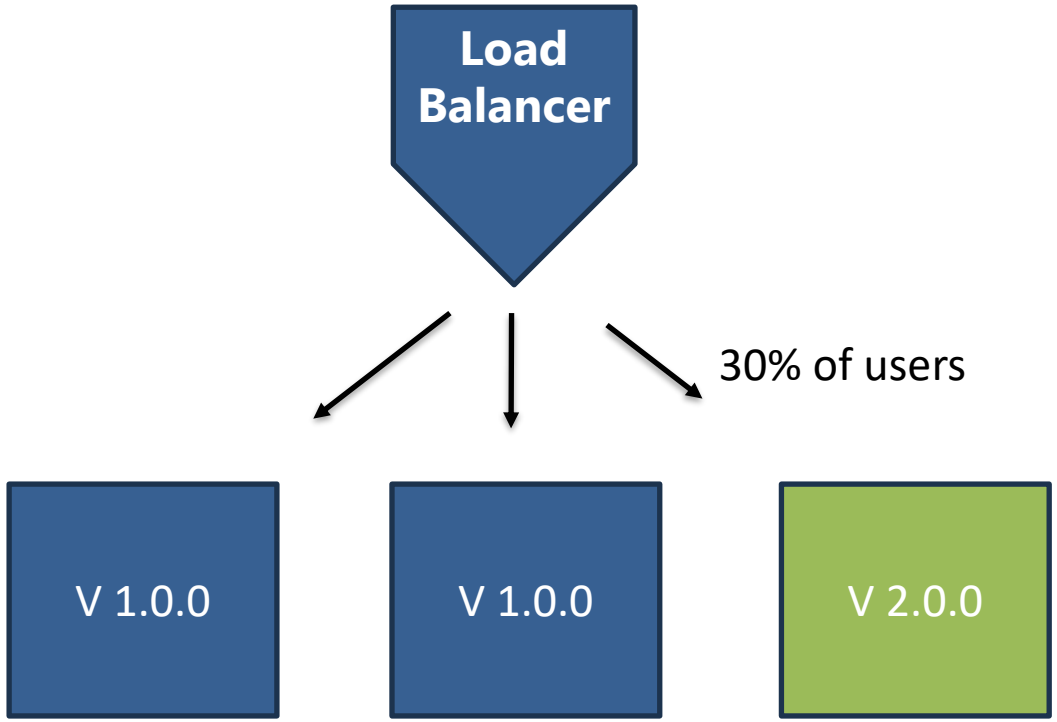
Canary



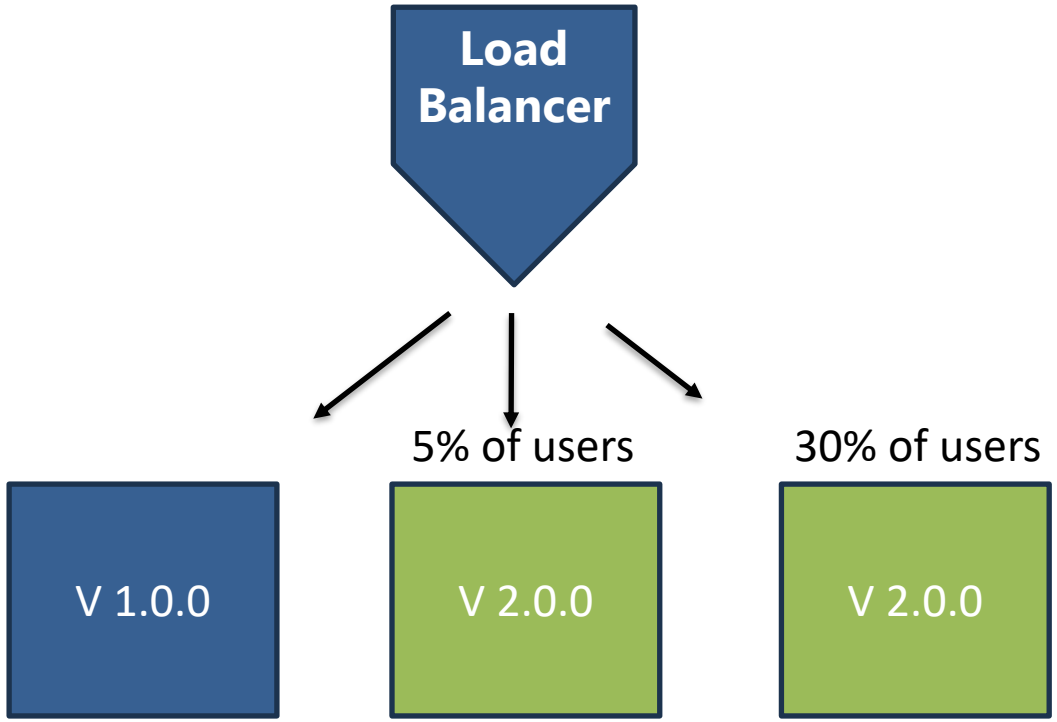
Canary



