

API & Microservices

What is the purpose of warmup? (Select all possible answers)

- To ensure that the first request is handled as quickly as all following requests
- To make sure all components and connections are initialized when the application starts
- To discover configuration issues at startup

What is the purpose of a canary release? Select the correct answers.

- To expose a small group of users to the new software version before a full release deployment
- To validate the stability of the new software version
- To decrease the number of avoidable incidents
- All the answers are correct

Tightly coupled services enhance Agility and Changeability and are vital in a microservices architecture

- TRUE
- False, loosely coupled services are advantageous in case of microservices architecture
- Both Choice1 & Choice2
- Neither Choice1 & Choice2. Depends on the design

What do the liveness probes for ICHP deployments indicate?

- Unsuccessful indicates that the container is not running properly

- Successful means that the application is up, but it might not be ready for business
- Liveness probe determines if a container is ready to serve requests

Which statement is true?

- Deadlines are useful for any request
- Deadlines are used to stop complex chains to continue, when they take too long for the client to wait
- Deadlines make sure that applications prioritize some requests over other, less time-pressed requests

What are recommended approaches for application upgrades in ICHP?
Select the correct answers.

- Use Rolling updates
- Use Canary releases
- Use the Kingsroad
- All the answers are correct

What happens if all instances are marked as unavailable?

- The client will pick one random unavailable instance and call it anyway
- The client will not make any calls to any instance for a while
- The client will call all unavailable instances at once, hoping at least one responds, accepting the one that responds first

When using Autocert integration in Kingsroad for IPC VMs, how do you reload the renewed certificate into the application?

- This is done automatically by a certbot
- There is not such capability
- By providing a custom script in your config-zips that restarts the application.
- None of the above

How are resource metrics collected in ICHP? (Select all possible answers)

- By scraping prometheus metrics directly from the application pods
- By pulling metrics using Grafana
- Metrics are automatically collected by the 'Cluster Monitoring Prometheus'
- All the answers are correct

What are the most common timeouts during API communication?

- Request and connection timeouts
- Query timeouts
- File timeouts
- Garbage collection timeouts

What is critical for availability in a long API chain? (Select all possible answers)

- Using a single data center for hosting the entire chain
- Implementing proper error handling and retries between API
- Reducing the number of APIs in the chain to minimize complexity
- Ensuring that each API in the chain has its own dedicated database.

How are resource metrics collected in ICHP? (Select all possible answers)

- By scraping prometheus metrics directly from the application pods

- By pulling metrics using Grafana
- Metrics are automatically collected by the 'Cluster Monitoring Prometheus'
- All the answers are correct

What do the API operational modes of a touchpoint service indicate?

- Whether the endpoints are approved or not.
- Whether the application instance is ready to serve request
- Whether TLS negotiations are not successful or not.
- AED Connectivity established or not

When do we report that we are "Live" in AED?

- When service the service to "live" on the API & Stream Marketplace of the Touchpoint portal
- When the application server is able to respond to a request. i.e. the liveness probe is successful
- When all components and connections are initialised

Why do we need back pressure, in addition to request timeouts?

- Fixed timenute in a long comnley chain are problematic
- Because back pressure enables exponential backoff in the client
- Back pressure allows for servers to organize the load amongst themselves

What is the purpose of warmup? (Select all possible answers)

- To ensure that the first request is handled as quickly as all following requests
- To make sure all components and connections are initialised when the application starts.
- To discover configuration issues at start-up

Why is idempotency of the application an important request for the IHCP platform?

- Idempotency ensures application can handle varying user loads
- Idempotency guarantees zero data loss during a disaster
- Both choice 1 & 2
- Idempotency allows for reliable and repeatable application deployments.

Why is TracING vital for HA in a microservices architecture?

- Root cause Analysis and isolating failures.
- Scaling Resources and Real time monitoring
- Both Choice 1 & 2
- Only choice 1.

When does circuit breaker kick in?

- When all the instances are unavailable
- When the client observes error behavior with an instance
- When there is no point in waiting for the response anymore
- When the called instance determines it cannot handle the load.

Why is TracING vital for HA in a microservices architecture?

- Root Cause Analysis and Isolating Failures
- Scaling Resources and Real time monitoring
- Both Choice1 & Choice2
- Only Choice1

For what ICHP versions is the quota autoscaler available?

- ICHP V1
- ICHP V2
- ICHP V3
- None of the above All the above

What do the API operational modes of a touchpoint service indicate?

- Whether the endpoints are approved or not
- Whether the application instance is ready to serve request
- Whether TLS negotiations are successful or not
- AED connectivity established or not

How can you mitigate data loss in a long API/microservices chain where data is persisted at the end of the chain?

- Increase the network bandwidth between service
- Implement frequent backups of the data
- Use asynchronous communication between services
- Add redundant instances of the last service

What is critical for availability in a long API chain? (Select all possible answers)

- Using a single data center for hosting the entire chain
- Implementing proper error handling and retries between APIs
Reducing the number of APIs in the chain to minimize complexity
" • Ensuring that each API in the chain has its own dedicated database
Explanation: API & Microservices

What are recommended approaches for application upgrades in ICHP?
Select the correct answers.

- Use Rolling updates
- Use Canary releases
- Use the Kingsroad
- All the answers are correct

Can confidence checking (operational mode) deployments be disabled for Production

- Yes
- No

How can you mitigate data loss in a long API/microservices chain where data is persisted at the end of the chain?

- Increase the network bandwidth between services
- Implement frequent backups of the data
- Use asynchronous communication between services
- Add redundant instances of the last service

What is critical for availability in a long API chain? (Select all possible answers)

- Using a single data center for hosting the entire chain
- Implementing proper error handling and retries between APIs
- Reducing the number of APIs in the chain to minimize complexity
- Ensuring that each API in the chain has its own dedicated database

When does circuit breaker kick in?

- When all instances are unavailable
- When the client observes error behavior with an instance
- When there is no point in waiting for the response anymore
- When the called instance determines it cannot handle the load

Tightly coupled services enhance Agility and Changeability and are vital in a microservices

- * TRUE
- * False, loosely coupled services are advantageous in case of microservices architecture
- * Both Choice 1 & Choice2
- * Neither Choice1 & Choice2. Depends on the design

Data

What is the difference between Gold and Platinum service level backup for virtual servers in IPC?

- ✓ Platinum is intended to be used for DR enabled workloads only, and stores backups data in both datacenters while Gold stores backups only in one datacenter
- Gold provides retention period of 31 days while Platinum provides retention period of 14 days
- Gold requires downtime during restore process while Platinum does not

What are the two service level options you can choose from when ordering a Pluggable Database (PDB) in DBaaS?

- ✗ Gold and Platinum
- ➔ Thin and Normal
- Thin and Full
- Non production and Production

Zero data loss is very difficult because:

- Backup and recovery processes are inherently slow
- Data centers are often located in remote areas

- Network connections are unreliable
- It's challenging to synchronize data in real-time

What is the advantage of using DBaaS by just one application or purpose (DBMS tightly coupled to one purpose)?

- We can have multiple purposes enter data if they belong to one squad
- It's a disadvantage if we don't use resources to the maximum
- It's isolated data and integrity could be maintained without any dependency on other purposes
- Both Choice2 and Choice3

Which of the options is NOT available as a second day operation for DBaaS users?

- Database resizing
- Changing patch rings
- Starting and Stopping your Oracle Database
- Switching between data centers on container level

A DBMS is considered tightly coupled to an application, why?

- App's code and database schema are tightly coupled components
- They are loosely coupled and advantageous that way
- Both services to go as one microservice
- Both Choice1 and Choice3

A snapshot is the standard tool used to manage your backups in IPC for your VM's, attached storage and SQLServer (Select all possible answers)

- False, databases backup is not based on snapshots (image level)
- False, there is no default method as users select file level or image level
- True, by default all backups are based on snapshots
- False, VM storage backups doesn't have snapshot capability

What is Cassandra's primary advantage over traditional relational databases like Oracle, Microsoft SQL Server, and MySQL?

- Cassandra provides structured data storage with schemas.
- Cassandra is not distributed and has a single point of failure.
- Cassandra is schema-less and handles unstructured data and has no single point of failure.
- Cassandra uses SQL and CQL as query languages

What type of environments are Development and Test in Cassandra KaaS hosted on?

- Stand-alone platform hardware
- Shared virtual infrastructure
- Laptop

The RPO describes:

- The maximum tolerable downtime for an application
- The amount of data that can be lost during a disaster
- The time it takes to recover an application after a failure
- The geographic location of a Disaster Recovery (DR) site

How many nodes per datacentre are required in a KaaS Cassandra cluster?|

- 1 node
- 2 nodes
- 3 nodes
- At least 5 nodes

What does 'RTO' stand for:

- Recovery time objectives
- Retention time objective
- Restore time objective

How often are provider-driven patches rolled out for DBaaS?

- Monthly
- Annually
- Quarterly
- Weekly

What do you need to do to minimise the risk of DBaaS outages affecting end users and applications?

- Monitor health status
- Implement failover mechanisms and maintain services availability
- On an incident check with concerned team
- Investigate and fix it.

What is the 'Recovery Point Objective' (RPO) for the DBaaS service levels?

- 5 minutes for Normal and 3 minutes for Thin
- 1 & minutes for Thin and 4 minutes for Normal
- 4 minutes for Thin, and 2 minutes for Normal
- 2 minutes for both Thin and Normal.

Which appliance is used by DBaaS to employ the Oracle backup feature?

- Elastic Cloud Storage (ESC)
- Network Attached Storage (NAS)
- Zero Data Loss Recovery Appliance (ZDLRA)

A NoSQL datastore should never be used as a SoR for business data.
why? (Select all possible

- NoSQL is document oriented
- NoSQL can still be used and data can be managed
- SQL relation database ensures consistency
- Depending on the requirement, we can choose accordingly

When you need to guarantee the consistency in your data of your VM.
what kind of backup do you

- Full backup
- Incremental backup
- Differential backup
- Snapshot Backup

What is Cassandra's primary advantage over traditional relational databases like Oracle. Microsoft, SQL Server and MySQL?

- Cassandra provides structured and data storage with schemas.
- Cassandra is not distributed and has a single point of failure.
- Cassandra is schema-less and handles unstructured data and has no single point of failure.
- Cassandra uses SQL and CQL as query languages

Which environments are provided on dedicated infrastructure in Cassandra Key space as a service (KaaS)

- Sandbox and Development
- Development, Test
- Development, Test and Acceptance
- Acceptance and Production

What is the primary purpose of the Oracle Zero Data Loss Recovery Appliance (ZDLRA)?

- To provide continuous backups
- To monitor database performance
- To switch between data centers
- To monitor database data consistency

By backing up your data you can ensure the integrity of your data

- True, backups always prevent data corruption
- True, backups provide real-time data synchronization
- False, backups only help in case of hardware failures
- False, backups can sometimes introduce data inconsistencies

What is the 'Recovery Point Objective' (RPO) for the DBaaS service levels?

- 5 minutes for Normal, and 3 minutes for Thin
- 8 minutes for Thin, and 4 minutes for Normal
- 4 minutes for Thin, and 2 minutes for Normal
- 2 minutes for both Thin and Normal

A snapshot is the standard tool used to manage your backups in IPC for your VM's, attached storage and SQLServer (Select all possible answers)

- False, databases backup is not based on snapshots (image level)
- False, there is no default method as users select file level or image level
- True, by default all backups are based on snapshots
- False, VM storage backups doesn't have snapshot capability

A DBMS is considered tightly coupled to an application, why?

- App's code and database schema are tightly coupled components
- They are loosely coupled and advantageous that way
- Both services to go as one microservice
- Both Choice1 and Choice3

Which environments are provided on dedicated infrastructure in Cassandra Key sapce as a Service (KaaS)?

- Sandbox and Development
- Development and Test
- Development, Test and Acceptance
- Acceptance and Production

When you patch your machine/VM a backup is created automatically

- Yes, a backup is created automatically when you patch your machine/VM.
- Backups are only created automatically for critical security patches
- No, backups are not related to patching and need to be initiated separately
- Backups are only created automatically for physical machines, not VMs

Which team guides application teams through the onboarding process for Cassandra in Key sapce as a Service (KaaS)?

- Minions squad
- DataStax Academy

- BCM Department

What is a responsibility of application teams using Cassandra Key space as a Service (KaaS)?

- Managing cluster infrastructure
- Performing OS patch management
- Configuring the consistency level for their Keyspace

When you use DBaaS normal, failover by the application of the Oracle DB is managed for you

- No, it is not managed
- Yes, Normal service level offers switchover(by application team)/ Failover(by provider team)
- Application team need to manually failover
- Both Normal and Thin options, failover done automatically

High Availability basics

Local cluster zones in IPC can be used to achieve higher availability for an application within a single data center.

➔ TRUE

✗ FALSE

What is a risk with an active-passive architecture for HA?

- ✓ Complex failover procedures
- Increased resource utilization

- Reduced data redundancy
- Scalability limitations

What is the problem with the option "DR capable" during provisioning?

- It requires specialized hardware
- It may lead to overallocation of resources for the workload
- Enabling the feature can result in increased network latency for the workload
- It requires multiple private networks, one for each datacenter

What is the purpose of a Stretched Zone cluster in IPC?

- To achieve higher availability within a single data center
- To synchronize workloads across two physical datacenters for infrastructure disaster recovery
- To provide load balancing services across multiple private networks
- To ensure data is distributed

Can teams rely solely on IPC's infrastructure disaster recovery option for resiliency and availability in their applications?

- Yes
- No

How many data centers does IPC have for hosting infrastructure?

- Three (DCR/WPR/CSM)
- Two (DCR/WPR)
- Four (DCR/WPR/CSM/MRX)

- Five (DCR/WPR/CSM/MRX/LON)

Why should you never use IP-addresses in your application?

- IP addresses are harder to remember than domain names
- IP addresses can change, leading to broken connections
- IP addresses can reveal sensitive network information
- IP addresses are not compatible with modern network protocols

What is the purpose of sub-availability zones in IPC's local cluster zones?

- To provide disaster recovery capabilities
- To provide load balancing services across multiple private networks
- To achieve higher availability within a single data center
- For executing data center failover

Is infrastructure disaster recovery option recommended for all types of applications in IPC?

-  No
-  Yes

Why do we prefer an active-active configuration for HA?

- To reduce resource utilization
- To simplify management and maintenance
- Improve Availability, Reliability and scalability.

- To enhance data backup processes

What type of infrastructure Disaster Recovery method is used for deployments in the stretched cluster?

- Active-Passive infrastructure DR
- Active-Active infrastructure DR
- Hybrid cloud deployment DR\
- Active hot standby using asynchronous

Who is responsible for performing OS Patch Management and Cassandra upgrades in Cassandra Key sapce as a Service (KaaS)?

- Application teams
- Minions team
- DataStax Academy

When you have requested your VM in the IPC with the non-preferred option DR capable, IP address changes during a DR (True/false)

- FALSE
- TRUE
- It depends on the chose sub-availability zone
- It depends on the chosen backup service level

Why do we prefer an active-active configuration for HA?

- To reduce resource utilization
- To simplify management and maintenance
- Improve Availability, Reliability and scalability
- To enhance data backup processes

Local cluster zones in IPC can be used to achieve higher availability for an application within a single data center.

- TRUE
- FALSE

You have deployed your API four times, two in each DC, for HA reasons. Your total load is 100 TPS. How much load must one instance be able to handle?

- 25 TPS
- 50 TPS
- 75 TPS
- **100 TPS**

When is explicit routing used? (Select all possible answers)

- When you want the client to use a specific instance for a request
- For confidence checking a service
- Routing traffic to a data center

How can you ensure your VM's are provisioned in different update and fault domains?

- Choose different availability/sub-availability zones for your VMs
- Implement Disaster Recovery
- VMs cannot be provisioned in multiple availability zones
- Only data centers are selectable during a request

What do we mean by "an application must be idempotent"?

- The application should have a single point of failure
- The application's performance must remain consistent
- The application can handle varying user loads
- **The application's operations can be repeated safely**

Can teams rely solely on IPC's infrastructure disaster recovery option for resiliency and availability in their applications?

- Yes
- **No**

Can teams rely solely on IPC's infrastructure disaster recovery option for resiliency and availability in their applications?

- Yes
- **No**

Which are the sub-availability zones are there inside the local cluster zones in IPC?

- Red and Green
- Red and blue
- Yellow and blue
- Red, Blue & yellow

You have deployed your API four times. two in each DC. for HA reasons. Your total load is 100 TPS. How much load must one instance be able to handle?

- 25 TPS
- 50 TPS
- 75 TPS
- 100 TPS

Why is HA more difficult when your application is stateful?

- Stateful applications are inherently slower.
- Stateful applications lack redundancy
- Stateful data must be synchronised between instances
- Stateful applications cannot utilize load balancing

What are the sub-availability zones (failure domains inside the local cluster zones in IPC)?

- Red and Green
- Red and Blue
- Yellow and blue
- Red, blue & yellow

You deploy a COTS application HA in one DC in IPC. How can you configure Load Balancing for the API endpoints of the COTS application?

- ICHP would take care of load balancing of API Gateway even in one DC
- Deploy nginx and use as a load balancer
- Both choice 1 & 2
- Use the F5 LTM as a load balancer

What is the problem with the option "DR capable" during provisioning?

- It requires specialized hardware

- it may lead to overallocation of resources for the workload
- Enabling the feature can result in increased network latency for the workload
- It requires multiple private networks, one for each data center

Why is testing the non-happy flow important? Select all answers

- To ensure proper error handling and robustness of an application
- Avoid customer distribution
- Not important unless the testing of the base case is successful
- All of the above.

In case of a real disaster. what is the primary responsibility of KaaS users to ensure continued availability of their application.

- To turn off native transport
- To rely on IPCs automatic traffic switching
- To implement connections to both datacenters on the application level
- To contact the minions squad for assistance.

How does IPC synchronize workloads across two physical data centers for disaster recovery capable deployments?

- By using a-synchronous storage replication
- By using synchronous storage replication
- by using load balancers

SRE/Observability

What does it mean when the 99 percentile latency is 100ms?

- The standard deviation is 100ms
- 99 percent of the requests have a latency of 100ms or less

- 99 percent of the requests have a latency of 100ms or more
- 99 percent of the request have a latency of 100ms

What of the following statement is true about the spike test AND the breakpoint test?

- The spike test has a steady load before having a short burst of traffic and the breakpoint test has a gradually increasing load
- The breakpoint test has a constant high load and the spike test has a gradually increasing load
- The spike test has simulates a constant high peak load (eg black Friday) and the breakpoint test simulates and increasing load
- The breakpoint test simulates a constant high peak load (eg black Friday) and the spike test its goal is to test long term stability of a system

I have two priority 1 incidents with a parent - child relation. Which statement is correct?

- a Post Mortem is only mandatory for the parent incident
- a Post Mortem is only mandatory for the child incident
- a Post Mortem is only mandatory for both P1 incidents
- a Post Mortem is optional

Good observability can be only designed after the first run in production

- TRUE
- FALSE

What is full-stack observability? (Select all possible answers)

- ✗ Provides insight into application performance
- ➡ Monitoring based on user ratings
- monitoring of computer systems, servers, processes and equipment
- ✓ Provides insight into application, infrastructure, and UX performance.

Monolith are easy for observability (Select all possible answers)

- ✗ Monoliths have fewer components, making them simpler to monitor
- ✓ Monoliths often have centralized logging and monitoring
- ➡ Observability in monoliths can become challenging as they grow in complexity
- Monoliths are inherently designed for distributed tracing

What is part of the shift left basic principles (Select all possible answers)

- ✗ Canary releasing
- ✓ Test without dependencies
- ✓ Test frequently
- Test end to end by a central test team

What is the error budget? (Select all possible answers)

- The ratio of bad events to the total events

- ➔ The pain tolerance for your users
- ✓ The amount of errors over some period of time before your users start being unhappy
- ✗ The proportion of events detected that were significant.

What is a SLI?

- It specifies a target level for the reliability of your service.
- It is an agreement between the service provider and customer about service deliverables
- ✓ Is a measurement of the characteristic of a service

How is error budget calculated in percentage?

- ✓ 100 - SLO
- 100 - SLI
- 100 - SLA

What is a SLI?

- It specifies a target level for the reliability of your service.
- It is an agreement between the service provider and customer about service deliverables
- Is a measurement of the characteristic of a service

Which of below answers are non functional requirements? (Select all possible answers)

- Latency
- Throughput
- Contracts/pacts
- Success rate

What is the preferred solution for delivering alerts at ING?

- TracING
- IAT
- Elastic
- Prometheus

What does it mean when the 99 percentile latency is 100ms?

- The standard deviation is 100ms
- 99 percent of the requests have a latency of 100ms or less
- 99 percent of the requests have a latency of 100ms or more
- 99 percent of the request have a latency of 100ms

Monitoring helps engineers/SREs determine that a problem exists, while observability

- tells you when something is wrong
- while observability describes if you can monitor a system
- tells you "what's broken" and indicates the symptom

What is Singleview?

- It will ultimately allow you to tie several MDPL products into one single view
- A tool to find responsible teams/persons
- One dashboard to show all your assets deployed in IPC
- One dashboard to show all your firewall rules

Which tool is used for a unified approach for communication during critical events?

- ServiceNow
- SLIM
- Everbridge
- eReporter

What of the following statement is true about the spike test AND the breakpoint test?

- The spike test has a steady load before having a short burst of traffic and the breakpoint test has a gradually increasing load
- The breakpoint test has a constant high load and the spike test has a gradually increasing load
- The spike test simulates a constant high peak load (eg black Friday) and the breakpoint test simulates an increasing load
- The breakpoint test simulates a constant high peak load (eg black Friday) and the spike test's goal is to test long term stability of a

What is the error budget? (Select all possible answers)

- The ratio of bad events to the total events
- The pain tolerance for your users
- The amount of errors over some period of time before your users start being unhappy
- The proportion of events detected that were significant

What should teams do first if they want to use the Global Monitoring alerting service IAT?

- Create an account on their website
- Install their proprietary software
- Check how to onboard and use their monitoring services

What are the four golden signals of SRE?

- Latency error budget traffic and saturation
- Latency, error rate, alerts and saturation
- Metrics, error rate, traffic and saturation
- Latency, error rate, traffic and saturation

How to register an improvement action you have found during the RCA?

- As a child task under the post mortem
- As a child Story or Feature under the post Mortem
- As a task related to the Post Mortem
- As a regular Story or Feature related to the Post Mortem

Which tool is used for a unified approach for communication during critical events?

- ServiceNow
- SLIM
- Everbridge
- eReporter

Is the following statement correct: Deployment failure or Software failure are examples of incident root cause categories.

- Correct
- Incorrect

What is at the bottom of the Site Reliability Engineering (SRE) pyramid?

- Data
- Product
- Development
- Monitoring

What is singleview?

- It will untimely allow you to tie several MDPL products into one single view.
- A tool to find reasonable teams/persons
- It will ultimately allow you to tie several MDPL products into one single view
- One dashboard to show all your assets deployed in IPC

What does it mean when the 99 percentile latency is 100ms?

- The standard deviation is 100ms
- 99 Percent of the requests have a latency of 100ms or less.
- 99 Percent of the requests have a latency of 100ms or more.
- 99 Percent of the requests have a latency of 100ms.

Is the following statement correct: Incident triggers are the underline reasons why incidents occurred?

- Correct
- Incorrect

What is the difference between Splunk Synthetics and WTSS (Select all possible answers)

- Splunk controls maintenance while WTSS does not
- WTSS supports mobile apps, Splunk does not
- WTSS supports internal APIs. Splunk does not
- Splunk supports internal APIs. WTSS does not

What is the difference between ELKaaS and Log4all?

- With EL KaaS you build your own Elasticsearch cluster instead of centrally managed

- With Logall you build your own elasticsearch cluster, EL KaaS centrally managed
- ELKaaS is based upon open-source tools Prometheus and Grafana. loc4all is not

Intror2

0:03

Hi and welcome to this reliability training.

0:06

This training provides an overview of IT services, resilience, design principles as well as a deep dive into those services, infra services like Dbas or TPA, services like the Touchpoint service Mesh and API gateway and observability tooling like tracing and RTK too.

0:29

Of course the goal of this training is to increase the reliability of ING by helping you, our engineers, by getting this essential information, getting best practices, guidelines.

0:43

This training is a minimum viable product and the target is to have one platform and one format at the moment.

0:50

Some parts might be links to documentation and some movies might have a different look and feel and for example I will be replaced by an avatar soon.

1:03

That's AI for you, right?

1:07

But it will make the creation and maintenance of the content for this training much much easier.

1:14

So what is exactly reliability and related terms like resilience and high availability?

1:21

Reliability is about the customer experience.

1:24

Reliability ensures that your application can meet the commitments you make to your customers.

1:31

Resilience is about the ability of your system to gracefully handle and recover from failures.

1:39

For example, if you have a network outage of 10 or three seconds, it should not lead to a downtime of one hour.

1:49

And high availability is the ability of a system to operate continuously to have a second system to eliminate single points of failures.

2:01

So high availability helps you to be resilient and being resilient helps you to be reliable.

2:09

And check out the slides of the recent CTO webcast.

2:13

It states our mission and that we want to improve our reliability of our applications, platforms, infrastructures, processes and architecture.

2:25

And why is this urgent?

2:26

Well, you've seen the news that customers are complaining, customers are angry over yet another malfunction in a short time.

2:36

And of course, as a bank we offer an important function within society, and we are obliged to offer a channel availability of 99.78 or for ideal even 99.88.

2:53

And they emphasize also the difference between reliability and resilience.

2:58

Resilience is the capacity to absorb shocks, to adapt to changing conditions, and it mentions meantime before failure.

3:08

That's the total operating time of a system without a failure or meantime to repair.

3:15

How much time do you need to restore your service?

3:19

And to improve our reliability, we need a systematic approach.

3:24

We need to prevent any impact from occurring, we need to restore our services as fast as we can, and we need a systematic approach to identify the underlying cause and address it effectively.

3:38

So a comprehensive root cause analysis was executed and they identified 5 underlying causes that contributed to the outages.

3:48

The first one is quality, a systematic shortfalls in our quality of knowledge, process and technology.

3:57

And this knowledge part, that's where this training comes in.

4:01

5% was caused by a broken component, let's say a defect equipment.

4:07

30% was caused by a defect changes were broad life that contained A defect that 10% caused by overloading the system, perceives an unexpected load or excessive volume and collapsed.

4:24

And 20% had to do with capacity management under resource A part of the system had insufficient capacity.

4:35

So they stressed the urgency that we need to improve our reliability as soon as possible.

4:41

That's why we want you to follow this training.

4:44

But there are of course many initiatives and we want to have resilience embedded in the systems and processes.

4:52

We want infrastructure as code and policy based automation.

4:57

What do I mean?

4:58

For example, in King's Rd.

5:01

the pipeline for ICHP, if you choose two replicas it's auto magically deployed onto availability zones.

5:12

We want conformity bots like the health report in ICHP which checks if you are deployed into availability zones or if you deploy every 30 days.

5:26

We want to improve our resilience testing, maybe even chaos testing.

5:32

We want self healing capabilities.

5:34

In an ICHP.

5:35

There are already some self healing capabilities and we want AI OPS where machine learning learns from data, historical data finds a pattern.

5:48

For example, if we know the normal behavior of an application using its law files or traffic flows, we can do anomaly detection and take an automated action.

5:59

And these initiatives will make your life easier, will reduce the cognitive load.

6:05

Now you could say, on the one hand we want to automate resilience patterns, have resilience patterns built into the platforms and processes, and on the other hand, we want to enforce resilience using those conformity bots by using a health report to check if you're deployed on multiple availability Zones for example.

6:28

But by the way, about this cognitive load, I'm personally convinced that you always need to have some basic knowledge about resilience patterns.

6:37

Something always goes wrong, as was also stated in the slides of the CTO webcast, that if you are in the war room and you don't know your failover mechanisms, if you don't know your availability zones, if you don't know your traffic flows, that you really have a problem.

6:57

You always need some basic knowledge about your resilience patterns and your traffic flows.

7:04

For example, as long as we cannot generate a physical design picture automatically, you have to create it manually when you are in the war room.

7:15

You can see at a glance the availability zones, the data centers, your traffic flows, your network name.

7:22

It will also make communication with other engineers much, much easier.

1

00:00:14,310 --> 00:00:14,975

Hello,

2

00:00:16,064 --> 00:00:23,335

In this module we'll briefly go over some basic IPC concepts, namely the sub availability zones, how to

3

00:00:23,345 --> 00:00:27,212

order a service in them and how to use that in your application design.

4

00:00:30,112 --> 00:00:34,687

Let's deep dive now in IPC data centers and availability zones.

5

00:00:36,037 --> 00:00:38,507

For all the IPC family members

6

00:00:38,787 --> 00:00:44,149

the hardware to service infrastructure for DevOps teams is hosted in two data centers

7

00:00:44,809 --> 00:00:48,074

WPR (Data center Wilgenplas Rotterdam)

8
00:00:48,664 --> 00:00:51,399
and DCR (Data center Roosendaal)

9
00:00:52,019 --> 00:00:58,749
When you request services in IPC the two datacenters
are referred to as availability zones, as you can

10
00:00:58,759 --> 00:01:00,137
see in the request form..

11
00:01:00,756 --> 00:01:06,567
Based on your application requirements, you decide
how you will distribute your application components

12
00:01:06,616 --> 00:01:09,487
over the availability zones as part of your design.

13
00:01:11,387 --> 00:01:14,361
Now let's take a deeper look into the data center
setup.

14
00:01:15,961 --> 00:01:21,151
During the request of infrastructure in the self-service
portal you can select where you want to place your

15
00:01:21,161 --> 00:01:22,024
components.

16
00:01:22,644 --> 00:01:26,724
Based on the diagram I will explain what the specific
areas mean.

17
00:01:27,824 --> 00:01:33,564
We have two data centers where infrastructure can
be placed in so-called 'Local cluster zones'; These

18
00:01:33,574 --> 00:01:40,299
zones are used for placing infrastructure in a specific
data center, either WPR or DCR.

19
00:01:41,399 --> 00:01:45,736
Inside the 'local cluster zones' you can see there
is a 'Red' and a 'Blue' zone.

20
00:01:46,537 --> 00:01:52,616

During a server request the two zones are referred to as 'sub-availability zones' as you can see in the

21
00:01:52,626 --> 00:01:53,549
request form.

22
00:01:54,169 --> 00:02:00,269
The 'Red' or 'Blue' zone can be used to achieve higher availability for your application inside a single

23
00:02:00,279 --> 00:02:01,024
data center

24
00:02:01,374 --> 00:02:06,934
since the RED and BLUE zones are separate alleys in the data center with separated hardware, power,

25
00:02:06,934 --> 00:02:09,124
cooling and separate maintenance slots.

26
00:02:10,224 --> 00:02:13,854
In addition to this, there is also a 'Stretched cluster Zone'.