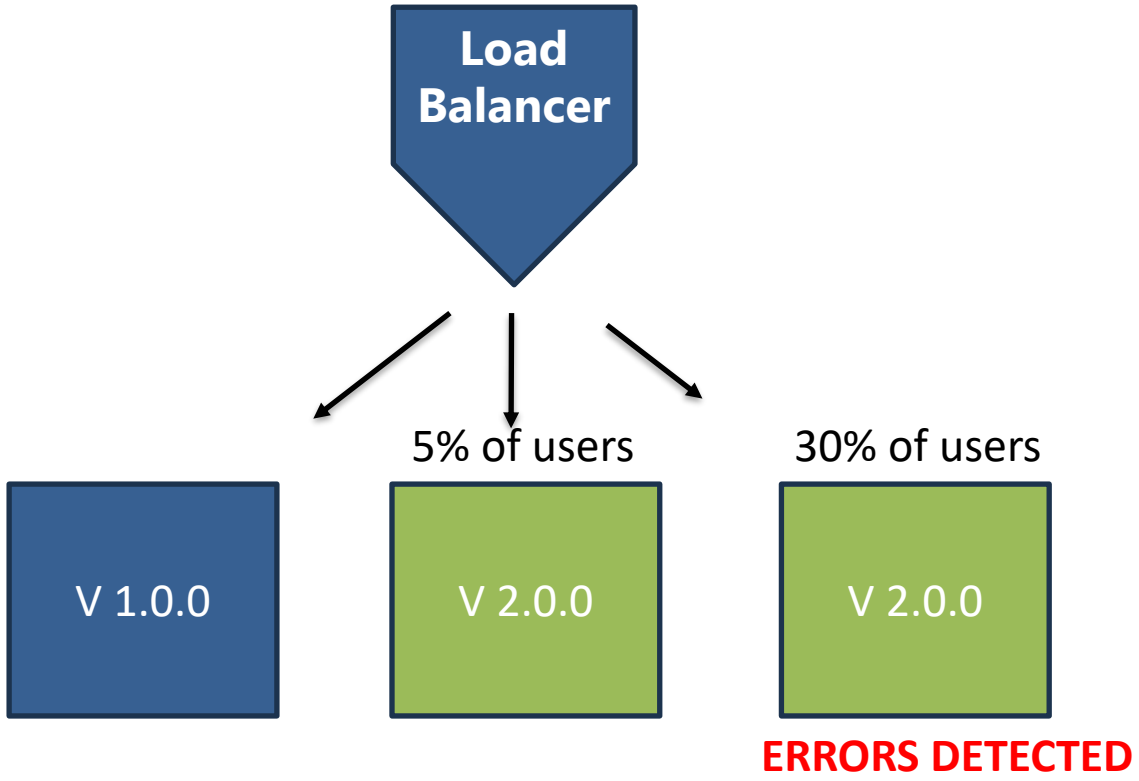
Canary



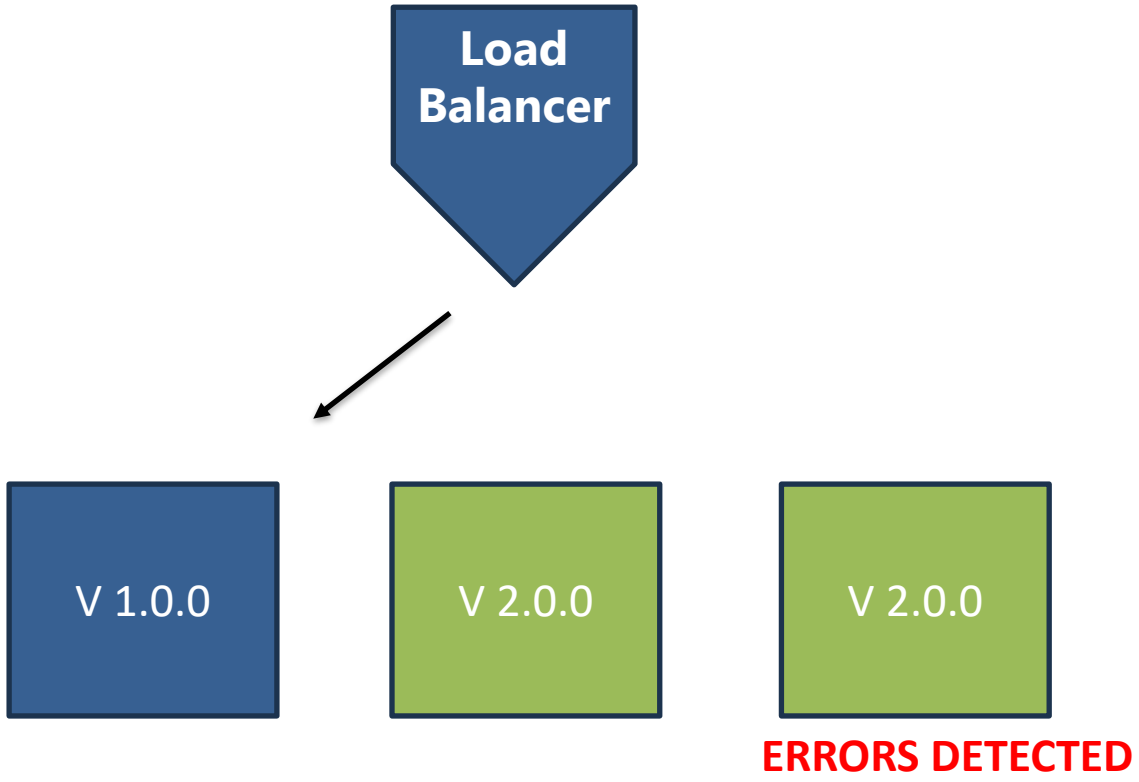
Canary



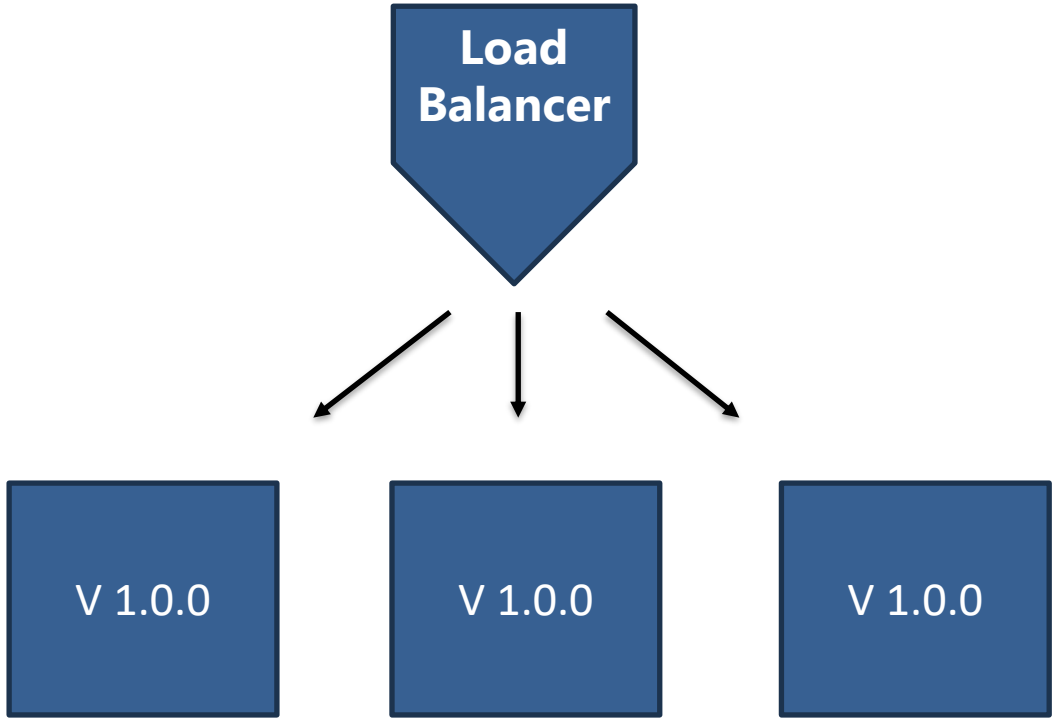
Canary



Canary



Canary



POP QUIZ:

What is a Custom Resource Definition (CRD) in Kubernetes?

A. A replacement for Deployments
B. A way to extend Kubernetes with new resource types
C. A built-in Kubernetes networking feature
D. A container image specification

POP QUIZ:

What is a Custom Resource Definition (CRD) in Kubernetes?

A. A replacement for Deployments
B. **A way to extend Kubernetes with new resource types**
C. A built-in Kubernetes networking feature
D. A container image specification

POP QUIZ:

Which release strategies are supported natively by a Kubernetes Deployment?

A. Blue/Green and Canary
B. Canary and A/B
C. Recreate and Rolling Update
D. Rolling Update and Blue/G

POP QUIZ:

Which release strategies are supported natively by a Kubernetes Deployment?

A. Blue/Green and Canary
B. Canary and A/B
C. **Recreate and Rolling Update**
D. Rolling Update and Blue/Green

POP QUIZ:

Why is Blue/Green deployment easier to test than a rolling update?

A. It requires fewer replicas
B. Traffic is routed automatically
C. The new version can be tested with no live traffic
D. Kubernetes pauses the deployment automatically

POP QUIZ:

Why is Blue/Green deployment easier to test than a rolling update?

A. It requires fewer replicas
B. Traffic is routed automatically
C. **The new version can be tested with no live traffic**
D. Kubernetes pauses the deployment automatically

POP QUIZ:

Why are major database changes considered higher risk than application changes?

A. Databases are harder to monitor
B. Databases cannot scale
C. Rollbacks may cause data loss or incompatibility
D. Databases do not support versioning

POP QUIZ:

Why are major database changes considered higher risk than application changes?

A. Databases are harder to monitor
B. Databases cannot scale
C. **Rollbacks may cause data loss or incompatibility**
D. Databases do not support versioning

POP QUIZ:

Which deployment strategy generally has the fastest rollback?

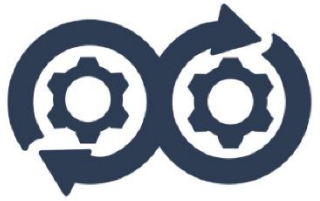
A. Rolling Update
B. Recreate
C. Canary
D. Blue/Green

POP QUIZ:

Which deployment strategy generally has the fastest rollback?