27
00:02:14,274 --> 00:02:21,144
IPC Provides Disaster Recovery capable deployments (Active-Passive infrastructure DR), which synchronize

28
00:02:21,144 --> 00:02:24,087
your workloads across two physical data centers.

29
00:02:24,697 --> 00:02:30,516
Capacity is reserved in the other data center for your workloads to be restarted automatically in case

30
00:02:30,527 --> 00:02:32,012
of a data center outage.

31
00:02:32,641 --> 00:02:37,824
This is what the stretched zone can be used for, specifically for legacy applications.

32
00:02:38,974 --> 00:02:44,994
Similar to the local cluster zones, the stretched cluster zone uses the Red and Blue sub availability

33
00:02:45,004 --> 00:02:45,937
zone principle.

34
00:02:46,536 --> 00:02:53,387
By using software-defined networking, private networks
are available across all the availability zones.

35
00:02:53,986 --> 00:03:00,307
In other words, when you request multiple servers
in different availability and sub-availability zones,

36
00:03:00,307 --> 00:03:03,062
these can all be placed in a single private network.

37
00:03:04,262 --> 00:03:09,299
IPC Provides the option for disaster recovery capable
deployments.

38
00:03:09,929 --> 00:03:15,437
This is based on active-passive infrastructure disaster
recovery, on the private network level.

39
00:03:16,067 --> 00:03:21,697
The feature synchronizes your workloads across two
physical data centers based on synchronous storage

40
00:03:21,707 --> 00:03:22,499
replication.

41
00:03:23,599 --> 00:03:29,249
Capacity is reserved in the failover data center
for your workloads to be restarted automatically in

42
00:03:29,259 --> 00:03:31,099
case of a data center outage.

43
00:03:32,229 --> 00:03:38,637
This option was built to accommodate legacy applications
that rely on infra-level architecture for resilience.

44
00:03:39,256 --> 00:03:45,127
The feature is not application-aware, therefore in
case of a data center outage, you are responsible

45
00:03:45,137 --> 00:03:49,824

to document relevant procedures to validate if your application is running properly.

46

00:03:50,954 --> 00:03:56,924

Please note that synchronous replication comes with a performance penalty on the storage level, systems

47

00:03:56,934 --> 00:04:02,699

requested with DR capability will have lower storage performance than systems without DR capability.

48

00:04:03,319 --> 00:04:08,909

When conducting performance tests on your acceptance environment, ensure that your production environment

49

00:04:08,959 --> 00:04:13,949

is set up similarly; either both environments with DR or both without DR.

50

00:04:16,350 --> 00:04:21,999

During the request, when you select the option of enabling the disaster recovery option, you will see

51

00:04:22,009 --> 00:04:24,237

that you can no longer select your data center.

52

00:04:24,837 --> 00:04:30,807

The primary location is selected automatically; It is based on which data center was selected as the

53

00:04:30,817 --> 00:04:34,200

primary data center during the creation of the private network.

54

00:04:36,100 --> 00:04:41,150

Here is an example of two Private networks created with different primary data centers:

55

00:04:43,480 --> 00:04:44,950

Private network A

56

00:04:45,212 --> 00:04:52,162

created where the primary data center is WPR: For any request of a DR-enabled server, the server will

57

00:04:52,172 --> 00:04:53,825

be deployed in WPR.

58

00:04:55,454 --> 00:04:56,944
Private network B

59

00:04:57,175 --> 00:05:03,785
created where the primary data center is DCR: For
any request of a DR-enabled server, the server will

60

00:05:03,794 --> 00:05:05,250
be deployed in DCR.

61

00:05:05,840 --> 00:05:11,720
In this case, you see that additional servers have
been deployed in both local zones using the same private

62

00:05:11,729 --> 00:05:12,324
network.

63

00:05:14,255 --> 00:05:17,874
What will happen to your workloads in case of a data
center outage?

64

00:05:20,204 --> 00:05:22,749
For the affected data center that is unavailable.

65

00:05:23,349 --> 00:05:28,525
In "this" example, the workloads in the local zone
WPR will not be available.

66

00:05:29,155 --> 00:05:35,685
Teams using servers in the local zones will need
to design their application to meet the availability-requirements

67

00:05:35,685 --> 00:05:40,387
for their application and have processes in place
for outage scenarios.

68

00:05:41,217 --> 00:05:47,912
For workloads of the stretched environment running
in WPR; The workloads will stop running in WPR.

69

00:05:48,522 --> 00:05:54,712
As the stretched zone uses storage replication, the
latest replicated state of DR-enabled servers will

70
00:05:54,722 --> 00:05:56,600
be available in WPR.

71
00:05:57,220 --> 00:06:04,230
The infrastructure DR functionality will start the
DR-enabled workloads in DCR based on the last synchronized

72
00:06:04,230 --> 00:06:04,662
data.

73
00:06:05,272 --> 00:06:11,382
Teams using these systems will need to have processes
described to validate if their application is up and

74
00:06:11,392 --> 00:06:17,437
running after a failover event as the infrastructure
DR functionality is not application aware.

75
00:06:19,936 --> 00:06:25,877
IPC provides functionality to move your workloads
from the primary data center to the failover data

76
00:06:25,877 --> 00:06:26,361
center.

77
00:06:26,981 --> 00:06:32,624
The functionality is the so-called Production Exercise
Disaster Recovery or PEDR.

78
00:06:33,424 --> 00:06:39,954
By using this option, DR enabled workloads can be
moved 'live' from the primary datacenter to the failover

79
00:06:39,954 --> 00:06:40,674
datacenter.

80
00:06:41,304 --> 00:06:44,624
This uses the VMware capability called vMotion.

81
00:06:45,254 --> 00:06:49,324
This will affect only the DR-enabled workloads in
your private network.

82
00:06:50,434 --> 00:06:53,074
The functionality can be triggered in two ways,

83
00:06:54,074 --> 00:07:00,194
1, the BTA triggers the operation on a private network level as a second-day operation

84
00:07:01,099 --> 00:07:08,269
2, ace Team members owning the servers, can run the PEDR operation from Cloud Atlas from the section Virtual

85
00:07:08,279 --> 00:07:09,449
Machines Management.

86
00:07:10,539 --> 00:07:15,361
In this case, a change ticket needs to be provided in Cloud Atlas to continue.

87
00:07:16,471 --> 00:07:22,261
Another difference here, is that Cloud Atlas expects you to have shut down the servers before you can continue.

88
00:07:22,891 --> 00:07:27,624
This means after running the operation, you also must power on the servers again.

89
00:07:28,224 --> 00:07:30,974
They will then be started on the failover data center.

90
00:07:32,894 --> 00:07:39,424
Be aware that Disaster recovery is not the same as High Availability, you will have to provide High Availability

91
00:07:39,513 --> 00:07:42,299
at a higher OSI layer in your application.

92
00:07:43,098 --> 00:07:49,449
The preferred approach for building applications in ING is to design the application in such a way that

93
00:07:49,459 --> 00:07:52,886
you are not completely dependent on the underlying infrastructure.

94
00:07:53,706 --> 00:07:59,646
You can achieve this by deploying multiple application servers in different availability zones and available

95
00:07:59,656 --> 00:08:04,811
failure domains such as the red and blue zone and
by using load balancers.

96
00:08:05,411 --> 00:08:12,149
You can use NGINX for load balancers inside a data
center to achieve higher availability for your application.

97
00:08:13,289 --> 00:08:20,969
For business continuity purposes, you should use the
data center locations WPR and DCR to build your environment

98
00:08:20,978 --> 00:08:25,723
in a way that it can overcome a partial outage or
complete outage of a data center.

99
00:08:26,823 --> 00:08:32,848
There is the possibility of using the Global Traffic
Manager for cross-datacenter DNS load balancing.

100
00:08:33,478 --> 00:08:39,869
This way you can build your active-active setup over
two data centers, or choose for the option of an active

101
00:08:39,939 --> 00:08:42,824
and hot standby setup over two data centers.

102
00:08:43,954 --> 00:08:50,444
This will give you flexibility when performing maintenance
tasks such as patching or upgrading, and it will give

103
00:08:50,454 --> 00:08:54,461
you much more control in case of an outage in the
underlying infrastructure.

104
00:08:56,372 --> 00:09:03,151
To come back to what the choice of using an IPC DR-enabled
server means; The option you select from the dropdown

105
00:09:03,151 --> 00:09:06,724
menu has the most effect on the IO performance of
your system.

106
00:09:07,544 --> 00:09:13,994
The infrastructure DR is a legacy option that writes
all data synchronously to the stretched zone in both

107
00:09:14,004 --> 00:09:14,936
data centers.

108
00:09:16,837 --> 00:09:19,387
The following will happen for any write action:

109
00:09:20,237 --> 00:09:23,699
First a write is made to the local array in microseconds.

110
00:09:25,029 --> 00:09:30,579
Then followed by a Write to the remote array over
the Wide Area Network (WAN), which will take about

111
00:09:30,579 --> 00:09:35,812
3 milliseconds, but is also dependent on network utilisation
during that time.

112
00:09:37,161 --> 00:09:40,387
After that it will return the result of the write
to the OS,

113
00:09:41,187 --> 00:09:46,599
In general, this means a latency for all writes of
between 5 and 10 milliseconds.

114
00:09:47,429 --> 00:09:54,249
For IO-intensive applications, especially applications
that do lots of small writes, this can be disastrous.

115
00:09:55,059 --> 00:09:58,199
Before enabling the infra-disaster recovery feature,

116
00:09:58,999 --> 00:10:01,924
Consider the IO needs of the application carefully,

117
00:10:02,774 --> 00:10:07,749
Ask yourself whether resiliency and availability can
be achieved in other ways.

118
00:10:08,539 --> 00:10:12,329
And consult with platform architects to discuss alternatives

119
00:10:14,242 --> 00:10:19,762
This concludes the IPC concepts of availability and

Sub availability zones topic.

120
00:10:20,592 --> 00:10:24,174
I hope you enjoyed and look forward to seeing you
in the next video.

Redbluereliab

0:03

Red and Blue are the different availability zones in each data center of ING.

0:10

They offer hardware failover and there are also different update and fault domains.

0:17

Update domains.

0:18

I mean, infra can update the hardware or let's say VMware without having a business impact.

0:28

Because think about it, if you order two machines in the IPC, do you actually have hardware failover?

0:37

Everything is virtualized.

0:38

Nowadays we have a software defined network software defined data center, meaning there's a management layer managing the hardware.

0:50

As an application you don't know on which hardware you're running.

0:55

Let me explain virtualization and the traditional architecture.

1:00

You have 3 servers, three physical servers that each having CPU and a memory chip.

1:08

An operating system was installed on each computer, server, middleware and an application.

1:14

But nowadays we have a hypervisor managing the hardware and hypervisor is a generic name and we use a vendor like VMware.

1:25

But there is already open source hypervisors or Microsoft Hyper V So now we have virtual machines and you go to the IPC and you click in the portal and in a couple of clicks you have a virtual server, a virtual machine and it gets its resources on demand from this virtualization layer.

1:51

Now, on demand, if one server virtual machine is not using the hardware, it can be given to another virtual machine.

2:04

But that might also mean that two virtual machines are using the same hardware.

2:12

There are many, many CPU and memory chips in the real physical machine in our data centers.

2:18

The whole point is that you don't know which hardware you're running.

2:24

So at ING we divided the hardware, you could say in each data centers in two colors.

2:32

If you go into a data center, you will see Ellie's hardware servers, and we divided the hardware in two colors, red and blue.

2:43

So you have hardware failover and this division in hardware we call availability zones Red, ING, Red and blue and they offer hardware failover and enables infra to do patching and hardware maintenance.

3:04

Actually in the IPC there is also a service called Baymass bare metal as a service that you can order a real physical or brake machine.

3:16

And this is for applications that have an extreme that have extreme non functional requirements that they need extreme CPU power or GPU for parallel processing.

3:29

But keep in mind there are less supporting services and there is less installed on these machines by default.

3:39

You have to do more work.

3:40

You have more responsibility.

Overloadprev

0:02

Let me explain why overload prevention is so important.

0:07

Your application is running in a shared environment.

0:10

What do I mean?

0:12

Your hardware is shared.

0:14

We talked before about virtualization and you don't have your own hardware.

0:20

Everything is software defined.

0:22

Nowadays.

0:23

The hardware is shared.

0:25

You're running in a multi tenant environment.

0:29

You're using probably a shared platform like ICHP or Day Bus or Key Space as a service and your service is being consumed by multiple consumers.

0:44

So if your service is overloaded with maybe a day Dos attack, a retry storm or excessive lookup queries and not only your service is affected, but multiple services and multiple platforms might be affected and this the customer might be affected.

1:06

And you can use several techniques for overload prevention.

1:10

With throttling you control the consumption of resources used by your service.

1:15

This can allow your service to continue to function even when an increase in the marketplace is an extreme load on your resources.

1:25

A service may throttle based on different metrics over time, such as the number of operations, for example maximum 20 requests per second or the amount of data, for example 2 gigabytes per minute.

1:40

And examples of throttling strategies are maybe rejecting requests from a consumer who's already accessed your APIs a number of times.

1:51

Or another totaling strategy might be to disable or degrade the functionality of selected non essential services.

2:00

And there is another technique that can be used that's called grade limiting.

2:05

Grade limiting is a technique used to control the number of requests a user can make to a service over the given period of time.

2:13

Rate limiting pattern can also help you to avoid or minimize throttling errors related to throttling limits.

2:22

So if you compare throttling and rate limiting, Throttling is implemented at the server side or network level.

2:30

Rate limiting is implemented at the user side or client level, and the main goal of throttling is that the API can handle the traffic it's receiving.

2:42

For rate limiting, it's to ensure a user doesn't make too many requests to your API or abuse your API.

2:53

So throttling is implemented by setting a limit on the number of requests that can be made to your API within a certain time period.

3:02

Rate limiting is implemented by setting a limit on the number and speed of request that a user can make within a specific time period and the result is with throttling, no more requests will be processed in this time period by your service and with rate limiting the result will be that no more requests will be processed until the time period expires.

3:29

And with overload prevention, don't forget your data sources.

3:34

Be aware of a sudden increase of the size of your data source or of excessive lookup queries on your data source on a shared platform like day bus or key space as a service.

3:49

This may impact the performance and quality of service of multiple services of multiple consumers.

3:57

So set a rate limit on the number of requests or the amount of data that a user or consumer can request within an interval of time.

4:07

For example, set a maximum for the number of rows a user or consumer can fetch and realize.

4:16

Implementing overload mechanisms are not without a risk.

4:21

It can have a direct impact on our customer or your consumer.

4:25

For example, the customer receives a message like please try again later or consumer API cannot retrieve its data.

4:36

Implementing overload mechanisms require a thorough investigation, Sealing tests a coordination in a service chain.

4:47

But implementing overload mechanisms is very important in a shared environment, and if you do it right, you can score quite some points for reliability.

Basicsreliab1

0:11

If you look at 2 systems, you have a secondary system and we sometimes call this clustering.

0:17

And please be careful with this term.

0:20

Depending on technology, a cluster can mean a lot of things.

0:24

But high level a cluster means I have multiple of the same and multiple identical processes or machines.

0:34

But we recognize two different flavors.

0:38

Horizontal clustering, vertical clustering.

0:42

With horizontal clustering we mean we have multiple machines, multiple hosts are running the same application.

0:51

That might be for hardware failover in the context of scaling.

0:56

If we're talking about horizontal scaling now, we're talking about adding an extra host, adding a virtual machine.

1:04

If we talk about vertical clustering, I'll deploy my application multiple times on the same host, or in the context of scaling.

1:14

If we talk about vertical scaling, we're talking about adding CPU memory to your machine, making it bigger and stronger.

1:24

So horizontal usually means extra machines, multiple virtual machines and vertical process failover.

1:35

I have multiple processes on the same machine and if a failure occurs, first the failure must be detected and then work in progress.

1:48

That must be managed.

1:49

If it's data in memory, probably it's lost.

1:53

Of course a failover must take place.

1:56

That's so monitoring must be in place, like black box monitoring or performing health checks.

2:04

There are many many monitoring tools out there depending on your application and the protocols used.

2:11

Now we are not taking a look at all the monitoring tools in this training, but worst case is when OPS becomes aware of a problem after the customer makes a call to the help desk.

2:24

You have to have active monitoring in place or post to passive monitoring.

2:29

That's vital for detecting a failure early and work in progress, as I just mentioned.

2:36

If it's data in memory that probably is lost, yes, we have solutions where we can copy data in memory, but you have to build that.

2:45

By default it's lost.

2:47

But for the request reply pattern, keep in mind that the requester must be able to resend the request, that the provider must be able to process the same request twice.

3:02

That's very important.

3:05

A microservice must be idempotent.

3:09

You say it must produce the same output if given the same inputs.

3:16

I'll repeat.

3:17

It must produce the same output if given the same inputs, produce the same outputs if given the same inputs.

3:27

That's not the same by the way, as at least once at least once delivery is sometimes demanded from middleware like messaging or tip code.

3:39

But your application must be able to handle the same output twice and no double processing and no double transactions.

3:50

Maybe you have to check build in a check in your application.

3:55

For this Eden potent is very important in the whole microservices concept.

4:02

First of all, you're resilient that you can replay any invocation without producing a new side effect.

4:11

That's what we say.

4:12

If an error occurs, we can simply retry that we can simply retry the request that you're able to do parallel processing.

4:21

You can process multiple requests in parallel.

4:25

If one of them fails, just retry for data consistency.

4:30

That's what I'm mentioning and no double transactions.

4:33

Let's say if you're using a database, in the end you're doing an insert, update, delete.

4:38

If you are idempotent, you do not create a second instance, it has exactly the same side effect.

4:47

So you are able to handle double requests or a double invocation.

4:54

This will guarantee data consistency that you don't have duplicate records or malformed data.

5:04

So if you're stateful when persistent data is in scope and you cannot accept data loss, you have to have extra measures.

5:15

And maybe before switching to the backup system that you have to make sure that the data is available on this backup system, you have to have some kind of data replication in place and is that singleness or any singleness.

5:30

So if your application is using transactional processing, a transactional processing, rollback, commit, then you do not lose data after last committed transaction.

5:47

You will lose the data in memory.

5:50

But if it's committed you're using a transactional.

5:54

If you're using transactional processing we say yeah you do not lose data after last committed transaction.

6:02

We'll talk more about the persistency and state for applications and middleware like XFA, messaging and database in a separate section.

6:15

So if your stateless, it's much much easier to be highly available.

6:23

So therefore I always try to separate your data from your application or application server.

6:31

Now let's say you get a.

6:33

You're processing a file, XFB that you get a file on your machine using the XFB flow, and you need to process the file.

6:44

Put that file separate, not on the same machine as your application.

6:51

We are saying create a connectivity node, a node with NFS on it, a shared file system.

6:59

The file comes in on this machine, on this NFS machine.

7:03

That's the one that you have to worry about for your data.

7:07

But your application is stateless.

7:09

It's on a separate machine with no data and you can simply make that high available.

7:15

High availability is much, much easier if you're stateless and if a failure is detected and a failover must take place, you must ask yourself how quickly can the backup system take over or how does the consumer handle this first failure?

7:35

With the first failure, maybe an error message was produced?

7:39

Does the application stop right away after the first error message or does it do a retry and a failover there?

7:48

Are the timeout settings correct?

7:51

Are they set correctly in your chain?

7:54

We'll take a look at the API chain later on, but is your backup system up to date?

8:01

If it's a cold standby system and it hasn't been up and running for months, let's say, how Are you sure that it's operational?

8:10

Can you handle the data loss if data in memory and you lose the data in memory, Are your consumers and providers idempotent?

8:20

That's a very good question.

8:23

So consider these scenarios.

8:26

I've made everything high available, I've made sure that the failover takes place, and I let's say in an ideal situation I do.

8:35

I don't lose any data.

8:38

So your primary system caches it, produces an error, but the secondary system takes over within milliseconds.

8:46

However, your consumer stopped after the first error.

8:51

That's an issue, right?

8:53

Or another scenario, let's say you're stateful, you're using a database, your database is high available.

9:00

You have maybe day bus, which we'll talk about later DAY BUS NORMAL.

9:05

The flavor normal is a data guard, It's a high available Oracle.

9:10

It's active will stand by you could say, but your first instance of the database it caches produces an error, but the secondary database takes over and the data is.

9:21

The transactions are let's say moved to the secondary system, the second database and you do not lose 1 transaction.

9:30

However, let's say you have a frontend application which is working with this database.

9:37

That the first database cache the first error and right away after the first error the application does not do a retry.

9:45

But right away after the first error it produces an error message on the screen of the end user.

9:53

What do you think the end user will do?

9:57

The end user will reenter his values and then again you have a double transaction.

10:05

Where am I getting here?

10:06

A getting at here we can make everything high available.

10:11

We can eliminate all single points of failure.

10:16

But if you do not test your non happy flow, I call it.

10:22

If you do not test your non happy flow, you're not highly available.

10:27

You can create a high available architecture, but you have to prove that you're high available.

10:34

We're still dependent on the application handling the failover.

10:41

The application must be able to handle the failover correctly.

10:46

So please trust me, if you do not test the normal happy flow, you never know you're high available.

Dr-Aa

0:05

Let's have a closer look at disaster recovery at ING and Active Active.

0:11

At ING, we have two data centers, DCR, WPR and Belgium, Kusha, Michel and Marnex.

0:18

Although they have to be closed down, they're too close to each other or it has to do with the metro line going underneath both of them.

0:29

But in the Netherlands we now have two data centers, DCR, WPR and for disaster recovery we saw

earlier you can have an active Active scenario, Active active architecture where both data centers are processing requests are handling load or an active passive scenario architecture where the second data center is just standing by doing nothing now just one data center that is processing the load.

1:02

In this active passive architecture we can use optionally the option Dr.

1:10

Capable.

1:11

If you order a machine at ING you get the question the option Dr.

1:17

Capable?

1:18

Yes.

1:18

No, we don't prefer this by the way.

1:22

We rather not want this and I'll explain later why.

1:27

But let's first take a look at Active Active.

1:31

So nowadays we deploy our stateless APIs.

1:34

Active Active, we have two in each data center, so both data centers are processing requests.

1:42

There are APIs deployed twice in each data center.

1:47

So we have in total 4 instances.

1:50

And by the way, we want to deploy all our APIs like this.

1:54

That's consistent.

1:55

That's easy.

1:57

I don't care about the A rating, maybe if it's a one or less, but a three, a four, We all deploy them 4 * 2 times in each data center.

2:07

But a Dr.

2:08

test is very simple that we don't.

2:11

We shut down the traffic to one data center.

2:14

Any other APIs which are already open and running and operational just take over the load.

2:22

So why, by the way, 2 instances per data center?

2:27

Why not one instance per data center, right?

2:30

If I have one instance per data center in total two, I'm already high available, aren't I?

2:39

Well, I think about the Dr.

2:42

test and not the actual plane hitting the data center, but we have to do a Dr.

2:49

test twice a year for one week.

2:53

I repeat, we have to do a Dr.

2:55
test twice a year for one week.

2:57
And do you want to be high available during that Dr.

3:01
test?

3:02
Let's say you want to do maintenance, you want to upgrade, you want to deploy a new version of your API during the Dr.

3:11
test.

3:12
If you just have a single instance, you're down.

3:15
You're not high available during the Dr.

3:18
test if you have just one instance.

3:21
So that's the reason why we deploy in total 4 instances, 2 per data center.

3:28
So during a Dr.

3:29
test we're still high available.

3:32
Now also keep in mind that high availability means in this case that if during a Dr.

3:40
test 1 instance crashes, so we just have one left, we are we must be still able to handle all the load, one instance must be able to handle all the load.

3:58

And I have to repeat this because I nowadays if I talk to sports, I hear many times that they've deployed 4 instances.

4:08

But as a matter of fact they're saying yeah, we need those two instances per data center to handle the load.

4:15

Well then you're not highly available because if during the art test that a complete data center is down, but you need 2 instances to handle the load.

4:26

If one instance goes down and you have only one left and you have an issue, it can't handle the load, you're not highly available.

4:37

Then you have to add instances.

4:39

Maybe you need 3 instances per data center, maybe 6 in total.

4:46

So keep in mind with a let's say standard architecture for APIs where you have 4 instances in total.

4:54

One instance must be able to handle all the load, so active, active.

5:02

The two two pattern I will want to be highly available during a Dr.

5:06

test.

5:07

We want to be able to do lifecycle management during a Dr.

5:11

test.

5:13

So why don't we do everything?

5:15

Why don't we have an active Active Architecture for everything?

5:20

The problem is the stateful applications, the stateful components.

5:24

It's about data integrity and data consistency.

5:28

Now, if I have persistent data in scope and one data center goes down and I have to switch to the other data center, I have to be sure that the data is In Sync.

5:42

The data from the first data center is also available in the second data center.

5:48

For a database, by the way, that's very difficult.

5:52

Let's say I want an active active database, so an Oracle.

5:57

Let's say an Oracle which is active in data center one and active in data center 2.

6:02

While being consistent, guaranteeing my integrity.

6:07

There can be no difference.

6:10

No bit can be different between the two instances.

6:13

That's very, very difficult.

6:15

Probably the performance won't be that great because it continuously needs to sync the data.

6:23

And is that synchronously or asynchronously?

6:28

That's what we're going to talk about in a couple of minutes.

6:31

By the way, by default Daybus offers an RPO recovery point of objective of 5 minutes at the moment a day bus normal.

6:44

That means you have a data guard configuration and there is a configuration also for disaster recovery, a data replication mechanism in place in day bus.

6:56

But still you have an RPO of 5 minutes.

7:00

We'll look into detail into the stateful components in another section of this training.

7:09

Then we'll take a look also at Cassandra for instance, or XFA messaging, Tipco AMS.

7:14

Kafka.

7:15

Well, Tipco AMS is contained, but we still have it of course, but we'll take a look at the stateful components.

7:22

But that's the problem with active active.

7:25

Our stateful components, usually they are not active active.

7:33

For stateful components we often use an active passive architecture.

7:38

But still you have to keep your data In Sync.

7:42

A couple of advices.

7:44

If possible solve this data replication in your application, Do it yourself, maybe copy the data, write it twice.

7:54

But don't depend on an infra mechanism or don't depend on the technology which is specific for one infra provider.

8:04

If you can solve it in your application, please do.

8:09

And another advice, always separate your data from your application.

8:14

Don't put a file or persistent data on the same machine as your application.

8:22

So if you're using XFA a file, don't send the file to the same machine as the application, but create a connectivity node, a separate machine with the file, and a separation from your application server.

8:38

The application server is stateless.

8:41

It's connected to this connectivity node using NFS.

8:45

And this connectivity node we will make high available in some way.

8:51

Maybe we copy the data ourselves or maybe we send the file to two machines to connectivity nodes.

9:00

But if we separate the data from our application, our application is stateless, our application server is stateless and maybe I'm still able to deploy my application server active active.

9:16

If you look at the two data centers and active, active, latency is something to think about.

9:23

Well, I have to make a reservation.

9:28

It's especially important for databases for API calls only.

9:33

With an extreme load with extreme Nfrs non functional requirements, this might be an issue.

9:41

Latency can be maybe 6 milliseconds.

9:45

That's already quite a lot, but don't take my word on it.

9:49

It's probably less, but for a database with a very high load it might, and I'll repeat myself, it might be an issue, but you have to do a performance test.

10:02

In most cases it's not a problem, but I don't want data center affinity.

10:09

Data center affinity means that you keep all the processes, all the processing in one data center.

10:17

It only makes things more complex.

10:20

It makes your Dr.

10:22

much much more difficult.

10:25

We prefer to create your application service stateless, separate your data and only the machine with your data is active passive.

Drcapable

0:05

So what is they are capable?

0:08

If you order a machine at ING you get the option Dr.

0:12

capable Yes or no.

0:16

Dr.

0:16

capable means data replication is configured by the infra provider.

0:23

So if I order my machine Dr.

0:27

capable, all the persistent data, the disks which are connected to my virtual machine is continuously copied to the other data center.

0:39

And by the way in the IPC this is a synchronous replication.

0:46

In many older environments like DCN and IIS solutions, those are network zones that we'll talk about later and the reapplication is asynchronous.

0:57

Now what's the difference?

0:59

A synchronous replication means I send the data to the other data center, but I wait for a confirmation, let's say from the disk that the disk in the other data center has to tell me, yes, the data has been written on disk.

1:15

Now you can go on processing so you know for sure the data has been written on the disk in the other data center.

1:24

Synchronous means I'll wait for a confirmation of the other data center that the data has written to disk.

1:32

This means I send the data to the other data center, but I right away continue processing.

1:39

I do not wait for confirmation for performance.

1:43

That's much better.

1:44

You're much faster.

1:45

You're not waiting for another data center that you do not have any latency issues, for instance, but you're not waiting for another data center.

1:55

Asynchronous is much faster, but with asynchronous I do not have a guarantee that the data is written on the other disk.

2:05

So with synchronous replication, my data loss is decreased.

2:13

My RPO recovery point of the objective is lower, It's better for data loss.

2:21

You could say well I mean you lose less data if replication is synchronous.

2:29

So remember in the IPC the data replication is synchronous, that's new opposed to your old environment.

2:39

So another thing, if they are test takes place, your virtual machine is restored in the other data center.

2:51

If you are, they are capable.

2:53

If you ordered your machine, they are capable.

2:56

The infra provider can restore your virtual machine in the other data center.

3:03

Your virtual machine is just a let's say.

3:06

If you take a snapshot it's just a file.

3:10

Because of virtualization, VMWare is now able to start your virtual machine in the other data center.

3:19

It will start up your machine with the same host name and in the IPC even with the same IP address.

3:27

Now if you are the er capable, your machine is restored with the same host name in the old environment with a different IP address by the way, but the IPC is using the latest technology, it's using a stretched fail on in this case.

3:45

And this means that even your IP address stays the same.

3:51

Again, this is only the case if you're Dr.

3:54

capable.

3:55

If you're active active, you already have a second machine in the other data center.

4:00

Of course.

4:01

It's a different host name, a different IP address.

4:04

As we saw with the APIs, you have four different machines, 4IN total, two in each data center, but you're already up and running active.

4:15

You don't have to restore anything.

4:18

If you're Dr.

4:19

capable, you're dependent on the infra provider to restore your machine in the other data center.

4:26

It will take more time.

4:29

And by the way, never use IP addresses in your application.

4:33

I'm sure you are aware of this.

4:36

IP addresses can always change.

4:38

Maybe you want to redeploy your machine, maybe you have a correct machine, whatever for any of your interfaces.

4:45

Always use a URL or a domain name.

4:49

Never use an IP address.

4:53

So why not to use the ER capable?

4:56

You're not in control during a Dr.

4:58

test that you depend on an infra provider.

5:01

If you're active, active, you are in control.

5:04

Maybe you can switch the load balancer or turn off the traffic to one data center.

5:09

You're not depending on the infra provider, so that's the first reason you're in control if you're active, active.

5:17

If you're Dr.

5:18

capable, you're not in control, you depend on the infra provider, it will take more time.

5:25

If you're Dr.

5:26

capable, your machine has to be restored.

5:28

Things have to be done by the infra provider, it just takes more time.

5:35

And don't forget, if you order your machine Dr.

5:37

capable with the synchronous reapplication, your machine is slower.

5:43

It can be let's say 20%.

5:47

So that's it.

5:49

Depends right?

5:50

So do a performance test.

5:52

But if you order your machine Dr.

5:54

capable, your machine will be slower, that's for sure.

5:59

So we don't want Dr.

6:01

capable anymore.

6:02

We want you to solve the data replication in your application.

6:08

Really only if you're not able to make the data high available yourself, then maybe choose Dr.

6:17

capable.

6:17

But first, go to your architect or BTA, discuss with them your problem, and discuss with them if you should order a machine with Dr.

6:29

Capable.

Tiktok

0:09

And quite recently enterprise architecture took some decisions about the future architecture of disaster recovery.

0:18

We want to move from a component based failover of our channels to a site based disaster recovery of our channels.

0:26

That's called TikTok.

0:28

We want to move to a Dr.

0:30

testing schedule that alternates the active site.

0:34

We want to move from an active active scenario to a scenario where one data center is active per month.

0:42

We switch each month.

0:44

The active site is a regular scheduled event.

0:48

The entire site is filled over in a controlled way and we want to run and switch this every month because we have learned that in a real active data center setup, there might be dependencies between the data centers.

1:07

And the whole point of disaster recovery is to have two completely independent data centers.

1:15

And that's what you achieve with this TikTok scenario.

1:21

Still, if you do it each month, you know for sure it's operational.

1:28

In the past we did it once or twice a year and then it might be a problem that is a secondary site is not operational, so update is forgotten.

1:38

That risk is minimized if you switch each month.

1:45

So in an active active scenario, the consumer sends its request to both data centers to all availability zone and in a TikTok scenario, the consumer sends its request to one data center.

1:59

The other data center is still up and running, it's hot standby you could say.

2:05

But all the load is going to one data center, one ING region and we switch every month TikTok to the other region to the other data center.

2:19

And for TPA, MARAC consumers and applications, there's already a feature available called the endpoint controller.

2:28

It uses service discovery, which we discussed earlier.

2:33

Now all consumers discover the endpoints from service discovery and all providers have registered their endpoints in service discovery.

2:45

But there is a feature where the providers can set their server status.

2:52

They can set their service in maintenance mode.

2:57

This means that the consumer which retrieves the endpoints from service discovery won't send any requests to the services in maintenance mode.

3:09

At the moment we have this active data center setup, but we want to move to this TikTok architecture.

3:18

We are not yet actively moving to this new TikTok architecture because it could have considerable impact and maybe it needs more preparation.

3:28

But if you're creating a new application and especially a TPA MARAC application, consider this new architecture if you are in control of your incoming traffic flows.

3:42

If you are in control which data center is active, you might already migrate to this TikTok architecture.

3:51

You could say this is a preferred local policy, a preferred local scenario.

3:59

With preferred local, I mean all traffic flows remain within one data center.

4:06

But knowing about this target architecture with TikTok, Please ensure you're high available in one data center and one zone, must be able to handle all load and for TPA, MARAC and sure you know how to use the endpoint control.

4:25

And here you see in a little bit more detail that this feature, this MARAC feature of the endpoint control, so you can put the operational mode in Service discovery from life to maintenance.

4:39

You can do this with the Service status override command, and the prerequisite is that the consumer and the provider must be using the Touchpoint automated framework and of course Service discovery.

HA-Requirements

0:06

So let's take a look at the requirements for high availability.

0:10

Then when you start designing your high availability, do realize high availability costs money.

0:19

Building a resilient infrastructure requires A considerable investment, so each design decision must be confirmed by a business requirement.

0:31

At ING, we have a process called the BIA, the Business Impact analysis, and in this business Impact analysis that you're together with the business and you're really asking the business, let's say if we lose data or let's say if this application is down for two hours, how much money will it cost ING?

0:57

That's really by the way, a justification for it to spend money.

1:03

But in the BIA you'll talk to the business and the business has to tell you what's the risk, what's the cost for ING if the service is down or what's the cost for ING if you lose 2 hours of transactions.

1:21

So one of the things that is coming from the BIA CIA rating, I hope you've heard of the Confidentiality, Integrity and Availability rating that those are international standards.

1:36

How to define your security in an IT system that the standard requirements and especially the A rating that A stands for Availability.

1:49

It's originally meant for the high availability within one data center.

1:57

We also have the RTO recovery time objective and RPO and not from Star Wars but RPO recovery point of the active and then also the MOT maximum outage time.

2:13

Those are all meant for disaster recovery and are also stated in the Bea.

2:21

However, unfortunately all those requirements are not set in stone.

2:27

Specifically the A rating it's not clear.

2:31

It depends on interpretation.

2:34

It depends on ORM, on your risk assessment.

2:38

So specifically for the A rating that consults your or ask your solution Act Architect or BDA, another example for APIs we want one consistent deployment.

2:52

I don't care it's a three or a four, I always want to deploy it 4 * 2 in each data center.

2:59

That's just another example.

3:02

Unfortunately the A rating, specifically the A rating is not set in stone.

3:08

The A rating formally we have also different documents, but let's say the A4 which is the highest rating and we have a document, this document stating the server may not be unavailable for less than or over 2 hours.

3:27

2 hours A service unavailable for two hours.

3:31

Well, that's a little bit old school right?

3:34

So we translate a four often to you need a second system and you need to be active, active.

3:46

So the A rating, unfortunately it's not set in stone.

3:50

The RTO RPO are already a little bit smarter as I mentioned these are meant for your disaster recovery.

3:59

Now although sometimes you get an RTO for component failover and an RTO for disaster recovery.

4:09

If we talk about component failover, that's for high availability within one data center.

4:16

But normally if we talk about RTO, RPO and they're not mentioning component failover, it's meant for your disaster recovery.

4:26

So the RTO is the time needed to restore a business process.

4:33

How quickly are you up and running again?

4:36

If a plane hits one data center, how quickly are the application up and running again?

4:42

The RPO is about your data.

4:46

If the plane hits the data center, how much data will you lose?

4:53

This is about data replication or backups.

4:57

How often do you make a backup?

5:00

Let's say you only depend on a backup, you are using a database and you do backup once a day.

5:08

Potentially you can lose 23 hours and 59 minutes, 59 seconds, etc.

5:16

That potentially you can use loose almost 24 hours of data at the maximum outage time is about you could say the decision at the time management needs to take the decision to take ADR to do a failover.

5:41

The maximum outage time is about the time management needs to take the decision to execute a Dr.

5:51

because realize in reality and we have fortunately I have to knock on the table but it doesn't the complete data center went down that's that's has not happened or a plane hit one of our data centers but the air conditioning went out some time ago or we had an electricity failure.

6:14

Now what happens do you think that management hears about this failure, this electricity failure do you think that either way say okay switch to the other data center they wait they wait and hope that the error in the electricity that can be fixed or the error in the air conditioning can be fixed.

6:37

How long can they wait?

6:40

Because the RTO, the recovery time objective, this counter it's a time value.

6:46

Let's say RTO is 2 hours.

6:49

So your service needs to be up and running within two hours.

6:52

This RTO value starts counting the moment management that the business tells it okay.

7:01

Now you need to recover.

7:03

Now start the Dr.

7:06

at the moment the business says now do Dr.

7:09

Then the counter starts, starts counting.

7:13

It's not the moment the disaster strikes.

7:17

The RTO starts counting the moment the business, our chief tells us do the Dr.

7:26

So maximum outage time.

7:28

That's the real, you could say realistic value that we're down.

7:33

That's the RTO plus the time spent by management to take the decision to execute the Dr.

7:42

That's about the maximum outage time.

Netwbasicb 1

0:03

Let's discuss some network basics. The most commonly used type of network today is a TCP IP network. It means the data is transferred over the network in small packets, and each packet contains a source IP address and a destination IP address. And we talk about network zones. A network zone is a group of computers or nodes.

0:29

That are connected with a cable that was in the past along a logical area network. They all this group of computers had to be connected to one physical switch and now we have Vlans virtual ones. With

software we can create local area networks and in the IPC a network zone or a network segment is called a private network.

0:58

And an important characteristic of a network zone private network is that each network zone has a firewall, and a firewall can filter the traffic, can filter the incoming traffic, and it can filter the outgoing traffic. But the traffic within the network zone, the traffic within a private network is not filtered.

1:23

And a model often used for the network architecture is the perimeter model or the castle wall model. It means we have a special network zone, which is the full door to the outside world, to the bad, bad Internet. That's called the DMZ, the Demilitarized Zone. It's a special secured network zone which connects with the outside world.

1:52

All external facing applications that ING must connect through this zone. I think about the API gateway and the external API gateway or the gateway if you want to do a file transfer with an external party. Or maybe you know the rebob proxies. But the machines and servers in this DMZ are hardened and meaning these machines are protected with special measures.

2:22

Like security monitoring tools and intrusion detection tools etc. And the formal definition of an external connection at ING is a connection between a non ING governed IT environment. So an external party and an ING internal network that's considered external facing at this traffic must go through the DM set.

2:49

And at ING we call the DMZ. We have some special network zones called the EAGN and Iron and actually connections from the ING internal network to our DMZ is also considered external, facing the traffic coming from Internet access zones.

3:12

Offices ING Internet and the ING user on are not considered external facing. Let me explain why we consider an API external facing even though it's behind an API gateway. The external API gateway the session as started by the user or the malicious user. You could say that the session exists with your API, not with the gateway.

3:42

That API gateway only does authentication, does stuff with the HTTP header, but the HTTP body is passed through and the session between the user exists with your API, not with the gateway. And that's why there is a considerable risk and therefore we call this external facing.

4:08

Then if we map the ING network zones to the perimeter model, you'll see that in the DM set, we have EAGN and I'm and I'll give you a definition after this slide of EAGN and I. And in the trusted and

restricted zones, we have our business zones, what we call business zones, a business zone that is a zone where the squats can deploy their applications and what we call in the IPC, a private network.

4:38

Or in ichp A namespace and a management zone is what we call the MZ at ING and we have an s s set supporting services zone. Some examples in the supporting services zone. Usually the infra provider installs services that must be available for all business zones like Active Directory or DNS.

5:08

And the management zone, think about tools like Omnibus or Nasas and the complete ING network, we also call DCN and quickly some definitions that a management zone host solutions aimed at managing the IT state, for example for the purpose of vulnerability scanning the supporting services zone, host servers and appliances to support ING applications.

5:38

As mentioned, maybe Active Directory, the elder DNS. Every application hosted in a bzet in a business zone should be able to establish a connection with the SSZ. The firewall is default open for the business zones and the EAGN. The external application gateway network offers an.

6:02

Intermediary function between the internal and external network. Typically reverse courses like the external API gateway and you need an air cap to open these firewalls in the DMZ like EAGN and I'm the external interface network is used to connect to known business partners using VPN or leased line.

6:27

But known and means Ings on contractual arrangement in place to guarantee good behavior. And also for you need an Aircag. So a business zone is any zone where they can deploy an application. And in the IPC a business zone is called a private network and it's managed by the self-service portal. And maybe you've heard the term user long. The user long is an access network.

6:56

Which is installed at an ING owned or rented location. Think of an office, branch, office, point of sale or an ATM. And we have different access networks depending on the connection type we see cross or long or per location. So remember an important characteristic of a network zone private network is that it has its own firewall and a firewall acts as a gatekeeper.

7:26

It checks source IP address, port, destination IP address, port and protocol and it checks per network zone within your private network. There is no filtering within the private network, all traffic is allowed. Only the incoming and outgoing traffic per network zone is filtered and the firewall is in the default deny policy. Default all packets are dropped.

7:56

And all traffic is disallowed. You need a firewall rule to allow traffic and you can create firewall rules in

the self-service portal within the IPC. We are using application aware firewalls or also called next Gen. firewalls. The firewall that I described before was a second generation firewall, a next Gen. firewall.

8:24

Can add context, can decrypt, encrypt. It uses deep packet inspection. But what do I mean with application aware? Let me give you an example. In a second generation firewall, if I want to allow Oracle traffic, I had to open port or create a rule for port 1521.

8:47

In the next Gen. firewall you have to open the port for Oracle. It will recognize the protocol, it will recognize Oracle traffic. Very important, it won't be fooled if a different port is used, It's much more secure. So when you are in the war room, the question is not only do you know your traffic flows, do you know which firewalls are involved?

9:16

Which firewall rules are needed if you have a traffic flow from 1:00 private network to the other private network? How many firewall rules are involved that you need? An outgoing rule and an incoming rule? And if there's a proxy or a load balancing in between, you even need more rules. Please realize this.

9:42

So even though in the self-service portal it might be one click or one request, but multiple firewalls or multiple firewalls might be created. And one more thing, if you're encountering connectivity issues, if you're communicating with network engineers, always mention the source IP address, destination IP address, the port, and your network names.

1

00:00:11,499 --> 00:00:12,164
Hello,

2

00:00:12,954 --> 00:00:19,724
In this module, we'll briefly go over the options
you have with VM based windows workloads from an infrastructure

3

00:00:19,734 --> 00:00:20,363
perspective.

4

00:00:20,983 --> 00:00:27,714
You can use this information to order, monitor and
use MSSQL to make your application as reliable as

5

00:00:27,723 --> 00:00:28,389
required.

6

00:00:29,218 --> 00:00:35,068

This module will not go into full details of everything offered, but a slide deck with more information is

7

00:00:35,078 --> 00:00:40,951
provided, together with important links to Windows documentation where you can find the latest information.

8

00:00:41,741 --> 00:00:46,226
It is assumed you have knowledge about sub-availability zones within IPC.

9

00:00:46,816 --> 00:00:53,513
If this is not the case, please follow the "IPC concepts training" first, before you start with this course.

10

00:00:55,424 --> 00:00:59,226
Before you order a windows VM you should start with a design.

11

00:00:59,806 --> 00:01:05,676
In your design you should make a decision if you want to use the DR capable service or go for the preferred

12

00:01:05,686 --> 00:01:10,564
method of a High available design with either a local or global traffic manager.

13

00:01:11,374 --> 00:01:18,363
As we saw in the previous training, IPC concepts, the DR capable service is not suitable for applications

14

00:01:18,374 --> 00:01:19,714
with high IO needs.

15

00:01:20,514 --> 00:01:24,214
The preferred way is to make use of the sub availability zones.

16

00:01:24,863 --> 00:01:30,704
Spread your workloads and use your traffic manager to set up either an active-active or active-passive

17

00:01:30,704 --> 00:01:31,626
configuration.

18

00:01:32,446 --> 00:01:38,896

An extra advantage is that this gives you extra options when you perform patches or release new versions of

19
00:01:38,906 --> 00:01:41,164
your application to minimize downtime.

20
00:01:42,473 --> 00:01:48,653
To minimize latency, it is prudent to make sure that your database and primary workload are in the same

21
00:01:48,653 --> 00:01:49,351
datacentre.

22
00:01:51,291 --> 00:01:56,001
SCOM is used to perform the monitoring, but it's not mandatory to have this enabled.

23
00:01:56,631 --> 00:02:00,276
This is a step you have to complete in ServiceNow to enable this.

24
00:02:01,086 --> 00:02:03,393
Find your CI in service now,

25
00:02:03,503 --> 00:02:05,306
go to the monitoring tab,

26
00:02:05,445 --> 00:02:11,226
check the monitoring and accepted checkboxes and choose GMCR in the "monitored by" field.

27
00:02:12,526 --> 00:02:19,051
If you don't want to use omnibus but use IAT instead, check the disable auto-ticketing events box.

28
00:02:20,371 --> 00:02:25,013
Make sure you onboard your application to IAT if you choose this option.

29
00:02:25,013 --> 00:02:32,113
Squaredup can be used to create your own dashboards and to view the SCOM alerts, onboarding to squaredup

30
00:02:32,113 --> 00:02:37,276
is done automatically on P-code level, but you need to onboard each server separately.

31
00:02:37,876 --> 00:02:42,876
You can verify if onboarding is successful by triggering
the maintenance mode for squaredup.

32
00:02:44,796 --> 00:02:49,651
Backup for virtual servers is provided in 2 service
levels, Gold and Platinum.

33
00:02:50,241 --> 00:02:56,513
It is an optional feature that can be enabled during
the request of your server or as a second-day operation.

34
00:02:57,333 --> 00:03:02,188
Backup for virtual servers is provided in 2 service
levels, Gold and Platinum.

35
00:03:02,778 --> 00:03:09,051
It is an optional feature that can be enabled during
the request of your server or as a second-day operation.

36
00:03:09,871 --> 00:03:15,150
Backups are machine-dependent, you cannot restore
to a different location than the original server.

37
00:03:15,960 --> 00:03:18,525
The following applies to both service levels:

38
00:03:19,985 --> 00:03:24,655
¬ΣThe option of selecting a retention period of 14
days or 31 days

39
00:03:26,078 --> 00:03:28,725
¬ΣThe option of File Level or Image level backup;

40
00:03:29,556 --> 00:03:35,456
Please note that image level backup also means an
Image level restore, which requires a restore of your

41
00:03:35,466 --> 00:03:36,176
whole server.

42
00:03:36,806 --> 00:03:40,250
This requires downtime of your server during restoration.

43

00:03:41,670 --> 00:03:44,241
→ You can set a time for the backup to start daily

44
00:03:45,563 --> 00:03:50,513
→ You can suspend or resume the daily backup if required
for a specific service window

45
00:03:51,298 --> 00:03:57,268
Similar to Linux based virtual servers, the difference
between the backup service levels is that Platinum

46
00:03:57,278 --> 00:04:00,125
creates a secondary copy in the second Data Centre.

47
00:04:00,755 --> 00:04:07,005
Please note that Platinum is only provided for servers
requested with the infrastructure DR capability which

48
00:04:07,015 --> 00:04:08,985
runs on the 'stretched environment'.

49
00:04:09,538 --> 00:04:15,238
In other words, Platinum can only be used when the
VM can run on both data centers.

50
00:04:17,169 --> 00:04:21,764
Following Microsoft, a monthly patch cycle is offered
using the ring model.

51
00:04:23,084 --> 00:04:27,876
Using SCOM portal you can verify in which maintenance
window your server is placed.

52
00:04:28,466 --> 00:04:33,776
It's possible to change this window using the Set-MaintenanceWindow.ps1
script.

53
00:04:34,406 --> 00:04:38,114
Please note that it can take up to 8 hours before
changes are set.

54
00:04:40,004 --> 00:04:44,351
If we look at DR capable deployed VMs, you can perform
a PEDR.

55
00:04:44,931 --> 00:04:50,661

Your active machine will be v-motioned to the other datacentre and will commence running from there, using

56
00:04:50,671 --> 00:04:53,064
the same IP address and machine name.

57
00:04:53,683 --> 00:04:58,239
Please make sure all application services are running after the exercise.

58
00:04:58,839 --> 00:05:01,914
You can perform the PEDR via cloud atlas.

59
00:05:02,734 --> 00:05:08,914
Note that you cannot select one machine, All VMs in the selected private network that are DR capable,

60
00:05:08,914 --> 00:05:11,401
will be moved as part of the PEDR action.

61
00:05:13,311 --> 00:05:17,375
In case of an actual disaster, the switchover happens automatically.

62
00:05:18,175 --> 00:05:22,838
If you've chosen for the High available setup you will have to perform the test yourself.

63
00:05:23,438 --> 00:05:26,950
You can do this by cutting off a data stream in your traffic manager.

64
00:05:27,560 --> 00:05:32,351
It would be interesting to see how the application handles running sessions in switchover

65
00:05:34,244 --> 00:05:36,739
This concludes the VM's Windows topic.

66
00:05:37,369 --> 00:05:41,026
I hope you enjoyed and I look forward to seeing you in the next video!

1
00:00:11,499 --> 00:00:12,164
Hello,

2
00:00:12,964 --> 00:00:19,574
In this module we'll briefly go over the options you have with VM-based Linux workloads, from an infrastructure

3
00:00:19,583 --> 00:00:20,238
perspective.

4
00:00:20,838 --> 00:00:27,088
You can use this information to order, monitor and use Linux servers to make your application as reliable

5
00:00:27,139 --> 00:00:27,976
as required.

6
00:00:28,806 --> 00:00:34,656
This module will not go into full details of everything offered, but a slide deck with more information is

7
00:00:34,666 --> 00:00:40,451
provided, together with important links to Linux documentation where you can find the latest information.

8
00:00:41,241 --> 00:00:45,726
It is assumed you have knowledge about sub-availability zones within IPC.

9
00:00:46,316 --> 00:00:52,986
If this is not the case, please follow the "IPC concepts training" first before you start with this course

10
00:00:54,824 --> 00:00:58,576
Before you order a Linux VM you should start with a design.

11
00:00:59,156 --> 00:01:05,026
In your design you should make a decision if you want to use the DR capable service or go for the preferred

12
00:01:05,036 --> 00:01:09,914
method of a High available design with either a local or global traffic manager.

13
00:01:10,723 --> 00:01:17,713
As we saw in the previous training, IPC concepts, the DR capable service is not suitable for applications

14
00:01:17,723 --> 00:01:19,063
with high IO needs.

15
00:01:20,964 --> 00:01:24,664
The preferred way is to make use of the sub availability zones.

16
00:01:25,314 --> 00:01:31,154
Spread your workloads and use your traffic manager to set up either an active-active or active-passive

17
00:01:31,154 --> 00:01:32,076
configuration.

18
00:01:32,896 --> 00:01:39,346
An extra advantage is that this gives you extra options when you perform patches or release new versions of

19
00:01:39,356 --> 00:01:41,614
your application to minimize downtime.

20
00:01:42,424 --> 00:01:48,604
To minimize latency, it is prudent to make sure that your database and primary workload are in the same

21
00:01:48,604 --> 00:01:49,301
datacentre.

22
00:01:51,201 --> 00:01:55,788
When a machine is deployed, it has a standard connection for performance monitoring.

23
00:01:56,378 --> 00:02:02,026
But it is still up to you to connect it to either your own solution or to global monitoring solutions.

24
00:02:02,835 --> 00:02:08,705
The monitoring data will be exposed in Prometheus format, and you can scrape that data and feed it into

25
00:02:08,715 --> 00:02:10,876
your monitoring solution of your choice.

26
00:02:11,715 --> 00:02:15,275

For logfiles there is no standard solution out of the box.

27

00:02:15,885 --> 00:02:21,845

A few tooling options are made available via cloud atlas which you can use to connect to your own logging

28

00:02:21,855 --> 00:02:25,050

monitoring solution or a global monitoring solution.

29

00:02:26,971 --> 00:02:31,826

Backup for virtual servers is provided in 2 service levels, Gold and Platinum.

30

00:02:32,416 --> 00:02:38,688

It is an optional feature that can be enabled during the request of your server or as a second-day operation.

31

00:02:39,508 --> 00:02:44,788

Backups are machine-dependent, you cannot restore to a different location than the original server.

32

00:02:45,598 --> 00:02:48,163

The following applies to both service levels:

33

00:02:48,983 --> 00:02:53,873

- The option of selecting a retention period of 14 days or 31 days

34

00:02:54,683 --> 00:02:57,588

- The option of File Level or Image level backup;

35

00:02:58,418 --> 00:03:04,318

- Please note that image level backup also means an Image level restore, which requires a restore of your

36

00:03:04,328 --> 00:03:05,038

whole server.

37

00:03:05,668 --> 00:03:09,113

This requires downtime of your server during restoration.

38

00:03:09,913 --> 00:03:12,743

- You can set a time for the backup to start daily

39

00:03:13,451 --> 00:03:18,601
- You can suspend or resume the daily backup if required
for a specific service window

40
00:03:19,351 --> 00:03:25,481
Similar to Windows based virtual servers, the difference
between the backup service levels is that Platinum

41
00:03:25,491 --> 00:03:28,326
creates a secondary copy in the second Data Centre.

42
00:03:28,956 --> 00:03:35,205
Please note that Platinum is only provided for servers
requested with the infrastructure DR capability which

43
00:03:35,216 --> 00:03:37,186
runs on the 'stretched environment'.

44
00:03:37,738 --> 00:03:43,438
In other words, Platinum can only be used when the
VM can run on both data centers.

45
00:03:45,339 --> 00:03:49,226
You can use our cloud atlas portal to perform your
infra patches.

46
00:03:49,836 --> 00:03:55,316
It goes without saying that you should start with
your Dev environment and move your way up to production,

47
00:03:55,316 --> 00:04:00,686
using testing and change management due diligence
to confirm the patch does not disrupt your running

48
00:04:00,696 --> 00:04:01,576
workloads.

49
00:04:02,376 --> 00:04:08,586
If you have chosen the high availability configuration,
you can use the features in cloud atlas to patch all

50
00:04:08,596 --> 00:04:11,151
machines in a given sub availability zone.

51
00:04:11,741 --> 00:04:17,191
Divert traffic away from the chosen zone in your

traffic manager first before you perform the patch

52

00:04:17,201 --> 00:04:18,326
for minimal impact.

53

00:04:20,236 --> 00:04:27,138
If we look at DR capable deployed VMs, you can perform
a Production Exercise Disaster Recovery (PEDR) .

54

00:04:27,718 --> 00:04:33,448
Your active machine will be v-moted to the other
datacentre and will commence running from there, using

55

00:04:33,458 --> 00:04:35,850
the same IP address and machine name.

56

00:04:36,471 --> 00:04:41,026
Please make sure all application services are running
after the exercise.

57

00:04:41,626 --> 00:04:44,701
You can perform the PEDR via cloud atlas.

58

00:04:45,521 --> 00:04:51,700
Note that you cannot select one machine, All VMs in
the selected private network that are DR capable,

59

00:04:51,700 --> 00:04:54,188
will be moved as part of the PEDR action.

60

00:04:56,099 --> 00:05:00,163
In case of an actual disaster, the switchover happens
automatically.

61

00:05:00,963 --> 00:05:05,676
If you've chosen for the High available setup, you
will have to perform the test yourself.

62

00:05:06,276 --> 00:05:09,789
You can do this by cutting off a data stream in your
traffic manager.

63

00:05:10,398 --> 00:05:15,151
It would be interesting to see how the application
handles running sessions in switchover.

64
00:05:17,081 --> 00:05:19,563
This concludes the VMs linux topic.

65
00:05:20,193 --> 00:05:23,851
I hope you enjoyed and I look forward to seeing you
in the next video!

Datadesign 1

0:05
Let's now have a closer look at our data and the data services available at ING and more specifically our persistent data as stored on a disk or managed by a database management system like Oracle Day Bus.

0:21
A non persistent data.

0:23
Data in memory is usually lost.

0:27
But keep in mind your data is your most critical asset, it's our crown jewels.

0:33
An application is just a means to access and manage data and data has weight, data has value, data has importance, data has a confidentiality, integrity and availability rating.

0:49
And data has a lifecycle and to visualize this weight of the data, here are some examples of AWS, a snowball or snowmobile.

1:01
That's a truck where you can store 100 petabyte of data.

1:07
But the size of your data is important for performance.

1:11
But also think of latency.

1:14
It's an important choice for your data store which data service you are going to use.

1:21

At ING we have day bus, SQL Server, Cassandra or key space.

1:26

As a server we can use a network attached storage and thus or maybe the ECSS THREE object store and that ING.

1:36

The value of our data is expressed in the CIA rating and the RPO.

1:41

The recovery point objective we have discussed earlier and confidentiality is about access to the data.

1:49

The data must be accessed only by authorized individuals or systems, and the integrity of the data is about can you trust your data?

1:59

Is it accurate?

2:00

Is it consistent over its entire life cycle?

2:05

Can it not be modified undetectably?

2:08

And availability of your data is of course about your data being available when it's needed.

2:15

Do you have backups or do you have a replication mechanism?

2:21

So to store business data at ING we have several technologies.

2:25

We have day bus database as a service and SQL Server both database management systems.

2:33

We have key space as a service, that's a no SQL data store.

2:38

We have the S3 object store or we have the shared file system, not.

2:46

And there are more services available in the IPC, but these are the preferred services and otherwise please discuss with your architect or BTA of your domain.

2:57

So which data store should you use?

3:01

But first let's talk a little bit more about database management systems like Oracle and SQL Server.

3:08

They offer some special features like keys and indexes to retrieve data.

3:14

You can create relations between schemas and databases.

3:20

But more important, it offers exclusive locking and support for transactions, rollback, commit.

3:29

And it supports SQL Structured Query Language.

3:33

Let me explain transactions in the context of a database, and I'm sure most of you know.

3:39

But let's say I have to do a money transfer from account A to B and this is an example of the happy flow.

3:46

So first I update account A, second I update account B.

3:52

And by the way, when you use an exclusive lock you can make sure that nobody can access account A or B during this update.

4:02

But let's now have a look at the non happy flow.

4:06

The same money transfer.

4:07

We have to update account A, we have to update account B.

4:12

So I update account A and then something goes wrong.

4:17

My application crashes or a network failure with the connectivity today bus or whatever, something goes wrong.

4:27

If you have updated account A and not updated account B, you have an integrity issue.

4:34

You didn't do the complete money transfer with the relational database management system like Oracle or SQL Server and using a transaction, the database will roll back the change.

4:50

It will revert the change or it will set the data to the original state.

4:59

That's ensuring integrity and now you did not do half a money transfer let's say and in your application you can update account A, update account B and then program a commit statement.

5:17

So during the updates it's locked and nobody can access it.

5:22

And only if you have done update A and B you give a commit and now the data is really changed in your database.

5:33

This guarantees integrity.

5:36

So depending on your data, these features of a database are quite important for the choice of your data service.

5:46

In a no SQL data store like Cassandra it's much more difficult to guarantee the integrity and consistency of the data.

5:56

So at ING, we have a rule a system of record should only be stored in a database like Oracle or SQL Server because of this transactionality.

6:10

A rollback commit.

6:12

That's one of the reasons.

6:14

A system of record means it's the authoritative data source for a given data element.

6:23

It's the master data, it represents the truth.

6:26

We sometimes use Cassandra like a cache.

6:29

It's a copy of the source, it's a copy of the original data.

6:35

If I lose all data in Cassandra, I can always rebuild the data in Cassandra from the original source from the database.

6:45

Another important design principle, also for resilience and high availability is that a data source or a data store can only be used by one application or at ING by one purpose.

7:01

Picode we consider a database tightly coupled to one application.

7:08

Consider this.

7:10

If you have a data store or a database that is used by multiple applications, multiple squats, so multiple picodes purposes and you want to change the data structure, have to coordinate with all other applications and squats.

7:29

If you have to change the technology you want to move from day bus to SQL Server or from Cassandra to graph or whatever.

7:39

You have to coordinate with all involved applications and squats.