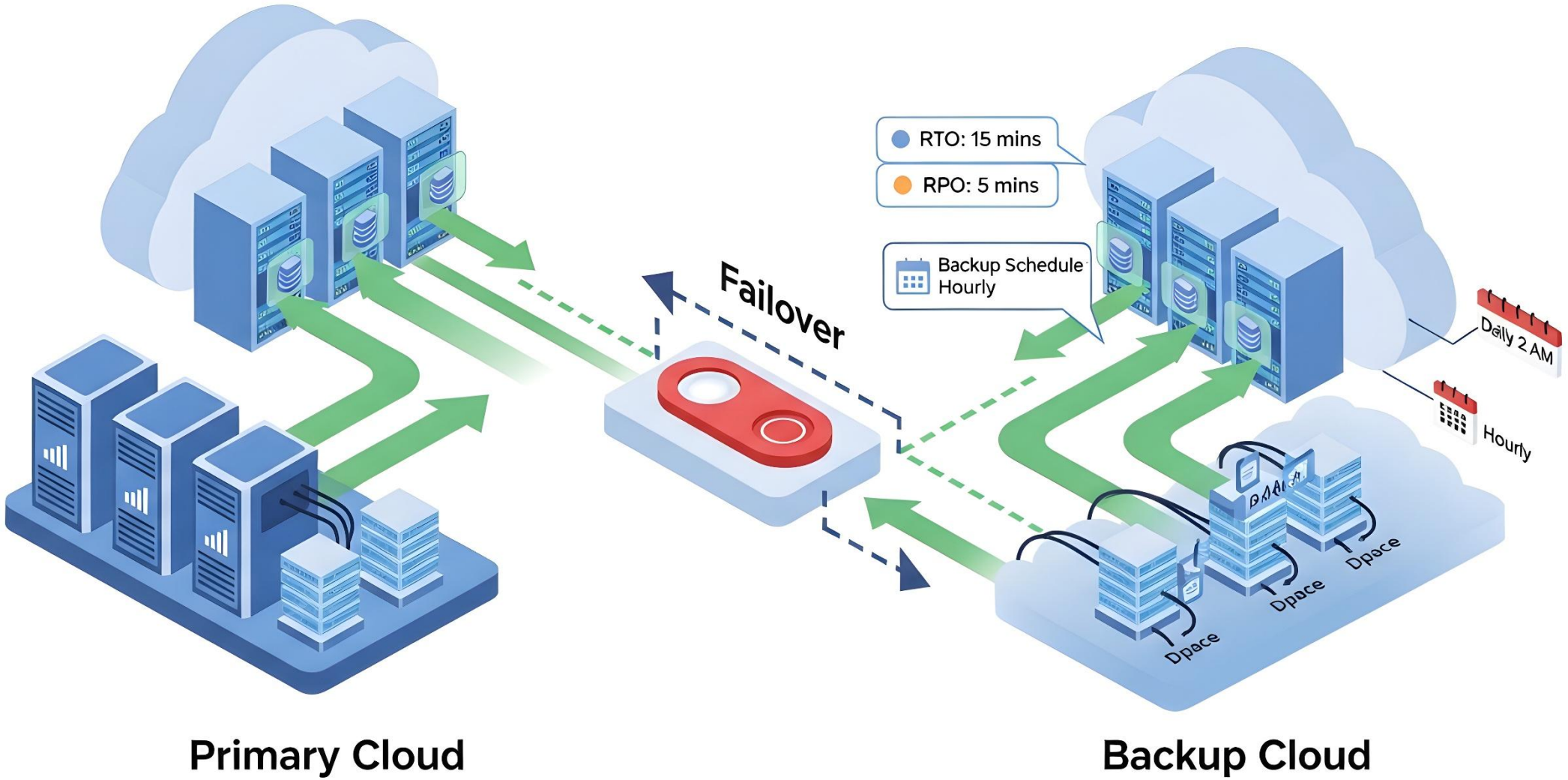
A. Rolling Update
B. Recreate
C. Canary
D. **Blue/Green**

Lab: ArgoCD Blue/Green



Data Center and Disaster Recovery

DR Plan



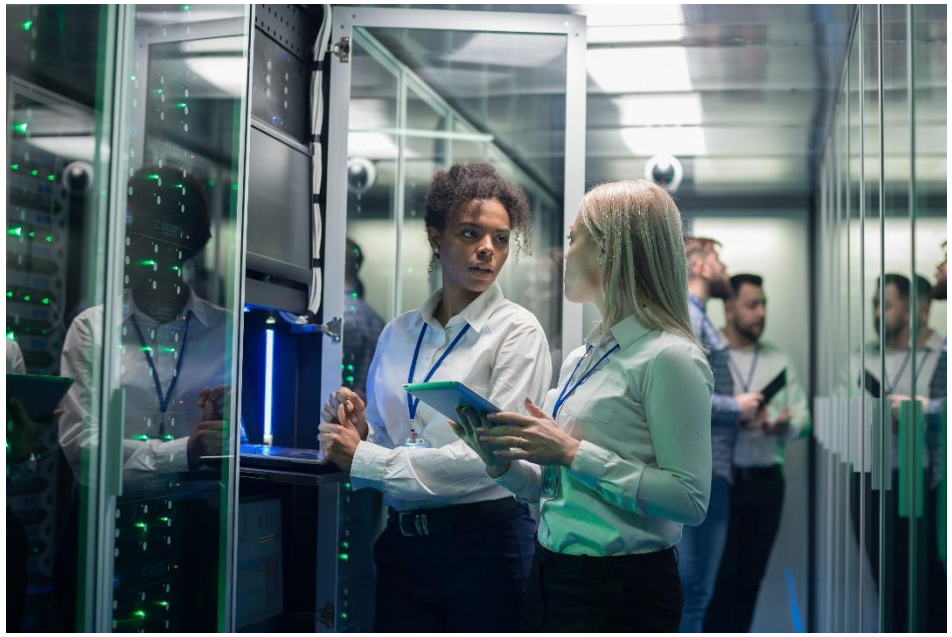
Data Center

A data center is a physical facility that houses the **compute, storage, and networking** systems required to run applications and services.