7:45

Who's responsible for this data store?

7:48

The ownership might not be clear.

7:51

And what about the load?

7:53

Do you manage coordinate the load?

7:56

If you have many multiple applications and squat using this data store, it's very difficult to control the performance and load on this data store.

8:08

So this dependency, if you have multiple purposes and P codes using this data store, this dependency hampers our changeability or you could say our agile way of work.

8:24

So let's look at a very simple example.

8:27

I have a database with my customer data and a lot of applications need customer data, right?

8:35

So the easiest way and the quickest way is all applications access this database.

8:42

Your PO is happy, you can do it in one Sprint.

8:46

But now you have multiple applications, squads, areas all dependent on this database.

8:54

How do you control the load?

8:56

Again, if the data structure changes, you have to coordinate with multiple parties.

9:02

So what's the solution?

9:05

Put a service in between?

9:07

An API in between.

9:10

Now you have one service which is tightly coupled to this database or data store.

9:17

1 application needs to be changed if the data changes, if the technology changes.

9:24

By the way this is an example with a database, but if you have a course application with a special interface or whatever, this is also the solution.

9:34

Don't depend on this specific technology or data store.

9:42

Put a service in between.

9:44

That's what we consider loosely coupled.

9:49

So if you want to ensure data integrity and data consistency, use a database, use day bus or SQL Server always on high available and a data store can only be used by one purpose by 1 picode.

Dataha1c-1

0:03

Hi and welcome back.

0:05

Let's now have a look at data high availability.

0:09

We'll start with the essentials and we'll talk about the difference between availability and integrity of your data.

0:16

We'll look at several backup considerations like which backup scheme to use and what about the performance of your backup and restore.

0:26

We will talk about RPO recovery point objective, the most important requirement for your data and why it is so difficult to have zero data loss.

0:37

We'll discuss in detail stateless and stateful and the concept of separating or isolating your data from your application or application server, and then we'll have a look at the different data technologies that are available within the ING or IPC.

0:56

To be specific, first we'll have a look at Convolt, that's the standard backup tool available.

1:03

We'll look at the databases, the relational database management systems like day, BASS and SQL Server and we will talk about a no SQL data store, CAS key space as a service, which is actually Cassandra.

1:19

And we'll talk about cool stuff like the ECSS 3 OPX store, that's technology which is also available in the public cloud.

1:28

And then we're going to talk about Nas network attached storage, which we can use as a shared file system.

1:35

And last but not least, we'll talk about archiving, which has some special characteristics like warm write once, read many.

1:46

When we think about high availability we tend to focus on the application and middleware.

1:52

But don't forget, if you lose your data, it's all over an application.

1:57

I can redeploy, reinstall an application is just a means to get to the data to manage your data.

2:06

And if you think about data resilience, you have to think about availability, backups, copying your data.

2:13

But also, is your data correct, Trustworthy.

2:17

That's data integrity.

2:20

Maybe one of the worst things that can happen is that your data is correct.

2:24

Your data is not correct.

2:26

For instance, I know from experience some years ago at another company, the business came to me and told me all birth dates in the database were set to 010101 all birth dates.

2:41

That's not good news, let me tell you.

2:43

So we restored a backup from the previous day.

2:47

What do you think?

2:48

It was already 010101.

2:52

So you have to think about how long do you keep your backups?

2:57

How many backups do I create, what's the retention.

3:02

And by the way, the requirements for the integrity of your data is the I in the CIA rating, and maybe it's 3333, CIA rating 333, but the I in the CIA rating is the requirement for the integrity of your data.

3:20

And in this context of consistency and integrity of your data, some might use the term logical failure, meaning there's a logical error, your data is not correct anymore.

3:33

Like the example where all birth dates were 010101 and a technical failure means and the middleware crashes.

3:42

Maybe an error in Oracle or in the hardware.

3:47

So for data integrity is very important that you have good input validation, maybe you have to write your own integrity check, a program that regularly checks the integrity of your data.

3:59

You need good logging and who did what when and who was the user, what transaction was *** doing, which data was entered.

4:10

You must be able to relate the input law to the transaction in the database.

4:17

The RPO or recovery point objective is the most important requirement for data high availability and the most obvious way to have data high availability is to create a backup to copy your data to a second device.

4:32

And we have different backup schemes and depending on technology we use different terminology.

4:38

But first of all, you have a full backup where you back up all your data to a second device.

4:45

Then we have an incremental backup where you back up only the changed data since the last backup.

4:53

And this last backup can also be an incremental backup.

4:57

And then you have a differential backup where you back up all the changed data since the last full backup, even when there's an incremental backup in between.

5:07

You always backup all changed data since the last full backup and then we have continuous backup or real time backup.

5:19

Some technologies like Oracle offer a continuous backup.

5:26

An important factor to consider is the performance of your backup or restore.

5:32

How much time does it take to create your backup or to restore your backup?

5:37

Is the performance affected during the backup and the time that is needed to restore the backup?

5:44

Probably means downtime.

5:47

Affects your RTO, your recovery time objective.

5:51

When your data is not available, your service is not available, so that's downtime.

5:57

It depends on the size of your data, on which technology is used, which backup scheme, and is the backup or copy synchronous or asynchronous.

6:09

Another important reason to have backups is of course ransomware.

6:13

Ransomware attacks are now over 50% of all malware attacks.

6:18

We just had an attack on the oil pipelines in the United States.

6:23

I don't know if you read about it, but the Dark Side group as they are called, they offer ransomware as a service.

6:31

Ransomware big business.

6:35

Creating a backup can mitigate the risk of ransomware, but be aware, put that backup in a different network segment.

6:43

You need network segmentation and this subject Network segmentation we will discuss in detail in a separate part of this training about networking and firewalls.

6:55

But network segmentation is in place by default for com fault and day bus.

7:03

Zero data loss or RPO 0 is very hard to guarantee even our most critical and advanced data store, day bus Oracle at ING.

7:15

It offers an RPO of two minutes, meaning you can lose 2 minutes of data.

7:22

If you have 100 transactions per second, that's 200 transactions.

7:29

That's 12,000 transactions, right?

7:33

So for your most critical data, use multiple solutions on the application level, middleware level, hardware level.

7:42

For your most critical data, create also a solution in your application.

7:48

Maybe keep an input law at the top of your application chain.

7:53

Or create an integrity check where you compare this input law with your database during a failure.

8:00

Or you compare your database with another related database.

8:05

Maybe write to multiple data stores, why not?

8:09

And of course test, test, test.

8:12

If you look at your data, you could divide your data in three categories.

8:18

You have your data in memory, transient data or non persistent data.

8:23

You have your persistent data on disk, the file managed by your OS, and you have persistent data managed by special middleware like a DBMS, Oracle, SQL Server or by no SQL software like Cassandra or Graph.

8:43

And if we talk about data, we have to discuss in detail the term stateful and stateless.

8:49

In the context of a web application.

8:52

Stateful means if a request goes to 1 server, session data is stored on this one application server.

9:03

So if the user using his browser clicks again, sends a second HTTP request, the second HTTP request, if it goes to another instance of your application, to another application server, there will be a problem.

9:20

There is no state, no session data stored on the second server.

9:27

The session data is only stored in the first server.

9:31

This is an example of stateful sessions.

9:34

Implicitly a single point of failure and we want our applications or APIs to be stateless.

9:42

It shouldn't matter to which instance the request is sent, there should not be a single point of failure.

9:50

I want to be high available, so if your application is stateless, high availability is much much easier and that's one of the reasons to separate your data from your application.

10:05

Separate your data from your application server.

10:10

Maybe a good example in the context of a web application is to use Cassandra or an ING key space as a service to store your session data.

10:21

So separate your data from your application server and store it in Cassandra.

10:27

Now each application or application server can retrieve the session data from Cassandra.

10:34

And yes, Cassandra needs to be high available, but your application it's now much easier to have your application high available.

10:44

Your application is stateless, scalability is much easier, High availability is much easier.

10:52

Let's look at another example.

10:54

Let's say your application needs to process a file and the file is stored locally on the machine of this application server.

11:03

What if you want to be highly available?

11:06

You need 4 hosts, maybe 4 instances of your application.

11:12

But if the file is stored only on the first machine, on the first host, how can the other instances access this file?

11:21

Or let's say you have to change this file.

11:25

How does that work?

11:27

That's a problem.

11:29

Isolate the data from your host, from your application server.

11:33

Put it on a separate machine, maybe put it in the S3 object store.

11:39

That's cool technology that we have available in the IPC we'll see later on.

11:45

I put it maybe on a NASA shared file system and maybe put it on a central managed device like the Nas or the S3 OpEx store.

11:56

Or order a separate machine in your private network and make that 1 high available for your data, but separate your data from your application and put it on a central on a separate machine or maybe on a device which is centrally managed like the S3 OpEx store or on us.

12:20

Yeah.

Dataha2-Reliaba 1

0:04

Let's now have a look at the different data technologies available at ING and. In this context I mean persistent data. Non persistent data memory is usually lost during a failure. If your applications

are using the request reply pattern, the requester must be able to resend the request. And maybe you remember we've seen.

0:28

Maybe you can store data in memory in a fast no SQL data store like gas key space as a service. At ING on the infra level there is a technology called vmotion. It's offered by the virtualization layer VMware. Actually it is able to move learning host including memory to different hardware but that's not available as a self-service for the consumer.

0:56

Within the IPC, several data services are available as some are standard, some are preferred. First, we'll talk about Comvolt, the standard backup tool for your host files and SQL Server. We'll talk about the databases, database management, systems, day bus, SQL Server. We'll talk about key space as a service, We'll talk about the ECSF 3 object store, and we will talk about Nas, shared file system and archiving.

1:27

Of course there are more services available, let's say Redis in the IPC, but please discuss with your architect or BTA which technology is preferred. An Xfba, file transfer technology and messaging like Kafka or NQ are discussed in a separate chat, but Please remember, because you can doesn't mean it's allowed.

1:53

If you log on to the IPC portal, you can order a lot of technologies, but that doesn't mean it's preferred within your domain. Imagine you log on to Azure or AWS or Google. It's like a candy store. You can order whatever you want. And they want to make money of course. And the IPC is built for the whole of ING, and ING is a big company, you know.

2:19

It's Belgium, the Netherlands, Germany, Italy, the Philippines. The preferred technologies can differ per domain. Please consult your architects or BTA which technology is preferred in your domain. In the IPC, Convolt is the standard tool for backups. And by the way, all data is stored on fast SSD.

2:46

So your host, your files, also SQL Server that you can backup using Convolt. Keep in mind when you order a machine in, the IPC backup is not enabled by default. It's your choice. It's a check mark during provisioning of a host in the IPC portal and you can configure in guest backups.

3:09

Meaning you're not backing up your complete host and all attached files, but you can make a selection and remember you can only restore to the same host to the same VM. There are several second day operations available for convolt in the IPC. First you can schedule your backup. You can set the retention. For 31 days or 14 days.

3:37

You can change the Service level agreement from Platinum to Gold, meaning you don't have replication over the data centers. Or of course you can suspend or resume your backup. Keep in mind Convault is a backup tool, not an archiving tool. We have different tools for that. In the IPC, two different functionalities are available. For backup, you have backup for Convault.

4:06

And you can create a snapshot with a second day operation. Please realize the difference. A snapshot which is VMware technology creates a delta file which is continuously updated. A snapshot is not meant for long term existence. You should remove a snapshot as soon as possible if you have a snapshot.

4:32

The snapshot. This delta file is continuously updated and it can take performance from your host. Not only that it takes storage. So please realize use a snapshot maybe before patching, but always clean up remove a snapshot as soon as possible. So as I just mentioned, keep in mind that you are responsible for backing up your host before patching.

5:02

Maybe you have a Patch me Now button in the IPC self-service portal. Please make a backup before patching. This is the shift left. Maybe you're used to an infra department creating the backup for you before patching. Now it's the responsibility of the consumer to create a backup before patching. Before we're going to talk about day bus and SQL Server, let's state the standard.

5:33

First of all, a database belongs to one application, or maybe I should say to one purpose or product ID. And a database belongs to one network zone private network. Implicitly I'm saying it's tightly coupled to an application. What do I mean? That a JDBC protocol used to connect to a database like Oracle?

6:00

Is a direct connection between the application and this database server. It's not like HTTP where I can put a generic proxy or HTTP server or a load balancer in between. Also keep changeability in mind. Let's say you have multiple applications going to the same database to the same data and the data structure changes.

6:27

In that case all applications must be changed. Put an API in between and only this one API can connect to this database. All connections to the database must be encrypted using TLS 1.2. Production and non production environments must always be separated and use data or SQL Server only for your critical business data.

6:57

Don't use it for non important technical data. Maybe install your own post address database, manage it yourself or put it in a different data store. Be cost rare. Day bus or database as a service has its own dedicated hardware within the IPC with built in redundancy. It means if you order day bus you don't get your own VM in your private network.

7:26

You get a PDB, a pluggable database. This means you get your own private database with guaranteed service level in a larger shared container database. It also means you cannot log on to your database server, you can Only Connect remotely using port 1521. It's also not possible to directly use the file system of the Oracle server.

7:55

However, there's a feature DBFS and when you order a day bus, 2 service levels are offered Thin and normal. Thin offers high availability within one data center. Normal adds high availability over the data centers. Keep in mind always order Thin for development and test. And please check the day bus blueprint on our conference page and keep in mind.

8:24

Even though Day Bus offers built in redundancy, it's your responsibility to configure your application correctly to configure application continuity as it's called within Oracle and look closely at the service description of Day Bus. Day Bus uses forever incremental backups with set DLR Zero Data loss Recovery appliance.

8:51

So it's using dedicated hardware, state-of-the-art Oracle software still RPO is 2 minutes. There is a risk of data loss of two minutes. Remember, zero data loss or to guarantee zero data loss. That's very very difficult. And Day bus offers several second day operations. Please check the slide.

9:17

May be worth mentioning, quite recently Mask Me Now was added. It involves contacting another central team to create the masking rules because remember there cannot be production data in a non production environment. In the IPC you can also order SQL Server, your own machine, your own private network. 2 flavors are offered.

9:44

Single instance 1. High available and SQL AlwaysOn where you get two or four machines divided over the two data centers and by default it includes the ETL components, Integration services. So SQL Server AlwaysOn is the high availability feature of SQL Server. You can choose two or four machines.

10:09

Having high availability within one data center and over the two data centers, you need an extra machine like a management machine of SQL Server to manage this failover. This is called the failover witness. Also like with day bus, you have to configure your application correctly. Your application needs to connect to an availability group listener.

10:38

Keep in mind SQL Server is an EAS offering infrastructure as a service. Opposed to data, you need an expert in your squad or a support team at hand. For example, there are no self-service functionalities in the portal. You can install self-service scripts or software on the machine, but that's your responsibility.

11:05

But that self-service software is created by the central infra team but you have to install it on your machine. Con Vault is used for the backups of SQL Server, but the RPO depends on your configuration.

Dataha3a-1

0:03

With a conventional database like Davis, Oracle, or SQL Server, all data has an inherent structure.

0:11

You need a schema, a formal definition of your data, how it's inserted in the database.

0:19

With no SQL, data can be stored in a schemaless or free form fashion.

0:25

Any data can be stored in any record.

0:29

Some of the characteristics of these no.

0:31

SQL data stores are well, first of all, they don't support transactional processing, rollback, commit.

0:39

They're very scalable.

0:41

It's much, much simpler to scale horizontally to spread the data over multiple physical devices.

0:51

The performance is much, much higher.

0:54

They're very, very fast.

0:56

They offer what we call eventual consistency.

1:00

Even though the data is copied over multiple devices at a certain point in time, I don't know if the data is exactly the same synchronized between all these devices.

1:13

Yes, after a certain point in time I can be sure the data is the same.

1:19

And often these no SQL data stores offer caching capabilities, increasing performance even more at ING key space as a service is available, that's actually a no SQL data store.

1:36

Cassandra it's a hotel.

1:38

So you can request your own key space within this hotel, but you don't get your own Vms or machines in your own private network.

1:48

With key space as a service, you always get multiple replicas of your data spread out over two data centers.

1:56

It's the only data store which offers active over two data centers, so your data can be accessed in both data centers.

2:08

The consistency levels within key space as a service are configurable.

2:14

That's your responsibility.

2:17

Encryption of data and transport is on the roadmap and point in time.

2:21

Restores are again the responsibility of the squads.

2:27

And implicitly I'm saying RPO depends on your configuration.

2:33

You are responsible for the consistency levels for the number of replicas etc.

2:41

You can send a request to the responsible squad for an additional backup snapshot.

2:48

But you really have to ask yourself do you need this keyspace as a service in OS?

2:55

SQL data store at ING can never be a store.

3:00

I mean system of record keyspace as a service should be used as a cache.

3:07

It can never be the origin of the business data.

3:11

It should be a copy of this business data.

3:14

Therefore, you must always be able to restore the data in an OSQL data store in Key Space as a service from your store which is usually a DBMS like day bus or SQL Server.

3:30

Let's look at an example of A use case for Key space as a service.

3:35

You can find more use cases on the Confluence space of Key Space as a service.

3:40

But let's say I have my origin, my business data, my critical business data in the DBMS in day bus.

3:49

All inserts, updates, deletes, changes of data are taking place in this database with transactional processing.

4:00

But let's say whenever the data changes, an event is sent to the event bus Kafka and this event is processed and key space as a service is updated.

4:14

So you have a copy of your data in key space as service and all the gets all the retrievals of the data are happening on key space as a service and that's super fast.

4:30

But your critical business data is in a database offering asset atomicity, Consistency, isolation and Durability.

4:40

And this is what we call Command Query Separation.

4:44

All the changes, insert, the deletes are taking place on a database, on database for example, but all the retrievals and gets are sent to the cache to key space as a service.

4:58

But be aware that there is a risk that the cache is not In Sync with the sort and now in the IPC the ECSS 3 OPX store is available.

5:11

It's a low cost and fast solution.

5:14

You can order your own namespace within the OPX store and within this namespace you can create buckets.

5:22

It uses as an interface the S3 API coming from Amazon.

5:28

Data and transport is encrypted and of course you can encrypt the data in REST.

5:34

Access is secured with secret codes, hashes, API keys and because of the popularity of Amazon S3, many storage providers are using S3 as a standard interface.

5:47

And in the IPC you can order your local OPX storage namespace in WPR or DCR or a replicated namespace from DCR to WPR or the other way around.

6:04

Then there are several second day operations available for the object store.

6:08

You can create, add your user, you can change the password, change the size, extend the namespace, retrieve details or remove your storage, your namespace.

6:20

When you have an object user and password, you can create a bucket using the S3 API.

6:29

With the ECS OPX store you are responsible for your backups.

6:34

Maybe you can use versioning and write the file twice.

6:38

And you can set your own retention settings using the S3 API.

6:43

And keep in mind even though you have ordered a replicated OPX store, you can lose 15 minutes of data.

6:52

RPO is at least 15 minutes, so the ECS OPX store is fast, it's cheap and you could say it's cloud ready, but it's also relatively new within ING.

7:06

So please consult your architects or Btas for the use cases and consider for instance the management of your keys, your API keys, the secrets in the IPC.

7:19

You can also order a Nos a file share and this can be replicated or non replicated.

7:27

This is also a central service so you don't get your own Vms in your private network.

7:34

The implementation of this not file sharing is on an IC loan MOS cluster and it's using asynchronous replication with at least 10 minutes delay, meaning your RPO is at least 10 minutes.

7:49

Some of the characteristics are you need client software on your machine.

7:54

There is no looking offered.

7:56

Write ones, read many.

7:58

The only protocol supported is NFS.

8:01

SMB is coming soon, you can't create backups.

8:05

And again Nas is not meant for archiving and like the ECS OPX store, no backups are offered.

8:15

That's your responsibility.

8:17

Maybe order 2 shares, one in each data center.

8:21

Solve it in your application, right from your application synchronously or asynchronously to both shares within market leaders.

8:33

Only use the Nos file share when the S3 object store is not possible.

8:38

And keep in mind use a file share only within one network zone for one application.

8:46

It's again considered tightly coupled.

8:49

There's a direct dependency connection from your application to this file share and then we have an archiving solution in the IPC.

9:00

For archiving you often need warm write once read many for compliancy reasons or legal reasons.

9:09

You have to be able to prove that the data has not been changed.

9:14

We call this a warm share.

9:16

You have to request a warm share or it's sometimes called in compliancy mode or smart lock.

9:23

It uses the same technology as the NUS, the network attached storage, Isolo actually for now it's not available as a self-service, but you have to fill in a request form and now it's now May 2021.

9:41

So maybe if you watch this movie it's already available as a self-service.

9:46

A worm is only request and if you need more than five terabytes you have to request more than three months or three months in advance.

9:59

And this has to do with capacity planning, capacity management and you can set quotas hard or advisory with hard limits.

10:10

If you exceed the limit, the operation will fail.

10:14

But you can set notifications in the archiving solution so you get a notification if this error occurs or if you exceed the limit.

10:26

And in an advisory mode, you will only get a notification, but the operation will not fail.

10:35

So you've seen the several data services available in the IPC and which are preferred within market leaders.

10:43

But I hope you do realize that none of the high availability solutions involved ordering a Dr.

10:50

capable machine do not order Dr.

10:53

capable machines in the IPC.

10:57

So I've stated here quite some key takeaways that I hope you will remember from these data high availability sessions.

11:05

First of all, availability is not the same as integrity.

11:09

Keep the performance of your backup restore in mind an RPO zero, Solve it in your application, separate the data from your application or application server and because you can in the IPC portal doesn't mean it's allowed.

11:28

Enable your backup during provisioning when needed.

11:33

Always remove a snapshot as soon as possible.

11:37

Create a backup before patching.

11:40

Use a database day bus or SQL Server only as a source for your critical business data.

11:47

And when you need a high available database like day bus or SQL Server, configure your app correctly and use the S3 OPX store whenever possible.

12:01

I would say that's a Cloudready solution.

12:04

O yes, because the aware, security aware, cloud aware.

1

00:00:11,499 --> 00:00:12,138

Hello!

2

00:00:12,938 --> 00:00:18,926

In this module, we'll briefly go over the options you have with DBaaS From an infrastructure perspective.

3

00:00:19,546 --> 00:00:25,696

You can use this information to order, monitor and use DBaaS to make your application as reliable as

4

00:00:25,706 --> 00:00:26,363
required.

5
00:00:28,294 --> 00:00:34,143
This module will not go into full details of everything offered, but a slide deck with more information is

6
00:00:34,153 --> 00:00:40,051
provided, together with important links to DBaaS documentation where you can find the latest information.

7
00:00:41,851 --> 00:00:48,500
DBaaS does not use the red and blue sub availability zones, but only focusses on the data centre locations

8
00:00:48,510 --> 00:00:54,738
for deployment, being Data centre WPR in Rotterdam and Data centre DCR in Roosendaal.

9
00:00:56,639 --> 00:01:02,746
When ordering your Pluggable Database, or PDB, there are two service options to select from;

10
00:01:02,906 --> 00:01:04,014
thin and normal.

11
00:01:04,844 --> 00:01:05,534
Thin,

12
00:01:05,663 --> 00:01:10,226
intended for Development and test, creates a database on one data centre.

13
00:01:11,046 --> 00:01:12,096
With normal,

14
00:01:12,226 --> 00:01:17,463
intended for acceptance and production, a second one will be created on the other datacentre.

15
00:01:18,093 --> 00:01:22,089
Oracle dataguard ensures that data is replicated to the other side.

16
00:01:22,899 --> 00:01:29,448
Although your database is accessible from both datacentres,

it is advised to deploy your primary database on the

17

00:01:29,459 --> 00:01:33,514

same datacentre as your application server to limit latency.

18

00:01:34,833 --> 00:01:39,439

When the order is complete, your PDB is created and placed in a container.

19

00:01:40,069 --> 00:01:45,288

This is important to keep in mind, as some actions are only possible on container level.

20

00:01:46,098 --> 00:01:52,268

From an infra perspective, there are no actions required on the configuration of your database to enhance the

21

00:01:52,278 --> 00:01:54,439

reliability of your database.

22

00:01:56,339 --> 00:02:03,464

A database provided by DBaaS at creation is maximized to shape 8 and 100GB storage.

23

00:02:04,294 --> 00:02:07,376

These are not the maximums you can use for your PDB.

24

00:02:08,006 --> 00:02:18,001

For your shapes you can go up to 32 GB on thin, 64 GB on normal and 128 GB on big data service levels.

25

00:02:19,911 --> 00:02:26,040

To change the shape and or the storage of your database, you can use the second day operation "Resize oracle

26

00:02:26,050 --> 00:02:26,875

database".

27

00:02:27,726 --> 00:02:30,800

Storage can be extended up to 15 TB.

28

00:02:31,420 --> 00:02:35,488

Please contact the DBaaS team if you require more storage space.

29
00:02:36,298 --> 00:02:42,128
Instead of adding storage to your database you can
also consider advanced compression, as you benefit

30
00:02:42,138 --> 00:02:44,338
from lower costs and better performance.

31
00:02:46,269 --> 00:02:51,351
Monitoring on your database can be done via Oracle
Enterprise Manager, OEM.

32
00:02:51,951 --> 00:02:57,964
You can log on with your corporate key, the PDBs
you will be able to see depend on your ACE team membership.

33
00:02:58,773 --> 00:03:04,604
A standard set of metrics and triggers are predefined
for you, but you can change and add new ones yourself

34
00:03:04,613 --> 00:03:05,414
if you need to.

35
00:03:06,023 --> 00:03:12,063
Please note that triggers to create incidents are
only possible on Acceptance or Production instances.

36
00:03:13,984 --> 00:03:17,601
Backups are continuous for both thin and normal service
level.

37
00:03:18,211 --> 00:03:23,501
Oracle Zero Data Loss Recovery Appliance (ZDLRA)
is used as the backup solution for DBaaS.

38
00:03:25,021 --> 00:03:32,081
Thin service level database only exists in one datacentre,
DBaaS continuously replicates the backup to the other

39
00:03:32,081 --> 00:03:38,051
datacentre to restore the database if the local ZDLRA
is unavailable.

40
00:03:38,861 --> 00:03:44,991
A normal service level database has an instance of
the database in the two different datacenters and

41
00:03:45,001 --> 00:03:50,551
each instance is continuously backed up to each local
ZDLRA.

42
00:03:51,371 --> 00:03:55,714
There's a 5 day retention on thin, and a 10 day retention
on normal.

43
00:03:56,314 --> 00:03:59,839
You can restore back to any time within that retention
window.

44
00:04:00,449 --> 00:04:05,799
Please note as this will replace the current running
instance you will need an approval of your BTA to

45
00:04:05,809 --> 00:04:07,001
initiate the restore.

46
00:04:08,111 --> 00:04:13,576
The "Recovery Time Objective" (RTO) for both DBaaS
service levels is 8 minutes.

47
00:04:14,686 --> 00:04:21,051
The "Recovery Point Objective" (RPO) for both DBaaS
service levels, is a maximum of 2 minutes.

48
00:04:21,661 --> 00:04:27,751
If an outage occurs, this means the maximum amount
of data that is allowed to be lost, is 2 minutes worth

49
00:04:27,761 --> 00:04:28,289
of data.

50
00:04:29,379 --> 00:04:35,209
It's important to know that the backup cannot be used
to copy your database, it is not intended to be used

51
00:04:35,219 --> 00:04:36,889
to archive or vault data.

52
00:04:37,509 --> 00:04:40,488
Please use the appropriate tooling for those functions.

53
00:04:42,379 --> 00:04:49,148

Because DBaaS runs on appliances, a quarterly full stack download has to be applied four times a year

54
00:04:49,158 --> 00:04:50,726
to remain a supported product.

55
00:04:51,356 --> 00:04:55,851
For now this is not a self service capability but
a provider driven patch.

56
00:04:56,551 --> 00:05:03,001
DBaaS uses the so called provider driven patches,
4 times per year the DBaaS team will role out these

57
00:05:03,011 --> 00:05:03,651
patches.

58
00:05:04,251 --> 00:05:08,326
The role out will be done in rolling fashion, container
after container.

59
00:05:09,136 --> 00:05:12,414
As a user, you can choose between two patch rings.

60
00:05:13,014 --> 00:05:16,488
The difference between the two is the date when it
will be patched.

61
00:05:17,108 --> 00:05:19,171
Ring one will be patched first,

62
00:05:19,301 --> 00:05:23,613
and after a grace period where stability is checked,
ring 2 will be patched.

63
00:05:24,233 --> 00:05:25,606
Out of the box dev,

64
00:05:25,736 --> 00:05:26,376
test

65
00:05:26,618 --> 00:05:30,326
and acceptance are placed in ring 1 and production
in ring 2.

66

00:05:31,146 --> 00:05:36,363
Via a second-day operation you can change the patch
rings to fit your own patch strategy.

67
00:05:37,213 --> 00:05:43,084
Patching does bring downtime to your database, but
will not change your database connection string, nor

68
00:05:43,094 --> 00:05:44,564
the datacenter it's running in.

69
00:05:46,474 --> 00:05:48,801
There are three scenario's with DBaaS.

70
00:05:49,601 --> 00:05:51,414
The first one is a restore.

71
00:05:52,044 --> 00:05:56,001
This scenario is possible for both thin and normal
service level.

72
00:05:56,611 --> 00:06:02,541
As mentioned earlier, this can be triggered by using
a second day operation and will require the approval

73
00:06:02,551 --> 00:06:03,426
of a BTA.

74
00:06:04,236 --> 00:06:10,239
The second scenario is a DR exercise and only applies
to PDBs with service level normal.

75
00:06:10,839 --> 00:06:14,213
In this scenario you switch from one datacentre to
the other.

76
00:06:14,843 --> 00:06:17,901
This can be done by performing a second day operation.

77
00:06:18,521 --> 00:06:24,713
An active PDB needs to be present, therefore you
can not perform this action during an actual disaster.

78
00:06:25,514 --> 00:06:28,264
The last scenario is an actual disaster.

79
00:06:28,864 --> 00:06:34,994
In this case there is nothing you can do on database
level, and only applies to PDBs with service level

80
00:06:35,004 --> 00:06:35,463
normal.

81
00:06:36,273 --> 00:06:42,523
If it's a local disaster, like a hardware failure,
your PDB will be switched over to a different computer

82
00:06:42,533 --> 00:06:42,963
node.

83
00:06:43,764 --> 00:06:49,573
In the case of a full datacentre failure, the DBaaS
team will perform the actual switch over, which will

84
00:06:49,584 --> 00:06:51,251
happen container to container.

85
00:06:51,851 --> 00:06:57,988
Your PDB will be available once the switchover of
the container that hosts your PDB is completed.

86
00:06:59,919 --> 00:07:01,889
This concludes the DBaaS topic.

87
00:07:02,519 --> 00:07:06,176
I hope you enjoyed and I look forward to seeing you
in the next video!

1
00:00:15,310 --> 00:00:15,975
Hello,

2
00:00:17,075 --> 00:00:23,387
In this module we'll briefly go over the options
you have with MSSQL from an infrastructure perspective.

3
00:00:24,007 --> 00:00:30,737
You can use this information to order, monitor and
use MSSQL to make your application as reliable as

4
00:00:30,747 --> 00:00:31,412

required.

5

00:00:32,242 --> 00:00:38,692

This module will not go into full details of everything offered, with this module a slide deck with more information

6

00:00:38,702 --> 00:00:45,925

is provided, also important links to MSSQL documentation where you can find the latest information is included.

7

00:00:47,825 --> 00:00:51,650

Let's look at the options you have to set up your SQL environment.

8

00:00:53,469 --> 00:00:59,537

When you order your SQL server, you can go for either a singular deployment or SQL always on.

9

x

The singular deployment can be a deployment in one of the four sub-availability zones.

10

00:01:05,627 --> 00:01:08,500

But with always on you can make your data high available.

11

00:01:11,159 --> 00:01:17,949

SQL Always on knows two variations, a 2 node configuration and a 4 node configuration.

12

00:01:18,569 --> 00:01:21,974

Both of the solutions have an availability rating of 4

13

00:01:22,174 --> 00:01:23,724

directly out of the box.

14

00:01:24,524 --> 00:01:29,934

Important thing to know is that the option, set multisubnetfailover is set to true

15

00:01:30,894 --> 00:01:36,044

Add this setting to the connection string of your Application to the Availability Group Listener to

16

00:01:36,054 --> 00:01:38,137

improve "re-connection" performance.

17
00:01:38,757 --> 00:01:45,457
The Always on setup is deployed cross-datacenter
and the Listener has 2 IP addresses, one for each

18
00:01:45,457 --> 00:01:47,024
Datacenter IP subnet.

19
00:01:47,614 --> 00:01:53,964
By using this setting, the Listener can connect without
connection-waiting time to the SQL Server that contains

20
00:01:53,974 --> 00:01:55,449
the Primary Databases.

21
00:01:58,849 --> 00:02:01,111
Let's have a look at Capacity management.

22
00:02:06,012 --> 00:02:08,486
You can use SCOM to monitor your capacity.

23
00:02:09,066 --> 00:02:13,687
If setup correctly, a ticket will be created for
you once the threshold has been hit.

24
00:02:14,286 --> 00:02:17,837
The standard threshold is set at 20% free capacity.

25
00:02:18,467 --> 00:02:23,362
This can both be set to your database filling up
or if you're running out of diskspace.

26
00:02:25,282 --> 00:02:30,252
There are a few simple steps you can take to keep
your database capacity healthy once you've hit your

27
00:02:30,261 --> 00:02:30,949
threshold:

28
00:02:31,749 --> 00:02:36,612
The first step is to properly identify the database
that is running out of space.

29
00:02:37,212 --> 00:02:40,886
You can use the appropriate self service stored procedure

to do that.

30
00:02:41,487 --> 00:02:45,011
The second step is trying to understand why it is
filling up.

31
00:02:45,621 --> 00:02:50,761
Up to a certain point you can keep extending, but
it's more prudent to verify what's happening.

32
00:02:51,911 --> 00:02:57,762
Ask yourself if it is possible to run a clean up
job to archived data that doesn't need to be online ?

33
00:02:58,872 --> 00:03:05,712
Is it a logfile growing due to a one off action or
is the transaction-log not being backed-up and subsequently

34
00:03:05,722 --> 00:03:06,349
be cleaned?

35
00:03:10,259 --> 00:03:13,848
The next step is to verify current disk usage on
the VM,

36
00:03:14,169 --> 00:03:16,699
confirm you're actually running out of space.

37
00:03:17,299 --> 00:03:22,111
By default the databases are stored on Disk number
4, labeled USERDB DATA.

38
00:03:22,701 --> 00:03:26,336
If there is still room you can immediately resize
your database.

39
00:03:28,237 --> 00:03:32,647
If that's the case you can extend the disks through
the IPC ordering portal

40
00:03:37,472 --> 00:03:41,499
The next step is to update the disk size on the windows
OS.

41
00:03:42,109 --> 00:03:49,199
As MSSQL runs on windows core, you will have to retrieve

the CAL account from password vault and use CLI to

42

00:03:49,209 --> 00:03:50,274

perform the action.

43

00:03:50,904 --> 00:03:56,634

Please ensure that you end the action with the logoff command, and not exit or quit as those commands will

44

00:03:56,644 --> 00:03:59,099

close the session but will not log you off.

45

00:04:00,189 --> 00:04:04,974

And finally, extend the database on the extended volumes in SQL server.

46

00:04:05,574 --> 00:04:11,524

Check the new size with the disk-usage command to confirm the added space before you perform the resize

47

00:04:11,534 --> 00:04:12,637

of your database.

48

00:04:14,537 --> 00:04:16,806

Let's have a look at Performance monitoring

49

00:04:19,677 --> 00:04:24,387

SCOM is used to perform the monitoring, but it's not mandatory to have this enabled.

50

00:04:25,017 --> 00:04:28,499

This is a step you have to take in ServiceNow to enable this.

51

00:04:31,329 --> 00:04:38,129

Find your CI in service now, go to the monitoring tab, check the monitoring and accepted checkboxes

52

00:04:38,129 --> 00:04:41,074

and choose GMCR in the field monitored by.

53

00:04:45,474 --> 00:04:52,036

If you don't want to use omnibus, but use IAT instead, check the disable auto-ticketing events box.

54

00:04:55,957 --> 00:05:00,599
Make sure you onboard your application to IAT if
you choose this option.

55
00:05:04,399 --> 00:05:08,998
Squaredup can be used to create your own dashboards
and to view the SCOM alerts,

56
00:05:10,148 --> 00:05:13,861
onboarding to squaredup is done automatically on
P-code level,

57
00:05:14,951 --> 00:05:17,636
but you need to onboard each server separately.

58
00:05:18,236 --> 00:05:23,236
You can verify if onboarding is successful by triggering
the maintenance mode for squaredup.

59
00:05:25,137 --> 00:05:27,366
Let's have a look at Back-ups and restore

60
00:05:29,129 --> 00:05:36,219
There are two types of back-up available for the MSSQL
service, regular snapshot back-up of the VM and database

61
00:05:36,229 --> 00:05:36,887
backups.

62
00:05:37,696 --> 00:05:44,377
The snapshot back-up is not mandatory and can be selected
either at deployment or as a second day operation

63
00:05:44,386 --> 00:05:45,412
in the cloud portal.

64
00:05:46,212 --> 00:05:51,149
The windows server back up will back-up all local
files, including the database file.

65
00:05:52,259 --> 00:05:56,161
However, data consistency is not guaranteed with
this option.

66
00:05:56,971 --> 00:06:03,951
To ensure data consistency for database backups, all

databases in your SQL server are automatically added

67

00:06:03,961 --> 00:06:08,824
to the Commvault back-up schedule, and it will perform
the back-ups using a pull mechanism.

68

00:06:09,424 --> 00:06:16,424
The retention period is either 14 or 31 days, depending
on your selection during the ordering process.

69

00:06:17,534 --> 00:06:19,637
There are two types of back-ups offered.

70

00:06:20,257 --> 00:06:22,887
A full database backup which runs daily,

71

00:06:23,987 --> 00:06:27,949
and a transactionlog back-up which is scheduled every
two hours.

72

00:06:29,049 --> 00:06:32,837
You can combine the two to restore to a point in
time you desire.

73

00:06:33,967 --> 00:06:39,877
Please note that each SQL deployment is configured
to "full recovery mode", if you've changed it to simple

74

00:06:39,887 --> 00:06:43,837
recovery, only full database backups are valid for
a restore.

75

00:06:44,937 --> 00:06:50,957
And finally you can create an adhoc back-up locally,
this option is especially helpful before a change

76

00:06:51,027 --> 00:06:52,524
or transfer purposes.

77

00:06:53,134 --> 00:07:00,104
Please be aware this local backup will only be stored
for 24 hours, after which it will be removed automatically

78

00:07:01,109 --> 00:07:07,459
In case you want to restore a SQL Server database
into a new database next to the existing one, there

79
00:07:07,469 --> 00:07:08,874
are two things to consider:

80
00:07:10,004 --> 00:07:16,137
First, 'alternate database path' and secondly 'the
co-location of data- and transactionlog-files'.

81
00:07:19,037 --> 00:07:24,937
You can restore your database from the cloud portal,
there are 8 potential scenario's to perform a restore

82
00:07:24,946 --> 00:07:27,337
with the several options that are available to you.

83
00:07:31,267 --> 00:07:36,211
Please visit the back-up and archiving documentation
to find all the examples.

84
00:07:37,021 --> 00:07:43,331
In case of SQL always on, you have to take the machine
you're going to restore on first out of the availability

85
00:07:43,341 --> 00:07:44,974
group before you can proceed.

86
00:07:46,874 --> 00:07:51,023
Let's have a look at Patches and change strategy using
IPC tools

87
00:07:53,939 --> 00:07:58,949
MSSQL server follows the same ring model patching
cycle as windows server.

88
00:08:02,869 --> 00:08:07,911
Using the "SCOM" portal you can verify in which maintenance
window your server is placed.

89
00:08:08,501 --> 00:08:13,836
It is possible to change this window using the Set-
MaintenanceWindow.ps1
script.

90
00:08:14,166 --> 00:08:17,874
Please note that it can take up to 8 hours before
changes are set.

91
00:08:19,774 --> 00:08:22,744
Let's have a look at Disaster recovery scenarios

92
00:08:23,846 --> 00:08:25,636
With SQL Always ON,

93
00:08:25,966 --> 00:08:32,356
the scenarios are the same during a DR exercise as
it would be during an actual disaster, when an outage

94
00:08:32,366 --> 00:08:34,286
of a complete datacenter occurs.

95
00:08:39,187 --> 00:08:42,086
The differences in scenario depends on your set-up.

96
00:08:42,696 --> 00:08:45,836
Have you chosen for a 2 node or a 4 node set-up?

97
00:08:46,446 --> 00:08:50,399
Depending on which node or nodes go down, the situation
differs.

98
00:08:55,329 --> 00:08:57,836
This concludes the MSSQL topic.

99
00:08:58,666 --> 00:09:02,249
I hope you enjoyed and look forward to seeing you
in the next video.

1
00:00:11,499 --> 00:00:12,164
Hello,

2
00:00:12,954 --> 00:00:20,244
In this module we will briefly cover available options
related to KaaS, Keyspace as a Service, from an infrastructure

3
00:00:20,253 --> 00:00:20,914
perspective.

4
00:00:21,733 --> 00:00:27,964
You can use this information to order, use KaaS and
consider reliability options for your applications.

5
00:00:29,864 --> 00:00:37,053
Next to traditional relational database management
systems like Oracle, Microsoft SQL server and MySQL,

6
00:00:37,053 --> 00:00:40,826
you also have the option to use Keyspace as a Service.

7
00:00:41,326 --> 00:00:48,326
KaaS consists of Cassandra keyspace which resides
on one of the Datastax Cassandra clusters within IPC.

8
00:00:49,126 --> 00:00:55,836
Cassandra is a distributed NoSQL database management
system designed to handle large amounts of data across

9
00:00:55,846 --> 00:00:58,651
multiple servers to provide high availability.

10
00:00:59,251 --> 00:01:05,126
Cassandra uses a leaderless topology, meaning all
nodes are capable of performing the same tasks.

11
00:01:05,726 --> 00:01:09,226
Next to that, Cassandra has no single point of failure.

12
00:01:10,046 --> 00:01:17,276
While traditional relational databases like Oracle,
Microsoft SQL server and MySQL, store structured data

13
00:01:17,286 --> 00:01:24,646
using schemas, and have to keep track of relationships
in the database, Cassandra is schema-less and it can

14
00:01:24,656 --> 00:01:25,889
handle unstructured data.

15
00:01:26,499 --> 00:01:31,539
Developers can use a query language CQL which is
very similar to SQL.

16
00:01:32,389 --> 00:01:38,778
Application teams for which their Application or API
is fit for use for Cassandra, will be guided through

17

00:01:38,789 --> 00:01:41,114
the onboarding process with the KaaS team.

18
00:01:41,914 --> 00:01:48,713
Cassandra is positioned as the ING standard for Key-Value stores, session data and cache use-cases.

19
00:01:49,334 --> 00:01:52,388
Using KaaS for System of Records should be avoided.

20
00:01:53,018 --> 00:01:59,688
For System of Records, traditional Database management systems should be used instead to ensure consistency

21
00:01:59,698 --> 00:02:00,739
of the source data.

22
00:02:02,649 --> 00:02:08,708
A keyspace is the top-level database object, that controls the replication for the object it contains

23
00:02:08,718 --> 00:02:10,476
at each datacenter in the cluster.

24
00:02:10,976 --> 00:02:15,989
Keyspaces contain tables, user-defined types, functions and aggregates.

25
00:02:16,599 --> 00:02:20,064
Typically, a cluster has one keyspace per application.

26
00:02:20,884 --> 00:02:23,926
A cluster can be stretched across multiple datacenters.

27
00:02:24,756 --> 00:02:31,596
The keyspace replication strategy and replication factor, control data availability for a set of tables,

28
00:02:31,596 --> 00:02:33,451
in each datacenter of the cluster.

29
00:02:35,341 --> 00:02:40,891
Depending on the target environment, the keyspace will be created on the following environments, with

30

00:02:40,901 --> 00:02:42,813
the following minimum configuration:

31
00:02:43,633 --> 00:02:49,526
The environment uses two datacenters per cluster,
where at least 5 nodes per datacenter are configured.

32
00:02:50,325 --> 00:02:55,256
The replication strategy option that is used, is 'Network
Topology Strategy'.

33
00:02:55,830 --> 00:03:00,876
This option applies the replication setting per datacenter
with the default value of 3.

34
00:03:01,705 --> 00:03:04,926
The environment uses 3 Seed nodes per datacenter.

35
00:03:05,746 --> 00:03:10,151
The application teams are responsible for data consistency
in Cassandra.

36
00:03:10,730 --> 00:03:16,013
There are different consistency levels (CL) available
for the Write and the Read path.

37
00:03:16,813 --> 00:03:19,688
The Write and the Read path are separate from each
other.

38
00:03:20,518 --> 00:03:27,168
The keyspace that is created, is replicated three
times in each datacenter; six times in total in a

39
00:03:27,178 --> 00:03:28,450
two datacenter setup.

40
00:03:29,260 --> 00:03:35,663
The delivered cluster configuration, either Shared
cluster or Dedicated cluster, depends on the use-case.

41
00:03:36,263 --> 00:03:40,413
The final decision is made by Minions squad which
delivers the service.

42
00:03:42,334 --> 00:03:46,614

The following types of environments are available for Development and test:

43

00:03:47,413 --> 00:03:51,326

A Sandbox environment is available and open to experiment.

44

00:03:52,136 --> 00:03:55,251

Development activities are done locally on a laptop.

45

00:03:56,081 --> 00:04:00,589

Test environments are created in the IPC shared virtual infrastructure.

46

00:04:01,399 --> 00:04:07,958

There are limitations to read; write, and capacity, for Development and Test environments as shown in

47

00:04:07,969 --> 00:04:08,526

the table.

48

00:04:09,226 --> 00:04:15,176

Acceptance and production environments are provided on dedicated infrastructure, based on stand-alone

49

00:04:15,176 --> 00:04:17,326

platform hardware (BMaaS).

50

00:04:19,236 --> 00:04:25,075

From an infra perspective, the Cassandra architecture provides high availability for a Keyspace.

51

00:04:25,675 --> 00:04:29,276

By default, an active-active configuration is provided.

52

00:04:30,106 --> 00:04:35,213

Please note that the application teams are responsible for data consistency in Cassandra.

53

00:04:35,823 --> 00:04:42,563

As an application team using the service, you can configure the consistency level (CL) for your Keyspace.

54

00:04:43,193 --> 00:04:45,588

This will affect your Recovery Point Objective.

55

00:04:46,428 --> 00:04:49,475
For development and test, this might not be crucial.

56
00:04:50,096 --> 00:04:54,913
This is why knowledge of Cassandra is a nice to have
for these types of environments.

57
00:04:55,553 --> 00:05:01,523
For acceptance and production, knowledge of Cassandra
is a 'must have', as users need to be aware of the

58
00:05:01,533 --> 00:05:03,975
consequences of configuration changes.

59
00:05:04,795 --> 00:05:10,826
When it comes to knowledge purely about Cassandra,
DataStax Academy has a large amount of content to

60
00:05:10,836 --> 00:05:13,163
explain all the fundamentals and beyond.

61
00:05:13,773 --> 00:05:19,338
The provided link in resources includes a training
path to get engineers started with Cassandra.

62
00:05:21,259 --> 00:05:28,151
Monitoring can be configured by sending the metrics
for the DataStax driver, to a team's own RTK2 Instance.

63
00:05:28,971 --> 00:05:34,613
A mandatory requirement is to add the Minions squad
as an observing team for your Ace team.

64
00:05:35,223 --> 00:05:41,039
Please add the following team code as Observer in
Ace: T01177.

65
00:05:41,848 --> 00:05:47,526
As part of the "How to" documentation, steps and examples
are provided by The Minions squad.

66
00:05:49,436 --> 00:05:56,666
As mentioned before: Cassandra is positioned as the
ING standard for Key-Value stores, session data and

67
00:05:56,666 --> 00:05:57,776

cache use-cases.

68

00:05:58,595 --> 00:06:01,651

Using KaaS for System of Records "(SoR)" should be avoided.

69

00:06:02,281 --> 00:06:08,111

For System of Records"(SoR)", traditional Database management systems should be used instead to ensure

70

00:06:08,120 --> 00:06:10,001

consistency of the source data.

71

00:06:10,831 --> 00:06:15,938

Please note that the application teams are responsible for data consistency in Cassandra.

72

00:06:16,548 --> 00:06:22,300

The configured consistency level is relevant to the Recovery Point Objective also called RPO.

73

00:06:23,131 --> 00:06:28,841

For development and test this might not be crucial, this is why knowledge of Cassandra is a nice to have

74

00:06:28,851 --> 00:06:30,738

for these types of environments.

75

00:06:31,368 --> 00:06:34,101

For acceptance and production, this is different.

76

00:06:34,731 --> 00:06:40,720

For these types of environment knowledge of Cassandra is a 'must have', as users need to be aware of the

77

00:06:40,731 --> 00:06:43,263

consequences of configuration changes.

78

00:06:45,164 --> 00:06:49,651

Cassandra within ING can offer snapshots depending on the use-case.

79

00:06:50,281 --> 00:06:54,801

For example, for selected customers with System of Record "(SoR)" use case.

80
00:06:55,601 --> 00:07:01,511
There is no point in time restore, the approach is
different to the traditional 'Backup and Restore procedure'

81
00:07:01,521 --> 00:07:02,401
for servers.

82
00:07:03,231 --> 00:07:10,271
For Acceptance and Production, an auto-snapshot is
in place to protect against user error (DROP or TRUNCATE

83
00:07:10,281 --> 00:07:11,551
a table keyspace).

84
00:07:12,341 --> 00:07:18,681
Before DataStax Enterprise upgrades, snapshot(s) are
made by the Cassandra team in case something does

85
00:07:18,691 --> 00:07:19,876
not go as planned.

86
00:07:20,696 --> 00:07:25,888
Currently, there are no methods in place to verify
that the snapshot contains all the data.

87
00:07:26,589 --> 00:07:32,968
Keyspace users do not have control of the snapshot
creation and/or restore, since these are manual procedures

88
00:07:33,039 --> 00:07:40,363
executed by the Cassandra Support team (Minions) and
also limited due to Linux rights, and users (NPA).

89
00:07:41,164 --> 00:07:48,214
A snapshot can be made by the KaaS team when a KaaS
user submits a request via a requestform, for example

90
00:07:48,224 --> 00:07:50,426
before a Cassandra datamodel change.

91
00:07:52,376 --> 00:07:53,600
Session use-case:

92
00:07:54,300 --> 00:07:59,713
Concerns services that store session information,

which is typically marked as short-lived-data.

93
00:08:00,313 --> 00:08:04,301
The data stored is only valid for x number of seconds or minutes.

94
00:08:04,920 --> 00:08:10,776
Most of these use-cases store their data for maximum 15 minutes, for example a login token.

95
00:08:11,606 --> 00:08:17,805
Therefore, a snapshot is not required, since the time to restore is greater than the time of the usage:

96
00:08:17,805 --> 00:08:18,551
lifetime.

97
00:08:19,351 --> 00:08:20,463
Cache use-case:

98
00:08:21,163 --> 00:08:27,053
Concerns services that cache data from other systems for performance reasons but are not the source and

99
00:08:27,063 --> 00:08:28,126
owner of this data.

100
00:08:28,726 --> 00:08:32,001
Losing this data might only have a performance impact.

101
00:08:32,811 --> 00:08:39,111
Recreation of this data can be done without recovery of a snapshot, for instance by requesting a new feed

102
00:08:39,121 --> 00:08:40,350
from the source of the data.

103
00:08:41,150 --> 00:08:42,913
Persistence of derived data:

104
00:08:43,743 --> 00:08:50,353
These services process, correlate and/or modify the data received from other systems but do not add new

105
00:08:50,363 --> 00:08:51,025

data to it.

106
00:08:51,635 --> 00:08:57,936
Recreation of this data can be done without recovery
of a snapshot, for instance by requesting a new feed

107
00:08:57,945 --> 00:08:59,175
from the source of the data.

108
00:08:59,975 --> 00:09:06,425
A snapshot should only be considered to speed up deployment
of a new instance for scalability or recoverability

109
00:09:06,436 --> 00:09:11,738
reasons, if rebuilding the data takes too long to
meet the availability requirements.

110
00:09:12,578 --> 00:09:14,438
System-of-Records use-case:

111
00:09:15,138 --> 00:09:21,448
For System-of-Records "(SoR)" use-cases, the KaaS
team have an automated snapshot creation at 08:00

112
00:09:21,448 --> 00:09:27,751
am UTC, which can be used as mitigation in case of
a data integrity issue.

113
00:09:28,550 --> 00:09:35,741
Due to the limitation of standalone or dedicated hardware
with local storage only, there is a maximum of 4 days

114
00:09:35,751 --> 00:09:43,463
for which the snapshot is retained before normal housekeeping
processes are impacted, for example compaction process.

115
00:09:45,394 --> 00:09:52,773
OS Patch Management and Cassandra upgrades are performed
by the Minions team as Cassandra is offered in Keyspace

116
00:09:52,773 --> 00:09:55,451
as a Service, similar to DBaaS.

117
00:09:56,251 --> 00:09:59,726
The following are all responsibilities of the service
provider:

118
00:10:00,526 --> 00:10:03,863
Ensuring functional correctness of the Cluster instance.

119
00:10:04,683 --> 00:10:09,839
Monitoring and acting upon Cluster availability, capacity and performance issues.

120
00:10:10,648 --> 00:10:16,989
With Keyspace as a Service, users of the service (Application Teams) only own the Keyspace layer.

121
00:10:17,619 --> 00:10:23,228
This means they are responsible for the functional correctness of their Keyspace instance and related

122
00:10:23,239 --> 00:10:23,651
data.

123
00:10:24,281 --> 00:10:30,791
This also means that teams are responsible for monitoring and acting on availability, capacity and performance

124
00:10:30,801 --> 00:10:32,751
related to the Keyspace instance.

125
00:10:33,571 --> 00:10:40,376
With regard to performance, there is a daily combi-test participation between 05:30 till 07:30 AM.

126
00:10:41,006 --> 00:10:45,264
This is not a hard requirement, but heavily recommended by the KaaS team.

127
00:10:45,894 --> 00:10:51,493
This is so you and the KaaS team can analyse the application's possible impact on the multi-tenant

128
00:10:51,493 --> 00:10:52,001
cluster.

129
00:10:52,601 --> 00:10:58,821
The scope and requirements for the tests are documented on the "How to" documentation page, as provided in

130

00:10:58,831 --> 00:10:59,926
the link resources.

131
00:11:01,826 --> 00:11:06,725
Cassandra clusters are placed in both datacenters
on highly redundant infrastructure.

132
00:11:07,335 --> 00:11:13,166
In case of failure on one datacenter, customer's
data won't be affected and will still be accessible

133
00:11:13,215 --> 00:11:14,513
in the second datacenter.

134
00:11:15,353 --> 00:11:20,275
IPC however, does not provide automatic traffic switching
in case of a failure.

135
00:11:20,905 --> 00:11:26,715
This means that it's the customer's responsibility
to implement connections to both datacenters on application

136
00:11:26,725 --> 00:11:30,438
level, in order to reduce risk against unavailability.

137
00:11:31,258 --> 00:11:37,528
To allow KaaS users to prepare and test out settings
in case of a real disaster, there are several moments

138
00:11:37,598 --> 00:11:42,126
where the KaaS team uses the practice of turning off
Cassandra in one datacenter.

139
00:11:42,826 --> 00:11:50,586
Keyspace as a Service, also participates in Production
DR exercises organised by the "BCM department" every

140
00:11:50,595 --> 00:11:51,913
six months for Production.

141
00:11:52,523 --> 00:11:58,188
On a weekly basis there are possibilities to test
DR capabilities for Test and Acceptance.

142
00:11:58,778 --> 00:12:04,698
Details about the times for the weekly tests are

available as part of the 'How to documentation' in

143

00:12:04,708 --> 00:12:06,300
the provided link resources.

144

00:12:08,201 --> 00:12:11,400
The following are possible ways to perform a DR test:

145

00:12:12,220 --> 00:12:16,251
The first option is to Turn off native transport,
a soft DR.

146

00:12:17,081 --> 00:12:23,161
This results in an Active-Passive environment where
queries are stopped being served in one datacenter,

147

00:12:23,161 --> 00:12:26,363
however replication remains active across datacenter.

148

00:12:27,193 --> 00:12:30,476
This is not an actual simulation of a datacenter failure.

149

00:12:31,066 --> 00:12:37,488
In this scenario the risk for the bank is lowered,
especially during the DR test on Production environments.

150

00:12:38,308 --> 00:12:42,600
The second option is to turn off the Cassandra service,
a real DR.

151

00:12:43,400 --> 00:12:48,175
In this case the Cassandra service is stopped on all
the nodes in one datacenter.

152

00:12:48,805 --> 00:12:52,213
This is the closest a simulation gets to a real outage.

153

00:12:54,144 --> 00:12:56,039
This concludes the KaaS topic.

154

00:12:56,668 --> 00:13:00,326
I hope you enjoyed and I look forward to seeing you
in the next video!

Intro3a

0:03

But let's start with the basics.

0:05

What's an application container?

0:07

And a container can already be a confusing term.

0:10

In the past, we call the VM container or Tomcat or Web Sphere.

0:15

But an application container consists of the entire runtime environment, the application plus all its dependencies, libraries, configuration files, all blindled in one package.

0:30

You can move easily from depth to test if you move your container from depth to test, it's much much simpler than in the past.

0:39

We used a container like Docker already a long time in the pipeline, for example to test if the application works.

0:49

But Docker, the most popular application container, combined with Kubernetes as an orchestrator, as we will see, it just reduces the burden of deployment.

1:02

It's a cost reduction.

1:04

Maybe another way to put it is containers virtualize at the OS host level as hypervisors like VMware virtualize at the hardware level.

1:15

Remember with VMware that you're blissfully unaware on which hardware you're running.

1:21

Now you're unaware on which host you're running.

1:25

You don't have to manage a host.

1:29

Let's repeat this virtualization with hypervisors.

1:33

Traditionally we had our hardware in the data centers computers.

1:37

If I needed a new server, I had to wait three months before a new hardware, a new server was installed in the data center.

1:47

Now we have a big pool of hardware and on top a hypervisor is running and like VMware and VMware will assign on demand resources to your Vms, to your virtual machines.

2:04

You don't know on which hardware you're running, you don't know which CPU or memory is used by your VM, It's abstraction.

2:16

But now I have virtualization at a higher level.

2:21

I don't know on which OS on which host I'm running.

2:26

I don't need a VM, I don't need to provision a VM, I just can deploy my container in the Kubernetes cluster and I don't know on which host I'm running.

2:40

It's managed by an orchestrator like Kubernetes and Kubernetes Open Source software Lite.

2:48

Docker is the most popular orchestrator for your containers, so it manages your containers.

2:55

It offers several features like out of scaling.

2:59

It can spin up extra containers if your load increases.

3:03

It can do automated rollouts and rollbacks, service discovery, load balancing, storage orchestration or secret and configuration management.

3:15

Self healing.

3:16

Well you can say auto scaling is self healing and it can more efficiently use the resources, CPU, memory etc.

3:27

So it's a cost reduction and it just makes life easier for you as a DevOps engineer.

3:34

And I must say not all features can be used at ING.

3:37

It depends on your application.

3:39

If you're a TPA application, MARAC or using a touch point service, mess or not, so it depends on your application which features can be used.

3:53

And application containers introduce the concept of immutable servers.

3:58

You cannot change your server, your container, your application.

4:03

You can only throw away order a new one, throw away, reinstall your container.

4:10

You need a mature pipeline.

4:12

You cannot change a docker image in production.

4:16

The only way to change your code is go through the pipeline.

4:21

This is also much safer.

4:25

You know the risk if you change something in production.

4:29

I hope you do.

4:31

Let's say you have a major incident.

4:34

You have to change something quickly.

4:36

You log on to your production server, change some property file or even your code and what happens a couple of weeks later or a couple of days later?

4:47

The next version?

4:49

Already in the pipeline?

4:50

Already in acceptance?

4:52

Maybe the next version is deployed in production and you've lost your change because your change was only done in production and not on the next version which was already in the pipeline in dev, test or acceptance.

5:09

So it's much safer.

5:12

Yes, you need a mature pipeline, but you can only change your docker container by going through the pipeline.

5:21

No manual changes.

5:23

But this is also a huge advantage for security.

5:27

You know the burden of all those security controls in production, evidencing, etc.

5:35

Having no access can simplify security procedures that can speed up the usability of your newly deployed application container.

5:45

So docker containers offer several benefits, shorter time to market.

5:50

I can more easily move from dev, test, acceptance to production an increased portability between environments but also to another infra provider.

6:04

Kubernetes is one of the most popular orchestrators for Docker containers and it's available in almost any public cloud.

6:13

It offers increased availability, security because of using this orchestrator and of this immutable servers it has a mature ecosystem.

6:25

Docker, Kubernetes are the most popular ecosystem for application containers and ICHP, the ING container hosting platform built using Kubernetes and Docker is our cloud native platform.

6:42

It's designed using the cloud native design principles, the seven commandments from Google so Google it.

6:50

It's built for microservices, cloud native applications and the most important characteristics.

6:58

It must be immutable, stateless, short living and maybe you remember that the term idempotent.

7:07

Please leave you the learnings where we already discussed idempotent.