- Contains servers, storage systems, networking equipment, and power systems
- Designed with redundancy for power, cooling, and connectivity
- Can be **on-premises**, **colocation**, or **cloud-managed**
- Cloud regions are made up of **multiple data centers**

A data center is the foundational building block where modern digital services physically run, whether managed directly or abstracted by the cloud.

Failover Site



A failover site is a **secondary data center or region** designed to take over when the primary site becomes unavailable.

- Usually **geographically separated** to avoid shared risks
- Can be active-active or active-passive
- Requires data replication or synchronization
- Used during disasters or major outages

Failover sites ensure services can continue operating even when an entire location fails.

Disaster Recovery

Disaster Recovery focuses on how systems recover after failure, while resilience focuses on minimizing the impact of failures in the first place.

- Plans for outages caused by infrastructure, software, or human error
- Defines acceptable downtime and data loss
- Uses redundancy, backups, and failover strategies
- Tested regularly through drills and simulations

Strong recovery and resilience planning ensure systems can fail safely and recover quickly

Types of Disasters

Disasters can come from many sources, not just hardware or cloud failures.

Common types:

- **Natural disasters:** earthquakes, floods, fires impacting data centers
- **Infrastructure failures:** region outages, network failures, power loss
- **Software issues:** bad releases, bugs, failed migrations
- **Security incidents:** credential leaks, ransomware, malicious changes
- **Human error:** misconfigurations, accidental deletions

Effective resilience planning assumes failures will happen and prepares for multiple disaster scenarios.

Disaster Recovery Plan

A DR plan defines **how systems recover** when a major failure occurs.

- Clearly defined **primary and failover sites**
- Data backup and replication strategy
- Application and infrastructure recovery steps
- Roles and responsibilities during a disaster
- Communication plan for stakeholders and customers

A good DR plan removes guesswork during high-stress incidents.

DR Planning

RPO and RTO define **how much loss is acceptable** and **how fast recovery must happen**.

- **RPO (Recovery Point Objective):** Maximum acceptable data loss (time-based)
- **RTO (Recovery Time Objective):** Maximum acceptable downtime
- DR should be activated only after confirming the failure is **significant and sustained**
- Teams must consider whether the failover site could be impacted by the same issue
- Regular **DR testing and failover drills** are critical to validate the plan

Disaster Recovery only works if it’s understood, tested, and activated.

Replication vs Disaster Recovery

Replication and auto-failover improve availability, but they do **not replace** a Disaster Recovery plan.

- **Replication** protects against infrastructure failure, not bad data or bugs
- **Auto-failover** handles outages automatically, but may fail silently or propagate issues
- **Disaster Recovery** includes people, process, backups, testing, and decision-making
- DR plans account for **data corruption, security incidents, and human error**

Replication keeps systems running. Disaster Recovery ensures systems can be **restored correctly** when things go wrong.

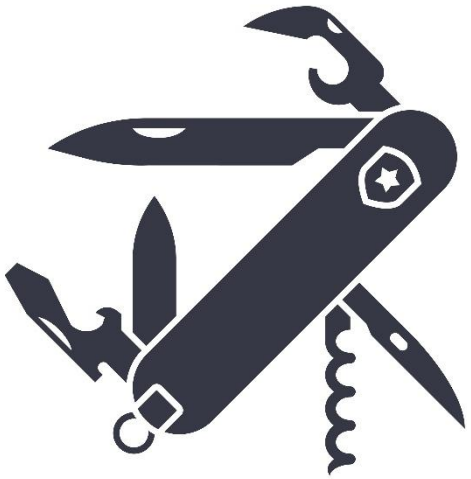
Agents

To support disaster recovery, it's common to use **agents or replication services** that continuously synchronize data from primary systems to failover locations.

- Agents replicate data to secondary sites in near real time or on a schedule
- Used for databases, file systems, object storage, and backups
- Helps meet defined **RPO targets** by limiting data loss
- Can be synchronous (lower RPO, higher latency) or asynchronous (higher RPO, safer)

Data synchronization reduces recovery risk, but it must be carefully designed to avoid spreading corruption or failures to the failover site.