7:13

But what I'm trying to say is ICHP is not a good fit for all applications if you want to use the full potential of Kubernetes which can stop start your application out of scaling rollouts, rollbacks, it can change routing, etc.

7:32

But can your application withstand any stop start?

7:37

Can it handle a stop start?

7:40

Is your application stateless?

7:43

If your container is destroyed, is there data in your container If you're restarted?

7:50

Is business logic affected?

7:53

Do you have double processing or did you lose a transaction?

7:58

And to avoid misunderstandings with the term stateless.

8:02

With stateless I mean the application cannot depend on data in the container.

8:07

Of course you can use data outside of the container.

8:11

Outside of the Kubernetes cluster.

8:14

You can use the ABAS Oracle or the ECSS 3 OPX Store or Kafka or maybe Cassandra CAS, Key space as a service.

8:24

But please review the design principles mentioned in the High Availability module about data principles like one data store, 1 application and the data store being tightly coupled to one application.

8:41

Or think about sure where do I store my data?

8:46

Is it a system or of record?

8:48

Yes or no.

8:50

So let's take a look at a couple of examples when it's maybe not such a good idea to migrate to ICHP.

8:57

Let's say a batch application.

8:59

If a batch application crashes in the middle of processing of a file, can it recover that?

9:05

Does it start all over or does it start where it left off?

9:11

Do you have a data inconsistency, double processing or an application that depends on data stored in the container?

9:21

If the container is destroyed, the data is lost.

9:25

Can the application recover from that?

9:29

Or a commercial off the shelf application?

9:33

An old fashioned monolithic application?

9:37

Can it be auto skilled?

9:39

Can you dynamically change routing?

9:42

How big is the container?

9:44

How quickly can you stop start this container?

9:48

Is it immutable?

9:49

Are all changes done using a pipeline And consider network access and we'll talk about networking later on in detail.

9:59

But within a Kubernetes cluster, there are no firewalls.

10:03

So if you deploy your API within a Kubernetes cluster, is the endpoint reachable for the other APIs in the Kubernetes cluster on the network level?

10:13

Now with the TPA application, TPA applications are protected using mtls mutual tls.

10:20

It means you cannot connect on the network level with this TPA API, only with a valid certificate.

10:30

But for non TPA applications, maybe you only have authentication on the network level you might be reachable.

10:39

Yes, you need to create routes within a Kubernetes cluster, but there is no accounting, logging, reporting on those routes and endpoint isolation.

10:52

Does your application offer endpoint isolation?

10:56

Please revisit the modules about endpoint isolation.

11:01

When an API contacts your API, connects to your API, can it hop from your API to another API?

11:11

That's a violation of endpoint isolation.

11:15

This is these factors are extremely important for external phasing applications.

11:22

So to be a good fit for ICHP you have to be immutable, short living, stateless item, potent and short living.

11:31

Maybe not like Google where they redeploy every day, but we redeploy every month, at least for security patching.

11:39

And in the future with microservices we want to redeploy more often.

Design2

0:10

Now let's have a closer look at Docker.

0:12

Kubernetes at ING, the ING container hosting platform, its infrastructure and design principles.

0:21

So in each data center at ING dcrwpr we have a Kubernetes cluster.

0:27

Well, in Nonprot actually we only have one cluster, but this is an example of production.

0:34

So in each data center we have a kubernitus cluster.

0:37

And a kubernitus cluster consists of an infra node and worker nodes.

0:43

The worker nodes are your actual hosts, but you remember you don't need to be aware about them.

0:49

You don't know which worker node you're running, and the infran node offers all kinds of generic services within the kubernitas cluster.

0:59

Maybe one of the most important routing implicitly all traffic coming from outside the kubernitas cluster.

1:09

So all traffic coming from private networks within the IPC, all traffic is routed through this infran node.

1:18

All incoming traffic that is going through this infran node.

1:22

And remember, we'll talk about networking in detail later on.

1:28

And in the IPC portal you can create your own namespace.

1:33

A namespace is a security boundary, it's a mechanism for isolating groups of resources.

1:40

It's a unit for sizing, capacity management and we use a design principle 1 namespace, 1 product ID or purpose.

1:51

Your PTA, Business Standard opening can change this product ID, but I'll get back to that.

1:58

And within your namespace you can deploy what we call a pot.

2:03

A pot is the smallest deployable unit and within the pot is your container and usually it's one container.

2:13

And maybe the exception is very tightly coupled services like a sidecar or a service mesh.

2:20

And then you might have two containers in your pot and within your namespace one business application is preferred or one functional pot.

2:31

Of course you have multiple APIs copies of the same pot, same API for high availability, but you have one API, one business function.

2:42

This is again the discussion 1.

2:44

Namespace 1 product ID.

2:49

Let's now discuss this concept of product ID purpose.

2:53

The goal of this concept is to make life easier for you to enable migration of a product ID from one owner to another.

3:03

To enable the migration of an API from 1 squat to another without destroying all the resources, removing all your resources and recreating them, we want to enable an automated migration of a product ID from one team or one owner to another, including all physical assets, hosts, network, firewall rules, but also access controls, the Mpas, solar, et cetera.

3:42

For example, your Business Standard admin at the moment can move a namespace to another product ID.

3:52

So this concept of product ID is again about changeability, future changes, changing of ownership of an API to another squad.

4:02

So maybe best practice is having one application and one product ID.

4:07

But what's an application?

4:09

An application can maybe be multiple tightly coupled services.

4:15

If this application moves to another squad, these services will always remain together.

4:22

They will always move together in a future change.

4:27

But maybe you also know security controls or processes like the business impact analysis, your CIA rating, but solar your Mpas.

4:40

These are bound to an ID to the ID in service.

4:45

Now in the CMDB your business CI.

4:50

So with the design of product ID, you have to keep in mind the design of the CMDB, the design in ServiceNow.

5:01

Do you have one business CI and multiple application components below it?

5:08

So an API being an application component or you create a business CI per API.

5:17

This will affect future changes.

5:22

So let's say you have one functional pot, one API in your namespace.

5:27

For high availability reasons, you have to set the number of replicas a parameter in your pipeline to at least two.

5:36

In that case you get 2 pots per cluster and if your TPA application using the Kings Rd.

5:43

pipeline it will be automagically provisioned on red and Blue.

5:49

If you don't know what red and blue is, please revisit the High availability module.

5:55

But for TPA applications using Kings Rd.

6:00

if you set your replicas at at least two, it will be provisioned all red and blue and you have hardware failover and you have to do this provisioning per cluster.

6:12

So you in the pipeline you will also deploy to the second cluster to the second data center and also with the replicas set to at least two.

6:25

And at ING we already have numerous namespaces running in the kubinitis clusters.

Ichpnetw 1

0:03

In ICHP, our Docker Kubernetes environment, networking works a little bit different. First, we have an ICHP Kubernetes cluster per data center. So we have two clusters and you can create a namespace over the two clusters. Let's first have a look at the incoming traffic for ICHP. An incoming traffic for ICHP is called Ingres.

0:30

Let's say you have a consumer API running in a private network in the IPC, so not in ICHP. This consumer will retrieve the endpoints from service discovery, the API endpoint discovery. But the point is the endpoints which are retrieved from service discovery are the endpoints from the ingress routers. The endpoints are not.

0:58

From your port from your API. All incoming traffic for ICHP is routed through the ingress router and the ingress router will forward the request to your port to your API. A namespace within ICHP is not a traditional network zone, it does not have a firewall within ICHP.

1:27

Network policies and routes are used and these are managed by TPA, by the Kings Rd. pipeline. There is one firewall for the whole ICHP cluster and of course for each private network outside of ICHP there is a firewall. So if you want to allow traffic from a network zone, a private network outside of ICHP to your namespace.

1:53

That the ingress routers must be defined as destination. And please always check the latest network info of ichp on Confluence or the Forge. So remember all traffic coming from outside ICHP has to pass through an ingress route. Let's now have a look at the outgoing traffic, so from your API in ichp on a port.

2:20

To an API in a private network outside of ICHP, outgoing traffic from ICHP is called egress and again for TPA API, the endpoints are retrieved from the API endpoint discovery. But to enable automation of the outgoing traffic using the self-service portal, we defined A namespace as a private network in the self-service portal.

2:50

And the IP addresses of your namespace. Your ports are not that not. That means your outgoing IP address is changed. Maybe if you are able to log on to the port, you discover the IP address starting with 172. That's not the IP address that is used on the network in the firewall, Your IP address is manipulated.

3:18

It's changed to a IP address starting with 10. Please do an NS lookup and check the documentation how to retrieve this. Not at IP address, but if you're communicating with network engineers. If you are in the war room and you have trouble with an outgoing flow, so from your API and ICHP.

3:44

To a private network or a network zone outside of ICHP. Remember your outgoing IP address. Your egress IP address is manipulated. It's not that. Be sure that you know the correct IP address. And yes, for incoming traffic ingress, remember the ingress routers are used.

4:12

And by the way, in ICHP version two there are multiple ingress routers for different purposes for TPA, non TPA, et cetera. Please check the documentation on the forge or conference. Let's look at an internal call. So within ichp, one API deployed in a namespace calling another API in another namespace.

4:39

For TPA APIs, the endpoints are retrieved from the API endpoint discovery and if this consumer API chooses an instance of the provider API in the same cluster, it will be a direct call. It will go from one pot to the other, but a network policy or route must exist and this is created and managed by King's Road and no firewall is involved here.

5:09

But if the consuming API chooses an instance of the provided API in the other ICHP cluster, then a call has to be made to the other data center to the other Kubernetes cluster and you have to pass through the firewall. But these firewalls are open by default between the ICHP clusters, but this call will go to the ingress router again.

5:36

And the ingress router will forward the request to the API to the provider API running on my port. And again a network policy and a route must exist which is managed by King's route.

00:00:11,499 --> 00:00:12,138

Hello!

2

00:00:12,938 --> 00:00:19,238

In this module we'll briefly go over the options you have with ICHP from an infrastructure perspective.

3

00:00:19,838 --> 00:00:27,164

You can use this information to order, monitor and use ICHP to make your application as reliable as required.

4

00:00:27,993 --> 00:00:33,843

This module will not go into full details of everything offered, but a slide deck with more information is

5

00:00:33,853 --> 00:00:40,264

provided, together with important links to ICHP documentation where you can find the latest information.

6

00:00:42,184 --> 00:00:46,263

With ICHP both horizontal and vertical scaling are possible.

7

00:00:46,883 --> 00:00:52,874

With horizontal scaling, more replicas are added, while with vertical scaling more resources are added

8

00:00:52,883 --> 00:00:54,239

to existing replicas.

9

00:00:55,339 --> 00:00:58,301

The preferred method is the use of horizontal scaling.

10

00:00:58,901 --> 00:01:03,889

You can perform this manually, by adjusting the number of replicas in your deployment config.

11

00:01:04,489 --> 00:01:09,226

Or you can scale automatically by using the function "horizontalpodautoscaler".

12

00:01:11,016 --> 00:01:15,826

In this example we see the settings for the function "horizontalpodautoscaler".

13

00:01:16,326 --> 00:01:20,439

Auto scaling is based on CPU and or memory metrics.

14
00:01:21,039 --> 00:01:25,576
In the example you see a minimum of two, and a maximum of 10 replicas.

15
00:01:26,676 --> 00:01:30,551
With both methods enough resources should be available within your quota.

16
00:01:31,151 --> 00:01:36,064
You can adjust the quota in the cloud portal if you are using ICHPv1.

17
00:01:36,664 --> 00:01:39,388
If you are using ICHPv2,

18
00:01:39,668 --> 00:01:44,551
your quota will be automatically increased and decreased with the function "QuoteAutoScaler".

19
00:01:45,171 --> 00:01:48,576
Note that decreasing the quota is limited to once per hour.

20
00:01:50,496 --> 00:01:56,106
The easiest way to make your deployment resilient and decrease the number of avoidable major incidents

21
00:01:56,195 --> 00:01:57,701
is to use the kingsroad.

22
00:01:58,801 --> 00:02:04,125
If you're not able to use the kingsroad, make sure to use the Kubernetes resilience features.

23
00:02:05,256 --> 00:02:11,486
These features are: Canary release, readiness and liveness probes, and make sure pods are placed in

24
00:02:11,495 --> 00:02:13,763
both Red and Blue fault domains.

25
00:02:14,883 --> 00:02:20,813
With a canary release you expose a small group of users to your new software version before you fully

26
00:02:20,813 --> 00:02:22,638
commit to the actual production deployment.

27
00:02:23,248 --> 00:02:27,925
Typically between 10 to 30% of the users are redirected
to the new version.

28
00:02:28,556 --> 00:02:30,651
This way you can validate stability.

29
00:02:31,751 --> 00:02:36,038
With a readiness and liveness probe, you validate
the running state of a pod.

30
00:02:36,638 --> 00:02:42,228
The liveness probe checks if a pod is still alive
and will kill the container if it isn't, while the

31
00:02:42,238 --> 00:02:45,976
readiness probe validates if a pod is ready for service
requests.

32
00:02:46,586 --> 00:02:51,050
It can be used to signal the proxy that this pod
should not receive any traffic.

33
00:02:52,181 --> 00:02:55,875
For greater resilience you want Pods to be as separate
as possible.

34
00:02:56,505 --> 00:02:59,163
This can be achieved by using anti-affinity.

35
00:02:59,783 --> 00:03:07,363
An example how to set-up anti-affinity settings can
be found on the ICHP documentation pages, or by visiting

36
00:03:07,373 --> 00:03:12,163
the Kubernetes or Openshift documentation websites
for more details.

37
00:03:14,084 --> 00:03:20,351
Resource metrics for all namespaced objects are automatically
collected by the "Cluster Monitoring Prometheus".

38
00:03:21,441 --> 00:03:26,671
A ServiceMonitor object (prometheus-operator) in the
Namespace enables scraping of prometheus metrics directly

39
00:03:26,681 --> 00:03:28,313
from the application pods.

40
00:03:29,313 --> 00:03:37,226
"ICHP User Workload Prometheus" watches for ServiceMonitor
objects, and adds them to its scrape target configuration.

41
00:03:38,346 --> 00:03:43,089
Metrics are collected in the "central Prometheus"
and the "user-workload Prometheus".

42
00:03:44,189 --> 00:03:49,513
The metrics scraped through your servicemonitor can
be federated from the "user-workload Prometheus".

43
00:03:50,614 --> 00:03:56,564
A namespace access token is required, the token is
automatically created for all namespaces.

44
00:03:57,164 --> 00:04:00,339
Look into the secret: "prom-tenancy-access-token".

45
00:04:01,439 --> 00:04:08,239
In the query, a namespace label filter must be provided,
only the same namespace as the token is allowed.

46
00:04:09,339 --> 00:04:12,464
You can also get this token in acceptance and production.

47
00:04:13,564 --> 00:04:15,564
The secret name is whitelisted.

48
00:04:16,214 --> 00:04:18,663
Only /federate endpoint is supported.

49
00:04:19,784 --> 00:04:26,433
To protect the central "user-workload-Prometheus"
against abuse and resource overflow, some limits are

50
00:04:26,443 --> 00:04:29,264
enforced automatically for the ServiceMonitor object.

51
00:04:30,384 --> 00:04:33,838
Alerting on Consumer Namespace metrics is not supported.

52
00:04:34,438 --> 00:04:38,776
The concept is, that alerting is configured within
RTK Prometheus.

53
00:04:40,576 --> 00:04:43,326
ICHP is stateless per definition.

54
00:04:43,936 --> 00:04:50,345
Containers are not intended to use persistent storage
on the platform, therefore back-up and restore capabilities

55
00:04:50,616 --> 00:04:52,963
are not provided as part of the service.

56
00:04:54,073 --> 00:04:59,763
As a result users of the service have to implement
backup and restore strategies for their persistent

57
00:04:59,773 --> 00:05:04,601
data on services outside of ICHP, like DBaaS for example.

58
00:05:06,521 --> 00:05:12,970
The best way to apply patches is to either use the
kingsroad, or make use of canary releases as mentioned

59
00:05:12,980 --> 00:05:14,275
earlier in this training.

60
00:05:16,206 --> 00:05:19,988
This concludes the ING Container Hosting Platform
topic.

61
00:05:20,618 --> 00:05:24,276
I hope you enjoyed and I look forward to seeing you
in the next video!

Lb1a-1

0:03
Hi and welcome back.

0:05

So we talked about availability and if we talk about availability, we have to talk about load balancers.

0:12

So we have different load balancers and we have hardware and appliance like F5 at ING, although contained, you can't use it anymore.

0:22

And we have software load balancers like an engine X for example.

0:27

You can order your own engine X in the IPC or a client library like the physical client.

0:34

A client library is part of the Mark framework.

0:38

It's also present in the sidecar at a touch point service mesh and we'll talk about that in detail later on.

0:47

That's by the way the preferred solution.

0:50

Don't depend on any middleware or infra, solve it yourself.

0:56

Use a client side library.

0:59

So why do I need a load balancer?

1:02

A load balancer offers availability if one is down that you can configure a health check or keep alive in your load balancer.

1:10

If one instance is down, you switch to the other one.

1:13

The load balancer will do the failover, but it's also very important for scalability.

1:20

If you can't handle the load, you can just add instances and the load balancer switches to the other extra instance.

1:31

And I just mentioned the engine X.

1:33

Be careful when mentioning the Engine X Engine X, we have different implementations of the engine X at ING.

1:43

So in the IPC you can order your own engine X as a load balancer.

1:48

But we use also the authenticating proxy or sometimes called the API Gateway or we have the security gateway.

2:00

We have many names for the same thing, right?

2:03

It depends on context, it depends on who you're talking to.

2:06

And we also have an engine X as a forwarding proxy.

2:10

So please be careful.

2:13

So yes, I am sorry, but you know in IT, in IT many many terms depend on context.

2:24

We also use a different terminology, an LTM and a GTM, a local traffic manager and a global traffic manager.

2:34

So for a local traffic manager, the traffic is divided within one data center.

2:42

It's load balanced inside one data center.

2:47

A global traffic manager can do load balancing over the two data centers, so an LTM and an engine X.

2:55

By the way, they can also be extended for some security functions like SSL offloading.

3:03

Be careful at ING, we have a rule all traffic must be encrypted only inside your own network zone.

3:13

Unencrypted traffic is allowed.

3:16

I'll talk about that later with network segmentation and in the Network Basics.

3:23

A GTM or Global Traffic Manager is an extension of the DNS.

3:28

You probably know what a DNS is, right?

3:31

It translates your host name or the domain name in your URL to an IP address so it routes.

3:39

It takes care of the routing and points to an IP address and you can manipulate this address or you can have multiple, even multiple IP addresses.

3:51

The GTM you can configure active passive and active active so you can point to one IP address and if this one fails you switch to the other one.

4:04

Or you can configure the GTM to let's say I want 50% of traffic going to 1 address and 50% going to another address.

4:15

The point is the failover.

4:17

The failover can take some time because it's DNS that you have to keep in mind.

4:23

Clients have DNS cached, so the DNS address, the IP address is cached.

4:30

You can tune this with your time to live, for instance on your servers.

4:35

But what about your clients?

4:37

So you're not sure how quickly the failover takes place.

4:42

If one address fails, it can take minutes before the traffic will be sent to the other side.

4:51

So the problem with the global traffic manager or a DNS is this DNS cache and this time to live.

5:01

You don't have this problem if you use for instance your own engine X in the IPC as a load balancer as you will see, or with client side load balancing that we will also see a little bit later.

5:16

Now with client side load balancing or an engine X, you're just pointing to all instances of your API for example.

5:25

So you're pointing to 4 instances, 2 in each data center and all traffic is active, active, sent to 4 instances and depending on timeout settings etc.

5:40

If one fails that you just send the traffic to the other endpoints.

5:45

There is no DNS cache that you do not have to wait.

5:50

So keep in mind with the global traffic manager you can choose two configurations, active, passive or active active, but the failover in both instances.

6:01

With a passive instance, if one is down it can take minutes to switch.

6:06

But also with active active, if one goes down and all the traffic has to be sent to 1 endpoint, it can take time.

6:15

It depends on tuning of your time to live and client settings.

6:21

In the IPC there's no generic local traffic manager available and F5 can't be used anymore.

6:28

But luckily you can order your own engine X, and we have different flavors, a single instance or even high available.

6:38

So you can order in the same network zone as your application.

6:43

Your own engine X.

6:45

You can make it high available and maybe using a global traffic manager you can handle the failover over the data centers.

6:54

Or maybe your requesters and consumers can handle the failover between the data centers themselves.

LB2-1

0:10

If your API is using your Mac framework, you'll get client side load balancing and the dynamic service discovery which we will discuss.

0:21

But also touch points offers a touch point mesh, a service mesh solution.

0:29

Then you get a sidecar, a separate process next to your application on the same machine or in the same port in ichp docker and it offers the TPA functionalities.

0:42

So if we're talking about TPA enabled applications, we're talking about the APIs built with monoch but also applications with a sidecar.

0:55

So the sidecar and TPA enabled applications that they offer client side load balancing, the dynamic service discovery and the security framework mutual TLS, but also tracing.

1:08

Very important.

1:10

But a touch point mesh is a service mesh solution, It's a cloud native solution.

1:18

And if you go to the website of Cloud Native Computing Foundation that you'll find a definition of cloud native.

1:27

And for instance it's about containers, docker, a service message, microservices, immutable infrastructure.

1:36

But the whole point is, and this is quite an ulterum, it's about loosely coupling, it's about loosely coupled.

1:44

I want to be able to change something, It's about changeability without dependencies.

1:51

I want to change my API.

1:54

I want to add an extra instance without a change in the middleware infra or even without a change in my client in my request or so.

2:07

What is a service mesh exactly?

2:10

A service mesh is an inter service communication infrastructure.

2:15

It handles your east West traffic.

2:19

We talk about East West and north-south NS is coming from outside the data center.

2:27

Coming in East West is all the communications between the applications in your data center and a sidecar next to your application can handle, you could say all your Nfrs nonfunctional requirements.

2:44

So it handles your connection logic, it can handle your logging, it can handle your security, your mutual TLS or for touchpoint, excess tokens, etc.

2:57

It can do a lot of things, but you could say you separate your non functionals from your functionals.

3:04

Your application can solely concentrate on the functional requirements.

3:10

That API is now only doing what it should do, the business requirements and you have a separate process, a separate deployable unit have for your nonfunctionals.

3:25

So in the past we had many monolithic applications.

3:30

As simply said, it's an application doing everything, showing the presentation layer, the screens for the user, going to a database, doing all the transactions in one application.

3:43

But now in the new world, we have microservices APIs and the traffic between our applications.

3:50

Microservices has increased with a monolithic application it was based basically north-south traffic and if you compare a service mesh with a library with a library you need to implement it in your application.

4:08

The controls in the squats in the teams.

4:12

But now with a service mesh I can decouple my nonfunctionals, my connection logic, my security from the functional API and I have a separate control plane you could say for my nonfunctionals, for my service mesh you might say.

4:32

It can enable platform teams for example.

4:36

So this service mesh is a separate process, a sidecar next to your application and it enables us to implement standards much more easily.

4:48

It can be deployed independently and all my non functionals like security like tracing and all the other stuff can be in this separate process.

5:00

And I can change this independently of your functionality in your API.

5:06

The advantages of using a service mesh are clear.

5:10

I can now implement my standard generic functions within my company in this sidecar.

5:17

You could say my commodity features I can implement in this sidecar.

5:22

I have an independent deployment.

5:25

It can solve my problems for tracing, blocking, security.

5:30

Moreover, I can also if needed.

5:35

May be difficult, but it is possible that I implement my sidecar in a different programming language than the functional API.

5:46

TPA enabled applications can use the Dynamic service discovery.

5:53

It's in the Mark framework and it's in your sidecar.

5:57

It's very important for changeability and flexibility.

6:01

I simplified a bit maybe, but what's the basic functionality?

6:06

If your API is a TPA enabled API and at runtime it starts up, it will register itself in the dynamic service discovery.

6:20

The dynamic service discovery knows the server name has the server name of your provider API for example.

6:30

The next step if a consumer or requester wants to connect to a provider, he will contact to the dynamic service discovery and he will get the addresses, the server names from the dynamic service discovery.

6:50

Usually that's four.

6:52

Remember your provider API is hopefully highly available, so 4 instances, 2 needs data center, so the consumer will retrieve 4 addresses from the dynamic service discovery and after retrieval discovery he can call the provider using those server names he has dynamically retrieved from the dynamic service discovery.

7:19

But I hope you see the big advantage here.

7:22

If your provider API adds an extra instance, you don't have to do anything.

7:29

Your consumer will retrieve the new address from the dynamic service discovery and if there is a client side load balancing in this consumer and there is in TPA enabled applications the client, the consumer will do load balancing using those dynamically retrieved addresses.

7:56

You don't have to change anything in a load balancer or in a property file of the consumer.

8:04

The consumer will retrieve the addresses dynamically.

8:10

Check the details you may be to if there is a new address of a provider.

8:18

Maybe you have to stop, start or tune to get those new addresses.

8:24

Get the details as usual.

8:27

You will find the links For more information on the same page on the same location where you found this movie.

8:37

And it's not only for API to API calls but also our security proxy, the API Gateway, the engine X be carefully remember but the security proxy authenticating gateway will also do load balancing and will also retrieve from the dynamic service discovery that the server names of the provider API.

9:05

So you do not need to change the gateway or the engine X, it will dynamically retrieve the server names from the dynamic servers discovery.

9:19

So I hope the benefits for TPA enabled applications using client side load balancing and the dynamic servers discovery are clear.

9:28

The squat is in control.

9:31

You are in control of the configuration of the dynamic servers.

9:35

Discovery enough for consumers, requester For configuration of client side load balancing you have dynamic routing.

9:45

If an extra instance is added, it will be added to the pool of your provider, sorry requester, and it will get the load.

9:56
Dynamic routing.

9:57
No dependency on infra.

10:00
I don't have to ask my infra provider or someone who is managing the middleware to add an extra pool member.

10:08
You could say in the pool in the cluster of your APIs an extra server name if you add or provision an extra machine.

10:20
And please keep in mind that the touch point service mesh is also specifically built for culture applications for commercial off the shelf applications.

10:31
So if you are using software from a vendor, you can deploy next to it a sidecar.

10:40
And the sidecar has many advantages.

10:43
We just discussed the client side load balancing and the dynamic service discovery, but also the mutual TLS and maybe you think that's all security stuff, but keep in mind.

10:56
And now the requesters will register using the developer portal.

11:00
You know your requesters.

11:03
And that's the other big advantage, observability, tracing.

11:09
You can trace a message through your API chain, that's if you're in the war room, that's a huge advantage.

11:19

Tracing is in the sidecar so there are many advantages.

11:25

In the future the firewall automation will be in place so firewalls will be opened automatically for your requesters.

11:36

So if you have a commercial off the shelf application please use the touch point mesh.

11:46

And if it's not possible please consult your solution architect or your BTA.

11:53

So if you don't you use the touchpoint service mesh you have your old fashioned calls application.

11:59

Difficult to integrate with the other ING applications in the ING infrastructure, but after then you have a modern you could say cloud native application.

Lb4-1

0:02

Load balancing at ING can be quite complex.

0:06

You've seen we have multiple solutions, multiple flavors, so be sure that you know which one is used in your production environment and maybe create a physical design picture.

0:21

If you're in the war room and you don't know how load balancing is done, you don't know how you're high available.

0:28

So we have multiple solutions and maybe I didn't even mention them all.

0:33

There are multiple variations possible, so please know for your solution how high availability and load balancing is configured.

0:46

And remember we prefer for new solutions for new applications client side load balancing.

0:54

So the key takeaways from these sessions are first of all use client side load balancing.

1:01

That's preferred.

1:03

Use the dynamic service discovery for dynamic routing.

1:07

A huge advantage for changeability.

1:10

Be smart when mentioning an engine X, we have multiple engine XS for different functionalities.

1:17

Use the touch point mesh sidecar for your commercial off the shelf applications or your known TPA APIs.

1:27

A global traffic manager is active passive.

1:31

It takes time for the fail over, you have to think about the DNS cache.

1:36

You have to think about the time to live and remember the F5 which is currently only used by infrastructure and maybe by the old APIs.

1:47

But you cannot use the F5 anymore.

Api-chain

0:05

So let's take a closer look at API chain complexity or application chain complexity.

0:13

If you have a request reply chain, and especially if you're doing a transaction in this chain, and with a transaction in this context I mean an insert, update, delete.

0:25

So somewhere at the end of the chain I'll do an insert, update, delete of persistent data, let's say in a database.

0:34

Especially if you're doing a transaction, it can be quite complex.

0:39

You have to tune your timeout settings.

0:43

An incorrect timeout setting in your chain might lead to unexpected results.

0:50

Let me give you an example.

0:52

If you look at this chain and this is a very simple chain, just four APIs.

0:57

I know chains of more than 30 APIs.

1:01

To be clear, 30 microservices.

1:05

But this is a simple example with four APIs.

1:08

So a request comes in at the top API.

1:11

This API calls another API, the second API.

1:15

It's an exaggerated example, but the second API needs let's say 2 seconds of processing time.

1:22

It calls Another API also needs 2 seconds of processing time, so we're already at 4.

1:29

The counter of the first API that the timeout counter is already at 4 seconds, and let's say the third API that will also call the next one, which also takes 2 seconds.

1:42

So now we're at 6 seconds in total, and at the bottom of the chain a update of the database takes place, which usually is the heaviest in performance.

1:54

So in this example it says it takes 5 seconds.

2:00

Well $5 + 2 + 2 + 2$, that's 11 seconds.

2:06

So now for the first API, you've reached the timeout setting.

2:12

What will it do?

2:14

Let's send back to the user an error message.

2:18

If let's say there's a front end on top of this chain and this API sends back the error message, what will the user do?

2:29

Maybe the user will reenter this address change, let's say it's a change of his address and then I get a double transaction.

2:39

So the time out settings in an API chain and an application chain very very difficult that we talked earlier about being idempotent.

2:51

This is one of the reasons why you need to be idempotent.

2:56

This is one of the reasons why you always need to test the non nappy flow, because this is still a very simple example.

3:05

As you already know there is not one instance of my API.

3:11

I have multiple instances of my API.

3:15

Normally my API is deployed 4 * 2 times in each data center and of course that's for each API.

3:25

So it's already getting a little bit more complex and if you even know your APIs that do realize also in which network zone, which firewall is being used.

3:39

If you are in the war room and there's an issue with your API or your API chain, a certain functionality, do you know all the network zones?

3:50

There may be a firewall, there's an error in a firewall, the firewall is closed, or even do you know how the load balancing is done?

3:59

Maybe there's a load balancer in between?

4:01

I've seen many logical designs or design pictures where the load balancing is not clear and probably you know and we'll talk about that later.

4:14

But at ING we have the physical client, we have client side load balancing.

4:19

Maybe you can do load balancing using an engine X or maybe we also have the old APIs can use an F5 load balancer.

4:29

So how is the load balancing done?

4:32

And if it's a TPA MARAC API, maybe the dynamic service discovery is not correct.

4:39

Well, I hope not, but anyway, my point is it's quite complex and especially if you're doing a transaction in an API chain that's quite complex, there is no issue with a get or a read that you can just reenter and just get the information.

5:01

But doing a transaction in an API chain, that's very complex, that's just some of the issues I mentioned here.

5:11

How is the load balancing done?

5:13

What are the network zones, the firewalls, etc.

5:18

So it's notoriously difficult to handle a transaction in a long API chain or we sometimes say in a distributed environment.

5:28

And also, do you know all your requesters?

5:32

Do you know your flow including the network zones, including the firewalls, the load balancers, etc?

5:40

Do you know your complete chain from top to bottom?

5:45

Do you know the timeout settings in the chain?

5:48

The failover mechanism is used, Can you observe the transaction in your chain?

5:55

And do you know in which business functionalities your API is deployed?

6:01

It's part of?

6:03

So those are very important questions.

6:08

Can you answer them all if you're in the war room?

6:11

I hope so.

6:13

It takes quite some more time to solve your issue in the war room if you don't know your requesters, if you don't know your flow.

6:23

So observability is maybe one of the most important things.

6:30

It's crucial that you're crucial that you can observe and trace the path of the data request across the applications and networks.

6:40

So a simple request to load a web page, Yeah, it can be right away multiple APIs, it might bounce between a dozen different applications.

6:50

At ING we have tracing A Tracing is part of the mock libraries.

6:56

Please use tracing.

6:58

Tracing is for observability and you can trace the message through a chain.

7:07

But maybe you already realize all the APIs part of your chain must then be configured for tracing.

7:17

If some of the APIs are not configured, you won't see it.

7:21

Please use tracing.

Backpres

0:05

There are quite some challenges with API chains, especially if you're doing transactions.

0:11

In this context I mean insert, update, delete and especially if your API chain is long, like 30 APIs.

0:21

What we have seen during major incidents is that the timeouts were not set correctly.

0:27

They were using global defaults.

0:30

The timeouts were set too long, so too many retries were executed.

0:36

Even systems went down with a retry storm and services are doing work that never can be used.

0:45

They're spending resources to which clients are not listening anymore.

0:49

Let's look again at this example.

0:51

I have 4 APIs.

0:53

At the end, the database is updated and the first API sends a request to the second API.

1:00

The second API needs 2 seconds to process the request and again sends a request to the next API who also needs 2 seconds and again forwards the request to the last API in the chain, also taking two seconds to process the request.

1:20

And this API doesn't update in the database, which is usually the heaviest for performance.

1:26

And let's say it takes 5 seconds and this example this is in total 11 seconds.

1:34

And at the edge at the top of the chain the timeout is reached or the deadline is reached, you could say.

1:42

But what we did not discuss yet is that all the APIs below the edge, below the top API, the

first API, are spending resources processing requests, creating replies that are never processed.

1:59

That's a waste of resources.

2:02

So what's the retry store?

2:05

So an API.

2:06

Your API might be part of multiple chains.

2:10

So in this example, multiple APIs, some part of multiple chains.

2:16

At the end of the chain, a database is updated.

2:20

But that's not really relevant for this example.

2:24

But let's say that this database has a problem.

2:29

So maybe a latency issue on the network or a performance issue in the database.

2:34

There is some problem at the end of the chain.

2:39

SO1API in your chain is having problems.

2:43

The timeout is reached of the consumer, it's spending more time, the response time is higher than the timeout of the consumer.

2:52

So your consumer starts retrying and sending the request again.

2:58

And maybe you're part of multiple chains, multiple APIs.

3:02

Start retrying your API.

3:06

You did not build in throttling.

3:08

What could happen, your service crashes and what we have seen happening during incidents is that your consumer APIs are also overloaded.

3:20

Their consumers are retrying and retrying.

3:24

Until those consumer APIs go down and this can go up the chain.

3:30

All the APIs in the chain are retrying and retrying until the provider API goes down is overloaded.

3:39

Well, worst case from one API having a problem, it results in seven APIs having a problem.

3:48

That's what we call the blast radius and we want to limit this blast radius.

3:54

You have to set your timeouts correctly.

3:56

You have to have a good retry policy, so setting correct timeouts and having a good retry policy is crucial in an API chain.

4:08

Did you set a static timeout which decreases from the edge?

4:14

Did you align your timeouts with your API chains you're part of?

4:19

Can your application differentiate between a connection timeout and a request timeout?

4:26

Now think about this, how often do retries actually succeed?

4:31

So never ask the same instance the same thing.

4:35

Always go to the second instance of your provider API on a different availability zone.

4:44

Only retry on a different connection and maybe use an exponential back off.

4:52

Don't send retries, retries at the same interval, but the interval must increase.

5:01

Maybe stop after sending a couple of retries and you might be thinking.

5:06

That's easy for me to say, set the timeouts correctly, have a correct retry policy, but do you know your chain and for which chain and decrease how much?

5:20

But in touch point marak, there are already mechanisms available to set your timeouts correctly and to set your retry policy correctly.

5:32

It has an exponential back offsetting.

5:36

And another way to prevent overloading is to implement throttling.

5:41

With throttling you control the consumption of the resources by your application.

5:46

This allows your system to continue to function, can need Slas even with an extreme load, but you have to have an estimation on how much load your service can actually handle.

5:59

You have to have done a stress test or a ceiling test, and production metrics can help here.

6:06

And you have to be careful that you're slowing down your consumers.

6:11

But some examples of throttling are that you reject a request from a consumer who's already accessed your API more than let's say 10 or 20 times per second, or maybe disable or degrade some functionality in your application.

6:30

That's normal, essential anyway.

6:33

Of course we recognize the importance of setting timeouts correctly in your chain, having a correct retry policies that therefore Touchpoint Marak created a couple of new features, they're both experimental at the moment, it's now July 2023 and at the moment these features are experimental that test it.

6:59

But be careful.

7:01

So one is deadlines.

7:03

When acting on deadline is enabled, the MARAC will compare the timestamp of the deadline to the current server time.

7:11

If the time is passed, it will reject the request to prevent wasting resources and then there is back pressure.

7:19

When back pressure is enabled, the MARAC will keep a count of the number of requests it can handle at the same time.

7:27

This limit is calculated on each request based on the ability of the server to serve without an error, and when this limit is reached, the request is rejected and again a special status code is returned.

1
00:00:11,499 --> 00:00:12,138
Hello!

2
00:00:12,968 --> 00:00:16,213
Welcome to the lecture on Touchpoint Mesh in a nutshell.

3
00:00:18,313 --> 00:00:21,839
We have a large and heterogenous landscape of services.

4
00:00:22,458 --> 00:00:29,008
To achieve high-availability, each service runs multiple instances, that are continuously being updated by

5
00:00:29,019 --> 00:00:30,701
their owning DevOps Teams.

6
00:00:31,521 --> 00:00:34,339
The only constants in this are change and failure.

7
00:00:35,128 --> 00:00:41,099
In this dynamic environment the applications still need to provide a robust and resilient service in

8
00:00:41,109 --> 00:00:46,626
case of failure and for that they need a robust and resilient connectivity with their dependencies.

9
00:00:48,556 --> 00:00:51,263
Every application has multiple instances.

10

00:00:51,853 --> 00:00:58,983
Each instance regularly announces its location and operational mode to API Endpoint Discovery, also known

11
00:00:59,033 --> 00:00:59,988
as AED.

12
00:01:01,108 --> 00:01:07,218
The operational mode indicates whether the instance is ready for live customer traffic, must only be used

13
00:01:07,228 --> 00:01:12,438
for testing traffic, or is completely in maintenance and will reject all requests.

14
00:01:13,538 --> 00:01:19,968
Each consumer of an endpoint regularly refreshes the list of available instances that serve that endpoint

15
00:01:20,038 --> 00:01:22,000
and respects the operational mode.

16
00:01:23,130 --> 00:01:29,160
The mesh divides the load fairly across all known "healthy" instances by load-balancing each request

17
00:01:29,170 --> 00:01:32,826
to one of the known instances, preferring the faster ones.

18
00:01:33,945 --> 00:01:40,795
Number 4 indicates where timeouts are applied (or, Timeouts are applied at each connection.): Each action

19
00:01:40,806 --> 00:01:45,625
an application takes comes with an implicit understanding of how long that action should take.

20
00:01:46,216 --> 00:01:49,625
By setting timeouts this understanding is made explicit.

21
00:01:50,725 --> 00:01:55,125
The most common timeouts are request timeouts and connection timeouts.

22
00:01:55,725 --> 00:02:02,026

The global request timeout is the time in which the client will do its outmost best to satisfy the request.

23

00:02:03,146 --> 00:02:09,735

When the client could not get a response or the response was not in time, the request is automatically retried

24

00:02:09,806 --> 00:02:13,925

when it is safe (executing the request twice has no different effect).

25

00:02:15,025 --> 00:02:16,688

Retries are not free.

26

00:02:17,288 --> 00:02:23,398

To prevent a disproportionate amount of retries, each client has a Retry Budget that limits the amount

27

00:02:23,408 --> 00:02:26,925

of retries in relation to the amount of original requests.

28

00:02:28,025 --> 00:02:34,375

In order to increase success with retries, the client automatically applies exponential backoff where it

29

00:02:34,385 --> 00:02:37,138

will wait increasingly longer between retries.

30

00:02:38,288 --> 00:02:44,148

Some instances will be completely unreachable or broken in such a way that the instance cannot be used to

31

00:02:44,158 --> 00:02:45,363

fulfil requests.

32

00:02:46,473 --> 00:02:52,276

The client will automatically detect this based on the observed errors and temporarily ignore the instance.

33

00:02:52,896 --> 00:02:58,526

When all instances are marked as unavailable, the client will pick one of them to optimistically send

34

00:02:58,536 --> 00:02:59,213

a request.

35
00:03:02,164 --> 00:03:06,426
Applications should reject traffic when they are not ready to serve requests.

36
00:03:07,056 --> 00:03:10,289
This happens especially during startup and shutdown.

37
00:03:11,388 --> 00:03:16,576
During startup code needs to JIT and connections still need to initialise.

38
00:03:17,726 --> 00:03:23,676
Only after this has happened and we know that our downstream connections are available we are done starting

39
00:03:23,766 --> 00:03:25,938
and will report as live in AED.

40
00:03:27,039 --> 00:03:31,851
When shutting down we want to inform the consumers to not use the instance anymore.

41
00:03:32,471 --> 00:03:35,739
We do this by deregistering the instance with AED.

42
00:03:36,849 --> 00:03:43,379
As it takes some time for instance information to propagate through AED to the consumers, in the meantime

43
00:03:43,389 --> 00:03:51,864
the application will respond with a 503 Service Unavailable HTTP status code and a not-acknowledged (NACK) header.

44
00:03:52,994 --> 00:03:59,404
This signals consumers that the request can be safely retried on other instances "and" count the request

45
00:03:59,414 --> 00:04:02,251
as failure so the instance will not be used anymore.

46
00:04:04,201 --> 00:04:09,770
Applications consist of several modules that need to be ready before the application can properly serve

47
00:04:09,781 --> 00:04:10,688

business traffic.

48

00:04:11,288 --> 00:04:16,001

To detect whether these modules are ready, the concept of ReadinessProbes is available.

49

00:04:17,101 --> 00:04:22,938

By default the service will respond as service unavailable and register as such in AED.

50

00:04:24,058 --> 00:04:30,251

Liveness probes are used to detect whether the application is somehow still running but not functioning properly.

51

00:04:31,361 --> 00:04:37,591

The results of both all readiness probes and all liveness probes are exposed as endpoints for the benefit of

52

00:04:37,600 --> 00:04:40,975

schedulers such as Kubernetes (ICHPS).

53

00:04:40,975 --> 00:04:48,775

Merak comes with a warmup feature to ensure the first business request will be handled as fast as the 1000th

54

00:04:48,775 --> 00:04:52,113

request by eagerly loading code and making connections.

55

00:04:53,243 --> 00:04:58,813

This mechanism is pluggable to also warmup connections to a database or other systems.

56

00:04:59,913 --> 00:05:06,743

Due to the nature of RPC (remote-procedure call) a single instance may receive an overwhelming number

57

00:05:06,753 --> 00:05:07,613

of requests.

58

00:05:08,423 --> 00:05:14,583

The instance will protect itself by continuously calculating the optimal throughput it can handle and shed any

59

00:05:14,593 --> 00:05:15,513

additional load.

60
00:05:16,343 --> 00:05:21,851
This prevents both the instance from overloading and
allows backends to come up more gradually again.

61
00:05:22,471 --> 00:05:24,000
This is called back pressure.

62
00:05:25,091 --> 00:05:31,380
In complex call chains and in services that serve
multiple different call chains it is hard, and sometimes

63
00:05:31,390 --> 00:05:37,540
even impossible, to define timeouts in such a way
that they both detect failures as allow for giving

64
00:05:37,550 --> 00:05:39,701
timely answers to the original client.

65
00:05:40,810 --> 00:05:47,560
Moreover, hiccups in the network can cause any request
to arrive at the server after the client already decided

66
00:05:47,571 --> 00:05:49,500
to stop waiting due to a timeout.

67
00:05:50,591 --> 00:05:55,826
A specific scenario where this can occur is when a
service makes two sequential calls.

68
00:05:56,435 --> 00:06:02,056
if the timeout of the consumer of that service is
shorter than the sum of the timeouts to the two calls,

69
00:06:02,056 --> 00:06:05,701
it may happen that there is not enough time left for
the second call.

70
00:06:06,500 --> 00:06:12,991
In order to prevent wasting effort creating unused
responses, each client calculates a timestamp until

71
00:06:13,001 --> 00:06:18,313
it is willing to listen for the answer and sends this
along with the request, see 12a.

72

00:06:19,413 --> 00:06:25,383
The server checks the deadline timestamp on each incoming request and rejects the request if it exceeded the

73
00:06:25,393 --> 00:06:27,201
deadline, see 12b

74
00:06:28,330 --> 00:06:31,638
This deadline is carried over the complete request chain.

75
00:06:32,258 --> 00:06:33,951
Indicated with 12c.

76
00:06:36,851 --> 00:06:42,641
All applications have an automatic way of checking their dependencies when starting up before announcing

77
00:06:42,651 --> 00:06:43,176
as ready.

78
00:06:43,766 --> 00:06:50,588
In some cases, however, it is desirable to make sure that a specific instance is used when handling requests.

79
00:06:51,708 --> 00:06:58,178
The mesh has a generic feature called "Explicit Routing" to specify a host to be used for a specific endpoint

80
00:06:58,248 --> 00:07:00,238
in the context of a single request.

81
00:07:01,358 --> 00:07:06,038
This configuration can be added on the request to the gateway (13)a

82
00:07:07,138 --> 00:07:11,001
and will then be propagated along the call chain (13)b

83
00:07:12,101 --> 00:07:15,013
and applied where applicable. (13)c

84
00:07:16,143 --> 00:07:22,573
This feature can be used to run test scripts or validation script while making sure it is actually the canary

85
00:07:22,583 --> 00:07:23,338
being tested.

86
00:07:24,458 --> 00:07:28,013
The mesh also implements Traffic Routing via AED.

87
00:07:28,633 --> 00:07:31,600
This applies on a flow rather than a simple request.

88
00:07:32,401 --> 00:07:38,781
Let's broaden the scenario by adding the concept of
Datacenters represented by Datacenter 1 and Datacenter

89
00:07:38,781 --> 00:07:39,101
2.

90
00:07:39,691 --> 00:07:44,038
By using the Touchpoint Platform Service, shortened
as TPS.

91
00:07:44,668 --> 00:07:52,058
This is done by creating a resource of type DatacenterOverride
for Datacenters or ServiceStatusOverride for services

92
00:07:52,058 --> 00:07:52,726
(14).

93
00:07:53,526 --> 00:07:58,288
We can instruct A to not use Datacenter 1, or also
a specific service,

94
00:08:00,168 --> 00:08:05,918
Even with all these mechanisms at our disposal, things
will fail and we need the ability to observe what

95
00:08:05,928 --> 00:08:10,317
is going on in our application to be able to detect
and be alerted by this failure.

96
00:08:11,317 --> 00:08:17,067
Merak provides out of the box extensive metrics on
incoming and outgoing calls, and metrics that expose

97
00:08:17,077 --> 00:08:19,548

the state of features such as readiness probes.

98

00:08:20,628 --> 00:08:26,398

Next to metrics we need details of the request in the call-chain to diagnose errors, this is called

99

00:08:26,408 --> 00:08:26,911

Tracing.

100

00:08:27,411 --> 00:08:31,857

Merak out of the box takes care of tracing all incoming calls and outgoing calls.

101

00:08:32,427 --> 00:08:38,037

It also provides a mechanism to automatically carry over the tracing context in the code, making sure

102

00:08:38,047 --> 00:08:40,751

every outgoing call is linked to the correct incoming call.

103

00:08:41,831 --> 00:08:46,918

Sometimes metrics and tracing alone are not enough to understand what is going on and we need logging.

104

00:08:47,418 --> 00:08:52,689

Merak out of the box comes with the ability to ship logs to a central place via Kafka topics.

105

00:08:53,249 --> 00:08:58,285

The Kafka cluster is automatically configured based on the environment detected in the certificate.

106

00:08:58,785 --> 00:09:04,355

Merak also ensures that all logging is correlated with tracing information, allowing to jump between

107

00:09:04,365 --> 00:09:08,306

logging and tracing and also to see all logging for a given trace.

108

00:09:11,206 --> 00:09:15,156

There are currently two options for building applications for Touchpoint.

109

00:09:15,756 --> 00:09:17,993

The first is to use Merak directly.

110
00:09:18,613 --> 00:09:25,203
The second option is to use the Touchpoint Sidecar
and have your choice of programming language and application

111
00:09:25,213 --> 00:09:25,868
framework.

112
00:09:26,678 --> 00:09:33,748
In the future, we are moving to a sidecar first paradigm
to support the LCM of the mesh, thereby ensuring that

113
00:09:33,758 --> 00:09:36,481
all connectivity stays secure and reliable.

114
00:09:38,401 --> 00:09:41,831
This concludes the "The Touchpoint Mesh in a Nutshell"
topic.

115
00:09:42,461 --> 00:09:46,118
I hope you enjoyed and I look forward to seeing you
in the next video!

1
00:00:11,499 --> 00:00:12,138
Hello!

2
00:00:12,938 --> 00:00:17,013
In this module, we'll go over Observability in the
Service Mesh.

3
00:00:17,833 --> 00:00:21,814
Observability is one of the key elements of applications
support,

4
00:00:22,423 --> 00:00:29,294
it is the mechanism to determine the state of applications
and the impact of any changes or instability on a

5
00:00:29,304 --> 00:00:30,401
given environment.

6
00:00:32,321 --> 00:00:39,213
At ING, we use Metrics, Tracing, Global Logging and
SEM-A as observability mechanisms.

7
00:00:40,013 --> 00:00:43,363
Let's see how to configure and use them on the Service Mesh.

8
00:00:44,193 --> 00:00:47,188
These are available both for Merak and the Sidecar.

9
00:00:49,109 --> 00:00:52,988
At ING, Prometheus is used as the metrics solution.

10
00:00:53,778 --> 00:00:59,789
It works by having a central process that can scrape application metrics using an endpoint implemented

11
00:00:59,799 --> 00:01:04,564
by the target application that contains the metric data in a specific format.

12
00:01:05,193 --> 00:01:11,603
This scraping is executed periodically and Prometheus allows users to build queries to analyse the scraped

13
00:01:11,614 --> 00:01:12,364
metrics.

14
00:01:13,184 --> 00:01:17,514
Usually a metric contains tags to add dimensions to a given metric.

15
00:01:18,323 --> 00:01:24,743
To set it up on a Merak application, the application must have the dependency 'merak-prometheus-spring-boot-starter'

16
00:01:24,743 --> 00:01:32,326
in the pom.xml and the endpoint '/admin/metrics' will be available and ready to be scraped by Prometheus.

17
00:01:33,126 --> 00:01:42,839
By default, the port for this endpoint will be '10080' but it can be customised by setting the property 'merak.administration.server.bind-port'.

18
00:01:45,749 --> 00:01:53,349
Here, a metric named api requests total can be seen with the tags service, applicationVersion, host, dc,

19
00:01:53,349 --> 00:01:55,939
method, pathTemplate and peerService.

20
00:01:56,748 --> 00:02:03,368
Also, before the metric entry, two lines are available
for HELP and TYPE to give context over the metric,

21
00:02:03,368 --> 00:02:05,564
finally the value of the metric itself.

22
00:02:06,374 --> 00:02:09,801
All collected metrics will be present on the metrics
endpoint.

23
00:02:10,421 --> 00:02:13,438
More metrics and more tags mean heavier payloads.

24
00:02:15,359 --> 00:02:21,368
The "Google SRE Book" defines that the four golden
signals of monitoring are latency, this is the time

25
00:02:21,378 --> 00:02:23,039
it takes to service a request.

26
00:02:23,858 --> 00:02:24,526
Traffic,

27
00:02:25,116 --> 00:02:28,601
it is a measure of how much demand is being placed
on the system.

28
00:02:29,421 --> 00:02:31,911
errors, the rate of requests that fail

29
00:02:32,616 --> 00:02:35,951
and saturation, a measure of how full the service
is.

30
00:02:36,751 --> 00:02:40,463
In this context we add a fifth one, Lifecycle management.

31
00:02:42,374 --> 00:02:47,489
On the category of Latency, there are many metrics
to use both for upstream and downstream.

32
00:02:48,119 --> 00:02:53,076
For the context of this course, we recommend to get familiarised with these metrics.

33
00:02:53,876 --> 00:02:55,676
Let's take a quick look over them.

34
00:02:56,376 --> 00:03:02,806
`api_grouped_response_time_seconds`, measures the duration of processing a request and groups them by the status

35
00:03:02,816 --> 00:03:03,501
code level.

36
00:03:04,201 --> 00:03:11,026
`api_response_latency_seconds`, measures the latency of processing a request, using the labels `pathTemplate`,

37
00:03:11,026 --> 00:03:18,001
`httpStatusCode` and `peerService` it can be used to segment into nice views per endpoint, result and consumer.

38
00:03:18,701 --> 00:03:25,171
`api_response_time_seconds`, measures the duration of processing a request, this metric adds a label per

39
00:03:25,171 --> 00:03:27,114
specific http code.

40
00:03:27,704 --> 00:03:29,876
It can be used as an Errors metric.

41
00:03:30,576 --> 00:03:37,866
`finagle_client_perHost_request_latency_ms`, measures the latency of outgoing calls separated by each available

42
00:03:37,876 --> 00:03:39,476
host for the given endpoint.

43
00:03:41,386 --> 00:03:45,288
Going on to the category of Traffic, lets focus on these metrics.

44
00:03:45,988 --> 00:03:51,801

api_incoming_requests, measures the total amount of requests received by the application.

45
00:03:52,500 --> 00:03:56,713
api_requests, measures the amount of valid requests received.

46
00:03:57,413 --> 00:04:01,676
finagle_client_requests, measures the amount of outgoing requests.

47
00:04:03,596 --> 00:04:07,675
Now for errors, this set can cover the errors from outgoing requests.

48
00:04:08,376 --> 00:04:14,416
finagle_client_failures, which represents the times that an specific exception has been thrown during

49
00:04:14,426 --> 00:04:15,513
an outgoing call.

50
00:04:16,213 --> 00:04:21,700
finagle_client_success, is the amount of outgoing requests that ended up as a success.

51
00:04:23,641 --> 00:04:29,760
Saturation is a problem that has the power to affect the app functionality to the point of a full crash,

52
00:04:29,760 --> 00:04:34,938
for that reason this section is a bit longer to cover multiple angles of this important topic.

53
00:04:35,538 --> 00:04:38,501
Let's see a sub segment of metrics to keep an eye on.

54
00:04:39,321 --> 00:04:44,511
The back pressure metrics give an view on how many request are being rejected due to the application

55
00:04:44,521 --> 00:04:47,138
detecting failures on outgoing requests.

56
00:04:48,458 --> 00:04:54,338

The `finagle_client_backups` metrics allow to measure the impact of sending a secondary outgoing request.

57

00:04:55,648 --> 00:05:01,798

The `finagle_client_failfast` metrics allow to detect outgoing requests that are routed to an unavailable

58

00:05:01,808 --> 00:05:02,501

instance.

59

00:05:03,830 --> 00:05:09,520

The `finagle_client_loadbalancer` give an overview on the behaviour of the Finagle load balancer on the

60

00:05:09,530 --> 00:05:10,876

available instances.

61

00:05:13,276 --> 00:05:18,588

With jvm metrics, the state of memory, garbage collection and threads can be measured.

62

00:05:19,918 --> 00:05:26,568

Finally, `merak_deadline` allows to detect the behavior of the deadline propagation and the rejection of requests

63

00:05:26,578 --> 00:05:27,876

that are no longer needed.

64

00:05:30,286 --> 00:05:35,015

On the topic of Lifecycle management, Merak provides two important metrics

65

00:05:35,688 --> 00:05:42,463

`merak_certificates_ttl` provides information on the expiration of the certificates used by the application.

66

00:05:43,263 --> 00:05:49,913

`And operational_mode` shows the current status of the application on regards of availability via Automatic

67

00:05:49,923 --> 00:05:51,138

Endpoint Discovery.

68

00:05:53,039 --> 00:05:58,788

There are also metrics related to the pods where the application runs, these are provided by infra via

69
00:05:58,798 --> 00:06:00,288
dashboards that they control.

70
00:06:00,908 --> 00:06:03,639
These dashboards are available per datacenter.

71
00:06:04,469 --> 00:06:10,578
The available metrics are CPU usage, CPU quota, Memory usage, and Memory quota

72
00:06:11,358 --> 00:06:14,564
These metrics are about ICHP v1.

73
00:06:15,164 --> 00:06:22,784
In ICHP v2 and v3, these metrics will change, as they will be included in the namespace metrics, making

74
00:06:22,794 --> 00:06:24,901
them available in the Prometheus operator.

75
00:06:26,801 --> 00:06:30,500
Now that you know what to look for, let's talk about the set up needed.

76
00:06:31,320 --> 00:06:37,101
Currently, there are two platforms for application deployments; ICHP and VMs.

77
00:06:37,920 --> 00:06:41,901
Using the Kings road you can deploy your application to any of these.

78
00:06:42,741 --> 00:06:49,751
For ICHP, the Kings road will handle the setup in the namespace to collect the metrics for all the instances

79
00:06:49,800 --> 00:06:52,088
into one central instance per datacenter.

80
00:06:52,718 --> 00:06:58,338
This central instance can be set up as the target for scraping on the RTK2 configuration.

81
00:06:59,188 --> 00:07:05,250

For VMs, each instance is added as a scraping target of the RTK2 configuration.

82

00:07:06,061 --> 00:07:14,441

The Reliability ToolKit 2, also known as RTK2, is the managed service provided by MDPL to have Prometheus

83

00:07:14,451 --> 00:07:17,038

and Grafana available for applications.

84

00:07:17,678 --> 00:07:25,101

Using a Touchpoint Automation Resource of type 'ReliabilityToolKit2', a configuration can be created for scraping.

85

00:07:25,741 --> 00:07:30,900

For more information, visit the RTK2 Onboarding instructions on The Forge.

86

00:07:32,801 --> 00:07:38,511

Now that we have covered metrics, lets take a minute to talk about the common factor for Tracing, Global

87

00:07:38,520 --> 00:07:40,113

Logging and SEM-A.

88

00:07:40,943 --> 00:07:46,751

These three mechanisms rely on Kafka to transmit the information relevant to each mechanism.

89

00:07:47,550 --> 00:07:52,913

Connectivity to kafka depends on two things, Firewalls and Access Control Lists.

90

00:07:53,513 --> 00:07:57,288

The team needs to handle these before being able to connect to a topic.

91

00:07:57,918 --> 00:08:01,238

Access Control Lists apply for Acceptance and Production.

92

00:08:03,139 --> 00:08:07,451

Tracing is the ING implementation of the Open Tracing standard.

93

00:08:08,031 --> 00:08:09,988

It is owned by MDPL.

94
00:08:10,778 --> 00:08:17,189
It works using data generated by apps that is sent via kafka, this data is correlated using Trace and

95
00:08:17,198 --> 00:08:22,164
Span IDs and then rendered showing all the interactions between applications.

96
00:08:22,864 --> 00:08:29,653
Merak, and the Sidecar, implement all that is needed for an application to use TracING, including the forwarding

97
00:08:29,664 --> 00:08:33,913
of the trace headers so the downstream applications can use TracING properly.

98
00:08:34,723 --> 00:08:39,226
At the moment, TracING is working on adopting the new OpenTelemetry standard.

99
00:08:41,136 --> 00:08:47,355
Another vital aspect of observability is logging, it allows for teams to have detailed information about

100
00:08:47,365 --> 00:08:50,013
specific events happening within an application.

101
00:08:50,643 --> 00:08:54,463
At ING, Elastic is used for logs visualisation.

102
00:08:55,163 --> 00:09:01,063
Merak implements this feature by adding the dependency 'merak-kafka-application-logging-spring-boot-starter',

103
00:09:01,063 --> 00:09:06,473
this will take the log events generated by the app and send it to the topic defined by the configuration

104
00:09:06,483 --> 00:09:09,051
setting 'merak.logging.kafka.topic'.

105
00:09:09,700 --> 00:09:13,450
For creating a topic, follow the Global monitoring onboarding.

106
00:09:14,270 --> 00:09:17,450
The detailed instructions are available on The Forge.

107
00:09:18,051 --> 00:09:23,751
Do mind that when following these instructions, Global Monitoring will create one topic for the team.

108
00:09:24,361 --> 00:09:30,100
If the preference for topics is per app, then the team must create the topic prior to doing the onboarding.

109
00:09:30,920 --> 00:09:37,251
The topic must be created by the team owning the application following the instructions defined on The Forge, the

110
00:09:37,261 --> 00:09:39,438
category of the topic must be 'Logging'.

111
00:09:40,288 --> 00:09:46,738
Once this topic is in place, a deployment is needed for the application to connect to it and start generating

112
00:09:46,748 --> 00:09:47,763
logging events.

113
00:09:48,563 --> 00:09:56,063
BTP has defined an internal naming scheme that can be used by non-BTP teams, the convention is available

114
00:09:56,113 --> 00:10:00,926
on the Confluence page "Touchpoint Tribe - Logging to Global Monitoring Solution".

115
00:10:02,726 --> 00:10:07,088
SEM-A stands for Security Event Monitoring for Applications.

116
00:10:07,678 --> 00:10:14,418
It is a service for continuous monitoring of applications regarding security, the most basic events being 'service

117
00:10:14,418 --> 00:10:16,528
started' and 'service stopped'.