Native Replication



Not all data replication requires installing agents on servers. Many systems support **native, built-in replication** at the platform or service level.

Database-native replication
- Databases like MySQL, PostgreSQL, SQL Server, and Oracle replicate internally
- Uses transaction logs or WAL streams

Storage-level replication
- SANs, cloud block storage, or distributed filesystems replicate data automatically
- Transparent to applications
- Services like S3 Cross-Region Replication or managed database replicas
- Fully handled by the provider

Application-level replication
- Apps write to multiple locations by design (dual-write patterns)

Whenever possible, prefer **native replication mechanisms**, and use agents only when the platform doesn't provide built-in replication.

Synchronous vs Asynchronous Replication

Data replication strategies directly impact performance, availability, and acceptable data loss.

Synchronous replication:

- Writes are not committed until replicas confirm
- **Lowest RPO** (near zero data loss)
- Higher latency and tighter coupling

Asynchronous replication:

- Writes commit immediately on the primary
- **Higher RPO** (some data loss possible)
- Better performance and scalability

Choosing between sync and async replication is a tradeoff between **data safety** and **system performance**, guided by business requirements.

POP QUIZ:

An applications files are asynchronously replicated to another region. The primary region fails suddenly. What is the most likely outcome?

A. No data loss due to automatic failover
B. The database becomes read-only automatically
C. Some recent data may be lost
D. The failover region rejects traffic

POP QUIZ:

An applications files are asynchronously replicated to another region. The primary region fails suddenly. What is the most likely outcome?

A. No data loss due to automatic failover
B. The database becomes read-only automatically
C. **Some recent data may be lost**
D. The failover region rejects traffic

POP QUIZ:

When should a Disaster Recovery plan typically be activated?

A. Immediately after any alert fires
B. After confirming the failure is sustained and significant
C. Only during natural disasters
D. When replication stops

POP QUIZ:

When should a Disaster Recovery plan typically be activated?

A. Immediately after any alert fires
B. **After confirming the failure is sustained and significant**
C. Only during natural disasters
D. When replication stops

POP QUIZ:

A company defines an RPO of 5 minutes. What does this mean?

A. The system must recover within 5 minutes
B. Backups run every 5 minutes
C. Up to 5 minutes of data loss is acceptable
D. Failover must be automatic

POP QUIZ:

A company defines an RPO of 5 minutes. What does this mean?

A. The system must recover within 5 minutes
B. Backups run every 5 minutes
C. **Up to 5 minutes of data loss is acceptable**
D. Failover must be automatic

POP QUIZ:

A service has an RTO of 30 minutes. Which situation violates this requirement?

A. The service fails over in 45 minutes
B. The service loses 2 minutes of data during recovery
C. The service remains partially available but degraded
D. Monitoring detects the outage within 5 minutes

POP QUIZ:

A service has an RTO of 30 minutes. Which situation violates this requirement?

A. **The service fails over in 45 minutes**
B. The service loses 2 minutes of data during recovery
C. The service remains partially available but degraded
D. Monitoring detects the outage within 5 minutes

POP QUIZ:

Why do many organizations test DR plans regularly instead of relying on documentation?

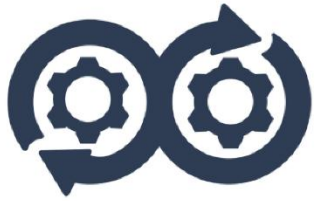
A. Documentation becomes outdated
B. Automation replaces human decision-making
C. Testing improves performance
D. DR plans may fail when executed under real conditions

POP QUIZ:

Why do many organizations test DR plans regularly instead of relying on documentation?

A. Documentation becomes outdated
B. Automation replaces human decision-making
C. Testing improves performance
D. DR plans may fail when executed under real conditions

Lab: Data Center DR



Congratulations

Day 5 Summary

Today you explored how production systems are deployed, protected, and recovered when things go wrong.

- Deployment strategies and their tradeoffs
- rollback planning, and release decision making
- Synchronous vs asynchronous replication and their impact on RPO/RTO
- Agent-based vs agentless data synchronization approaches
- Disaster Recovery concepts, data centers, and failover planning

Strong deployment and recovery strategies reduce downtime, protect data, and make change safer at scale.