118
00:10:17,278 --> 00:10:23,488
It works by sending Kafka events to a topic owned

by TechPL that complies with a Schema shared via the

119
00:10:23,498 --> 00:10:24,588
Schema Registry.

120
00:10:25,288 --> 00:10:31,178
Merak implements the events for 'service started'
and 'service stopped' with the possibility of extension

121
00:10:31,188 --> 00:10:33,138
by the team owning the application.

122
00:10:33,758 --> 00:10:41,128
To be able to use it, the dependency 'merak-sem-a-spring-boot-starter'
must be added to the application and the properties

123
00:10:41,128 --> 00:10:51,238
'merak.sema.kafka.enabled', 'merak.application.ing-identifier'
and 'merak.application.country-code' must be set.

124
00:10:51,858 --> 00:10:54,275
More details can be found on The Forge.

125
00:10:55,106 --> 00:11:01,338
Then, the SEM-A onboarding process must be started
as part of the application's risk journey.

126
00:11:01,938 --> 00:11:06,488
The process is defined on the SEM-A Onboarding Confluence
page.

127
00:11:07,288 --> 00:11:13,388
Do mind, that the application must be on the Access
Control List of the topic 'saac-schema-prod-json'

128
00:11:13,388 --> 00:11:14,901
for this process to work.

129
00:11:16,821 --> 00:11:22,910
Now you know the core observability mechanisms, put
them in practice and be confident on what happens

130
00:11:22,920 --> 00:11:25,050
on your application at any time

131

00:11:26,118 --> 00:11:29,626
This concludes the "Observability in the Service Mesh" topic.

132
00:11:30,256 --> 00:11:33,913
I hope you enjoyed and I look forward to seeing you in the next video!

Lb3-1

0:03
To clarify load balancing for our APIs a little bit more, I'll divide the APIs in front end APIs and back end APIs.

0:13
What do I mean with a front end API?

0:16
A front end API gets the request from outside the data center.

0:21
That can be Internet or intranet, but they're always behind a gateway and an external call from the Internet that is from the DMZ.

0:36
Your authenticating proxy API gateway is in the Demilitarized Zone.

0:42
That's the EAGN at ING.

0:46
I will talk in detail about those network zones in the network part of this training.

0:53
But front end APIs are always behind a gateway and many times this gateway is implemented using a engine X.

1:03
Remember mentioning an engine X be smart.

1:07
So let's take a closer look at these front end APIs.

1:12

But first let me be a little bit more specific about the gateways which are implemented with the engine X.

1:21

So we have a separate instance Gateway Engine X as an external gateway creating the TPA access token that is security framework of TPA.

1:37

And we also have the old security framework with using a channel context object for RIAF.

1:45

We even have a separate instance for the Belgium RIAF context object and we also have a separate instance which can do a translation from the TPA security to the security.

2:06

In the end, it's a modification of your HTTP header by the engine X, Well, actually the engine X, we have several APIs which are integrated in the gateway, which are integrated in the Engine X, who handle all this stuff, which changes the HTTP header, do authentication, generate a token, et cetera.

2:34

Again, maybe I simplify a little bit, but that's basically what's done.

2:39

So if you look at our Internet facing APIs, our external facing APIs that the failover between the two data centers high level is managed by BGP, the Border Gateway protocol at Google, the Border Gateway protocol.

2:58

Sometimes they say that's what the enabled the Internet because the border Gateway protocol enables us to choose the shortest path to choose which cable to use, to put it simply.

3:14

So I have a quick response.

3:16

If I go to Amazon, let's say in the United States about using BGP and we can manipulate the route, the traffic on those cables, BGP is used by Internet service providers between data centers and I can manipulate the route.

3:36

So if one data center goes down with BGP, we can manipulate the route and make sure traffic now goes to the other data center and within the data center.

3:49

If you look at our gateways, our authenticating proxy, of course they are high available, there's not one.

3:57

There's not one engine X or one authenticating gateway in each data center, there are multiple and they are made high available using a F5.

4:07

Remember an F5 can only be used by infra or by those gateways.

4:14

It's not for the squats or the APIs.

4:18

The F5 can only be used for infra.

4:21

So we have an F5 on top of the engine access on top of the authenticating proxies.

4:28

The API gateways and the API gateways can do load balancing over the two data centers.

4:36

Look closely at this picture.

4:38

So the gateways are pointing to all API instances in both data centers.

4:48

If you look at our intranet front ends, it works a little bit differently.

4:54

We have a failover using the GTM, the global traffic manager that's active passive.

5:02

It's a DNS like solution and if one data center goes down, the GTM will make sure traffic will be routed to the other data center.

5:12

But maybe you remember it depends on your DNS cache, time to live, etc.

5:18

So there's a failover using a GTM and again there are multiple gateways in each data center.

5:27

Of course it's high available and the gateways can load balance the traffic over all instances that the gateways will point to both data centers.

5:40

For back end APIs it's a little bit more complex.

5:44

We have multiple ways to do load balancing.

5:48

So we have client side load balancing for all the TPA enabled applications.

5:53

That way we use maybe the existing F5 which is at the moment contained, but in the past you could use it.

6:01

So maybe you still are behind the F5 or you're behind the gateway and the gateway will do the load balancing for you.

6:10

Or you have your own engine provisioned in the IPC combined maybe with the GTM.

6:19

But do keep in mind it's not allowed anymore to use the F5 and of course in other countries like Belgium you have your own load balancer, even a hardware load balancer.

6:33

But please check with your solution architect or BTA what is your best solution.

6:41

So for API to API calls, the target is for TPA enabled applications to use client side load balancing using the dynamic service discovery that the server names are retrieved dynamically and I stress the advantages already.

6:59

I think so.

7:01

We also have the F5.

7:03

You can still use the old APIs are still using the F5, and maybe you are using the engine X in the IPC, or you're using a gateway to do the translation from Mozak to Riaf, from the new security framework to the older security framework.

7:28

So some load balancers can do a lot more.

7:32

We talked about SSL offloading and sometimes the monolithic old applications need session affinity or even API.

7:43

We don't want that.

7:44

Session affinity means you keep data in memory of one of your instances, and if that's the instance crashes, you've lost your session data.

7:54

It's a single point of failure.

7:57

So if you are storing data in memory, you must be able to handle the data loss.

8:05

If not, store it.

8:06

May be in Cassandra for example, but we don't support session affinity on the load balancer or gateway.

8:15

If you really need it, consult your solution architect, BTA maybe on your own engine X in the IPC.

8:24

But this is really not preferred because keep in mind some important characteristics of a modern application or microservice.

8:33

First of all, design for failure, be able to handle the data loss of your session data.

8:40

Second, smart endpoints, done pipes.

8:44

I translate this, solve it in your application.

8:49

Do not depend on my infra solution or your load balancer doing stickiness.

1

00:00:10,299 --> 00:00:10,939

Hello!

2

00:00:11,739 --> 00:00:17,264

In this module, we explain how to request touchpoint endpoints on touchpoint domains.

3

00:00:18,094 --> 00:00:21,739

First we will discuss the pre-requisites needed before you can start.

4

00:00:22,339 --> 00:00:27,976

Next we will do a walkthrough on where and how you need to do your request with the APIGateway team.

5

00:00:28,806 --> 00:00:33,776

Followed by a brief summary on our way of working, so you know what the next steps will be.

6

00:00:34,586 --> 00:00:38,826

There are links to documentation provided in the resources section of the course.

7

00:00:42,126 --> 00:00:48,335

Your api needs to be registered in the API registry, and have a version with the endpoints you want to

8

00:00:48,346 --> 00:00:54,838

expose, also check the touchpoint API Management documentation link in the resources section of the course.

9

00:00:55,938 --> 00:01:00,048

You need to have a service that implements the endpoints

you want to expose

10
00:01:01,043 --> 00:01:08,233
Your service needs to have its proper TPA SSL certificates,
our gateways will not accept self-signed or expired

11
00:01:08,243 --> 00:01:09,133
certificates

12
00:01:10,226 --> 00:01:16,476
You can only request endpoints to be exposed that
you have the correct relations with in the API registry

13
00:01:17,558 --> 00:01:22,028
Make sure your endpoints are marked as exposable in
the API registry

14
00:01:23,098 --> 00:01:29,200
Exposing any application to the Internet means you
need a valid, specific RCEC for your Application.

15
00:01:29,820 --> 00:01:35,175
This also applies to externally exposed domains on
test and acceptance environments.

16
00:01:35,776 --> 00:01:41,113
You can do this using the fast-lane process for inbound
authenticating proxy on concert.

17
00:01:41,743 --> 00:01:46,125
Please refer to the resources section of the course
for the links to the documentation.

18
00:01:47,225 --> 00:01:53,735
Exposing any application on production requires the
correct ServiceNow UUID for your application, that

19
00:01:53,745 --> 00:02:00,375
is Operational, Production, and has a support group
attached, this is needed for risk/procedural reasons

20
00:02:00,425 --> 00:02:03,601
and is used for MCR incidents if outages are reported.

21
00:02:04,721 --> 00:02:10,610
We support only one domain per request which means

each kind of domain will require a set of different

22
00:02:10,620 --> 00:02:15,138
level of assurance and other specific settings that
requires separate tickets.

23
00:02:15,748 --> 00:02:21,325
Also note that each domain does have settings for
all the environments included in one request.

24
00:02:22,836 --> 00:02:28,735
To start with the configuration of endpoints, please
follow the link to the form which is also shared in

25
00:02:28,745 --> 00:02:30,651
the resources section of the course.

26
00:02:33,951 --> 00:02:40,241
You will be automatically authenticated against AD
using your known credentials, otherwise you are asked

27
00:02:40,251 --> 00:02:42,763
to login with your username and password.

28
00:02:43,393 --> 00:02:49,583
This makes it so we can query information, help you
track your tickets, and verify ownership and permissions

29
00:02:49,583 --> 00:02:50,188
where needed.

30
00:02:51,318 --> 00:02:57,908
Once logged in, you need to select the domain where
your api is hosted and this must be the the Api domain

31
00:02:57,978 --> 00:03:00,038
that will eventually be used in production.

32
00:03:00,668 --> 00:03:06,888
For all environments the production domain name will
be used as the identifier to start your request with,

33
00:03:09,199 --> 00:03:15,508
Then you can select the API from the drop down, all
API's that belong to the purpose code you are part

34
00:03:15,518 --> 00:03:17,639
of will be visible in the dropdown box.

35
00:03:19,939 --> 00:03:25,409
Now you can select the implementing service, the
gateway will only use one service for handling your

36
00:03:25,419 --> 00:03:31,364
requests, the instances your service is running on
will be discovered via AED at runtime.

37
00:03:33,674 --> 00:03:39,883
If a production change is requested, for example
adding, changing or removing endpoints in production

38
00:03:39,894 --> 00:03:44,051
domains, please enable the 'Production change' required
checkbox.

39
00:03:44,681 --> 00:03:50,676
This will enforce you to add a cmdb identifier which
we need in case of production incidents.

40
00:03:51,776 --> 00:03:57,601
You will need to add the RCEC if you are trying to
expose your endpoint on a internet facing domain.

41
00:03:59,901 --> 00:04:05,631
Next you see the overview of endpoints in this version,
keep in mind that when checking an endpoint for an

42
00:04:05,640 --> 00:04:11,340
environment it creates it, unchecking it as non-selected
means that you want to remove this endpoint from the

43
00:04:11,351 --> 00:04:12,213
configuration.

44
00:04:14,514 --> 00:04:20,664
Due to current limitations in touchpoint and AED,
if you want a dev environment, you will need additional

45
00:04:20,673 --> 00:04:26,603
information as may be mentioned via a popup, you can
go to the previous screen without losing your progress

46
00:04:26,734 --> 00:04:29,526
and add the static instances you want exposed on dev.

47
00:04:31,866 --> 00:04:37,556
Once the fields are filled in correctly and verified,
please click submit which will create an incident

48
00:04:37,565 --> 00:04:38,800
ticket in ServiceNow.

49
00:04:42,130 --> 00:04:47,063
Please note the link provided, and keep an eye on
the ServiceNow updates on your request.

50
00:04:47,673 --> 00:04:53,293
Also the screen provides important information for
you on the private networks that need to be allowed

51
00:04:53,303 --> 00:04:55,163
to access your private network.

52
00:04:57,464 --> 00:05:03,383
Your ticket will be on the queue with the API Gateway
team, one of the so called sergeants of the day will

53
00:05:03,394 --> 00:05:06,813
review your request and pick it up to add to our configuration.

54
00:05:07,433 --> 00:05:12,524
When there are questions they will added in a comment
and your ticket will be set to 'Awaiting customer'.

55
00:05:13,121 --> 00:05:18,789
When you answer any question you can set the ticket
back to active, and it will show up on our queue again.

56
00:05:19,899 --> 00:05:26,289
When the requested change is added to our git repository
a build will be started and directly deployed to the

57
00:05:26,299 --> 00:05:28,876
dev, test and acceptance environment.

58
00:05:30,006 --> 00:05:36,276
Production changes will be frozen for one day so you

have time to validate changes as described in our
59
00:05:36,286 --> 00:05:39,701
way of working, a link is added in the resources section.

60
00:05:40,331 --> 00:05:47,214
Production changes are only done on Monday and Thursday
between 6 and 7 am central European time.

61
00:05:47,824 --> 00:05:50,901
In case of major incidents we can make exceptions.

62
00:05:51,731 --> 00:05:57,461
Thank you for listening to this instructional video
on configuring your touchpoint endpoints on the API

63
00:05:57,471 --> 00:05:58,101
gateway.

64
00:05:59,931 --> 00:06:04,600
This concludes the "Expose touchpoint service endpoints
via the gateway" topic.

65
00:06:05,230 --> 00:06:08,888
I hope you enjoyed and I look forward to seeing you
in the next video!

Canary4

0:02
And why not do a Canary release?

0:05
A Canary release is when you deploy a new version of your application next to the existing version of
your application.

0:13
You send a small percentage of the load to this new version and test.

0:20
And if it's wrong, you can simply destroy the Canary and you don't have to do a rollback because your
existing application is still running.

0:30

And for applications running in the ING container hosting platform ichp, this is fully supported by the pipeline.

0:39

In the Kings Rd.

0:40

pipeline there is a feature for a Canary deployment.

0:45

One more instance per cluster will be deployed next to your current version and a certain percentage of the load is sent to this Canary.

0:57

In your pipeline you can approve or reject the Canary.

1:03

Then you have a time of 48 hours.

1:06

Keep in mind you have to create a separate dashboard for this Canary release to test to monitor.

1:13

So let's visualize this.

1:15

You have two clusters within ING, one per data center.

1:19

You've deployed all head and blue.

1:21

Also built in the Kings Rd.

1:24

pipeline by the way, always deploy to two different availability zones if you're business critical.

1:32

And with Kings Rd.

1:33

you can use this Canary feature and in one cluster you can add this Canary release.

1:43

So this new version, the pipeline is halted for 48 hours and you can decide to accept or reject and if it's wrong.

1:55

If you reject, it's destroyed and you don't have to do a rollback.

2:02

So if you use that feature in the Kings Rd.

2:04

pipeline, don't forget to create a separate dashboard for your Canary release.

2:10

A Canary deployment of Vms works a little bit differently in Cuban.

2:14

Need this ichp an extra runtime.

2:17

An extra pot is created for Vms.

2:20

If you want to create an extra runtime you have to create an extra virtual machine and that will involve quite some effort.

2:29

Also in the Kings Rd.

2:32

pipeline for ICHP there is a procedure for Canary deployment that the pipeline is halted and you can choose to accept or reject.

2:43

With Vms you have to manage this rollout or hold back yourself and you can do it yourself with your pipeline.

2:52

So for Vms you can also use a Canary deployment.

2:56

So let's say you manage a service with an availability rating of a three or a four.

3:02

You are deployed 4 * 2 in each data center, each on red and blue.

3:08

But remember, each availability zone must be able to handle the peak load.

3:15

So if you want to do a Canary deployment, you can replace 1 instance 1 runtime with your new version.

3:24

That's your Canary deployment.

3:27

If that Canary release is not correct, you can initiate A rollback with your pipeline.

3:33

The Canary release is not integrated in your pipeline.

3:36

Your pipeline is not halted like the kings row pipeline for ichp.

3:43

But if this new version is correct, if this Canary release is correct, you can initiate A rollout to all instances with your pipeline.

3:54

With the Canary deployment, 2 versions of your service are running in parallel.

4:00

That should not be a problem as each microservice must be backwards compatible.

4:07

However, if your service is using some data store or data service like day bus, SQL Server or Cassandra and you have data change, it's quite a bit more complex, but it's not impossible.

4:23

There are scenarios, techniques and tools that can support a rollback even with a change in your data store.

4:32

Think of a tool like Liquid Base.

4:35

So remember, even if you are using a data store and you have a data change, it's still possible to use a Canary release.

4:44

It's still possible to do a rollback.

4:48

Please check the documentation and the links provided in the notes of this slide.

1

00:00:11,499 --> 00:00:12,138
Hello!

2

00:00:12,938 --> 00:00:18,289
In this module, we will discuss "Reliability and Resilience features in the Kingsroad".

3

00:00:19,408 --> 00:00:25,976
The Kingsroad is the easiest, simplest and safest way to deploy a Touchpoint-based application to Production.

4

00:00:27,776 --> 00:00:36,426
Kingsroad offers several reliability and resilience features and settings for ICHP and IPC VMs deployments.

5

00:00:37,566 --> 00:00:42,588
For the ICHP based deployments, following types of features are available:

6

00:00:43,588 --> 00:00:50,248
, The deployment manifest settings are hardcoded as they are discussed with ICHP and promoted as best

7

00:00:50,258 --> 00:00:50,988
practices.

8

00:00:51,608 --> 00:00:54,813
These are fixed and cannot be overwritten by our users.

9

00:00:55,813 --> 00:01:01,943
, Additionally, some features are enabled by default and can be optionally disabled (although we strongly

10

00:01:01,953 --> 00:01:03,013

recommend against it).

11
00:01:03,613 --> 00:01:05,963
We also call them opt-out features.

12
00:01:06,583 --> 00:01:10,888
While others are disabled by default and can be optionally enabled as needed.

13
00:01:11,488 --> 00:01:13,363
We call these opt-in features.

14
00:01:15,193 --> 00:01:20,526
For the IPC VMs based deployments, following types of features are available:

15
00:01:21,526 --> 00:01:28,505
,Ã¢The features we offer for IPC VMs based deployments are disabled by default and can be optionally enabled

16
00:01:28,516 --> 00:01:29,126
as needed.

17
00:01:29,726 --> 00:01:31,600
We call these opt-in features.

18
00:01:32,721 --> 00:01:37,676
Continue this lecture to find out what are these features and how they create value for you.

19
00:01:41,286 --> 00:01:45,451
We are now explaining the deployment manifest settings for ICHP:

20
00:01:46,550 --> 00:01:50,175
Zero downtime with rolling updates deployment strategy.

21
00:01:51,275 --> 00:01:57,838
The settings of the rollingUpdate strategy are maxSurge of 25% and maxUnavailable of 0.

22
00:01:58,958 --> 00:02:01,188
Affinity rules for zone and host.

23
00:02:02,308 --> 00:02:09,368

With this, Kingsroad makes sure that pods are evenly spread across the red and blue zones of the datacenters

24

00:02:09,368 --> 00:02:14,728
and that preferably a pod is not scheduled onto a node that has already scheduled a pod from the same

25

00:02:14,738 --> 00:02:15,413
deployment.

26

00:02:16,533 --> 00:02:18,988
Image pull policy set to IfNotPresent.

27

00:02:20,118 --> 00:02:25,926
This means the application image will be pulled only if not already present on the ICHP node.

28

00:02:27,016 --> 00:02:30,438
Default number of replicas (or pods) set to 3.

29

00:02:31,038 --> 00:02:35,188
A good practice is to set the number of replicas equal or greater than 2.

30

00:02:35,768 --> 00:02:39,625
Be very careful when you set the number of replicas equal to 1

31

00:02:40,625 --> 00:02:47,155
(likely you are setting to 1 due to technical constraints), if your pod fails on one datacenter, you are running

32

00:02:47,166 --> 00:02:48,726
only on the other datacenter.

33

00:02:51,535 --> 00:02:57,835
All these settings are fixed, cannot be changed by the user except the number of replicas which is configurable

34

00:02:57,845 --> 00:03:00,163
by the user through the configuration file.

35

00:03:01,263 --> 00:03:07,053
If you want to learn more about these settings, please see the Reference links to documentation in the resource

36
00:03:07,063 --> 00:03:08,875
section of the training course.

37
00:03:11,786 --> 00:03:15,826
We are now explaining the opt-out features available
for ICHP:

38
00:03:16,946 --> 00:03:17,613
Probes.

39
00:03:18,613 --> 00:03:25,438
For Merak applications and Sidecar, Kingsroad deploys
default Merak readiness and liveness health checks.

40
00:03:26,068 --> 00:03:32,801
For ICHP V2 platform, the user also has the possibility
to deploy their own startup probe.

41
00:03:33,921 --> 00:03:36,888
Confidence checking and live operational modes.

42
00:03:37,888 --> 00:03:42,963
Merak has several operational modes: live, maintenance
and confidence checking.

43
00:03:43,593 --> 00:03:48,176
These are explained in details on the API SDK page
on the Forge.

44
00:03:48,676 --> 00:03:55,236
For production deployments, Kingsroad always deploys
an application first on confidence checking (to allow

45
00:03:55,246 --> 00:03:57,288
several checks) and then on live.

46
00:03:57,788 --> 00:04:04,268
For D/T/A deployments, the user can choose what operational
mode(s) they want their application to be deployed

47
00:04:04,278 --> 00:04:04,500
in.

48
00:04:05,600 --> 00:04:12,010
In the picture, you see a kingsroad deployment to

an Acceptance environment of the DCR datacenter, first

49

00:04:12,020 --> 00:04:14,488

on confidence checking mode, then on live.

50

00:04:17,288 --> 00:04:23,058

If you want to learn more about these opt-out features,
please see the Reference links to documentation in

51

00:04:23,068 --> 00:04:25,401

the resource section of the training course.

52

00:04:28,311 --> 00:04:32,251

We are now explaining the opt-in features available
for ICHP:

53

00:04:33,351 --> 00:04:35,113

Namespace autoscaling.

54

00:04:36,243 --> 00:04:40,263

This allows dynamically charge based on the resource
usage of pods.

55

00:04:40,903 --> 00:04:44,438

Available only for ICHP V2 platform.

56

00:04:45,058 --> 00:04:52,213

We make use of the QuotaAutoscaler CRD created by
ICHP team as you see in the first code snippet.

57

00:04:54,033 --> 00:05:00,113

The second code snippet shows an example definition
for autoscaling your namespace's resource quota.

58

00:05:01,943 --> 00:05:03,038

Pods scaling.

59

00:05:04,158 --> 00:05:09,788

This deploys more pods if the load increases based
on the HPA object configuration.

60

00:05:10,408 --> 00:05:15,976

We make use of the HorizontalPodAutoscaler CRD as
you see in the first code snippet.

61

00:05:17,786 --> 00:05:22,676
The second code snippet shows an example definition
for autoscaling your pods.

62
00:05:24,476 --> 00:05:25,838
Canary deployments.

63
00:05:26,938 --> 00:05:33,428
In general, a canary deployment is a deployment pattern
that allows you to roll out new code or features to

64
00:05:33,438 --> 00:05:35,900
a subset of users as an initial test.

65
00:05:36,491 --> 00:05:42,360
Historically, the canary technique was inspired by
the fact that canary birds were once used in coal

66
00:05:42,370 --> 00:05:46,438
mines to alert miners when toxic gases reached dangerous
levels.

67
00:05:47,088 --> 00:05:51,363
Somewhat gruesomely, the gases would kill the canary
before killing the miners.

68
00:05:51,973 --> 00:05:55,613
However, this provided a warning to get out of the
mine tunnels.

69
00:05:57,433 --> 00:06:03,288
The picture explains how canary deployments work in
Kingsroad and is comprised of following steps:

70
00:06:04,428 --> 00:06:05,288
Step 1.

71
00:06:05,888 --> 00:06:12,788
In the beginning, the current version (v1.3.4) receives
100% of user traffic.

72
00:06:13,928 --> 00:06:14,838
Step 2.

73
00:06:15,448 --> 00:06:24,498
A new deployment, the "canary" (v1.3.5) is performed

with 1 new pod which will get 1 divided by (n+1) traffic

74

00:06:24,498 --> 00:06:29,101

(where n = number of pods for current deployment across
your both data centres).

75

00:06:30,241 --> 00:06:31,163

Step 3.

76

00:06:31,783 --> 00:06:38,513

Errors are observed in the monitoring and logging
OR timeout (configurable, default is 48 hours) is

77

00:06:38,523 --> 00:06:39,050

reached.

78

00:06:39,670 --> 00:06:45,238

The canary pod is removed and production version
remains v1.3.4.

79

00:06:46,378 --> 00:06:47,313

Step 4.

80

00:06:47,933 --> 00:06:55,383

Errors are NOT observed in the monitoring and logging
and the canary version (v1.3.5) becomes production

81

00:06:55,393 --> 00:06:55,863

version.

82

00:06:56,963 --> 00:06:57,838

Good to know:

83

00:06:58,538 --> 00:07:04,408

,Ã¢In the current implementation, traffic load weight
cannot be throttled and traffic routing to a specified

84

00:07:04,418 --> 00:07:06,413

subset of users is not possible.

85

00:07:07,063 --> 00:07:13,233

So canary pod gets equal amount of load compared
to live pods and canary traffic is received from a

86

00:07:13,243 --> 00:07:14,851

random subset of users.

87

00:07:15,480 --> 00:07:21,131

To allow for manual weight and routing setting, the Touchpoint Service Mesh will be adjusted to route

88

00:07:21,200 --> 00:07:26,263

and throttle traffic to control the weight of load and specific routing to the canary release.

89

00:07:27,263 --> 00:07:32,813

,Ã¢Canary deployments are NOT a replacement of proper unit, integration and acceptance tests.

90

00:07:33,813 --> 00:07:37,950

,Ã¢Canary deployments can and should be performed one datacenter at a time.

91

00:07:38,950 --> 00:07:44,251

,Ã¢If you reject the canary deployment, the deployment immediately returns to its initial state.

92

00:07:44,851 --> 00:07:47,175

You can see this as an instant rollback.

93

00:07:48,275 --> 00:07:55,313

In the next Kingsroad major version 9.0.0, this canary deployments feature will be made opt-out.

94

00:07:59,934 --> 00:08:01,239

Auto resources.

95

00:08:02,369 --> 00:08:08,601

This feature enables the automatic fetching of peer-token, manifest and trust stores at each deployment.

96

00:08:09,211 --> 00:08:15,071

Explanations on how the peer-token, manifest and trust stores work you can find in other sections of

97

00:08:15,081 --> 00:08:15,876

this course.

98

00:08:16,976 --> 00:08:22,756

We add the peer-token and manifest to the /properties folder of the config-zips and the trust stores to

99
00:08:22,766 --> 00:08:23,951
the /binaries folder.

100
00:08:25,081 --> 00:08:31,089
This feature will be made opt-out in the next Kingsroad
major release 9.0.0.

101
00:08:32,789 --> 00:08:33,751
Autocert.

102
00:08:34,881 --> 00:08:43,039
This feature enables the automatic issuing of mTLS
certificates and it's available only on ICHP V2 platform.

103
00:08:43,539 --> 00:08:49,051
Lifecycle of the certificate is 90 days and the renewal
will be triggered in the 21st day.

104
00:08:50,181 --> 00:08:56,211
The first redeployment of your application after the
certificate has been renewed will also reload the

105
00:08:56,221 --> 00:08:58,051
certificate into the application.

106
00:08:58,681 --> 00:09:02,351
This means that the reload is fully in your control
as an user.

107
00:09:03,461 --> 00:09:08,251
The picture shows an usage example for Autocert on
ICHP V2.

108
00:09:09,381 --> 00:09:15,388
This feature will be made opt-out in the next Kingsroad
major release 9.0.0.

109
00:09:16,899 --> 00:09:18,188
Classic rollback

110
00:09:19,204 --> 00:09:22,614
If you need to do a rollback, you have this option
at hand.

111

00:09:23,214 --> 00:09:29,054
By using it, you are doing a new deployment with
a previous version of the application, only for the

112
00:09:29,064 --> 00:09:32,801
rollback the confidence checking operational mode
stage will be skipped.

113
00:09:33,411 --> 00:09:39,914
Moreover, the maxSurge (for the rollingUpdate deployment
strategy) will be set to 100%.

114
00:09:40,543 --> 00:09:45,774
This means that if you have the double amount of
resources available on the namespace, when the rollback

115
00:09:45,784 --> 00:09:50,351
is done, all the pods will be rolled out at the same
time, making it faster.

116
00:09:50,941 --> 00:09:56,063
If you have less resources, the surge will go up
to the respective maximum (eg.

117
00:09:56,663 --> 00:10:03,573
if you have 40% more resources available, the maxSurge
will become 40% basically), but it will NOT take up

118
00:10:03,583 --> 00:10:04,901
any used resources.

119
00:10:06,021 --> 00:10:10,538
Note that the same release policies apply as for a
regular production deployment.

120
00:10:11,138 --> 00:10:16,214
If these policies need to be bypassed, the break
the glass mechanism needs to be used.

121
00:10:16,834 --> 00:10:22,203
Break the glass is a mechanism that lets the users
continue with their Production deployment even if

122
00:10:22,214 --> 00:10:23,464
a policy is failing.

123

00:10:25,263 --> 00:10:31,189
The helm rollback is leveraging helm native mechanism
and is much faster than the classic rollback.

124
00:10:33,989 --> 00:10:39,619
If you want to learn more about these opt-in features,
please see the Reference links to documentation in

125
00:10:39,629 --> 00:10:41,976
the resource section of the training course.

126
00:10:43,886 --> 00:10:48,288
We are now explaining the opt-in features available
for IPC VMs:

127
00:10:48,288 --> 00:10:50,163
Autocert -

128
00:10:51,283 --> 00:10:58,875
This feature enables the automatic issuing of mTLS
certificates and it's available also on IPC VMs.

129
00:10:59,985 --> 00:11:04,688
The picture shows an usage example for Autocert on
IPC VMs.

130
00:11:06,388 --> 00:11:13,028
Cyberark Vault integration uses the task developed
by One Pipeline which creates connection between Azure

131
00:11:13,038 --> 00:11:16,338
DevOps OnePipeline and Cyberark Password Vault.

132
00:11:16,838 --> 00:11:24,113
Kingsroad made it easy to consume it, an user just
has to pass 2 parameters to the release-app-vms template.

133
00:11:25,933 --> 00:11:27,113
Auto resources -

134
00:11:28,243 --> 00:11:34,475
This feature enables the automatic fetching of peer-token,
manifest and trust stores at each deployment.

135
00:11:35,085 --> 00:11:40,945
Explanations on how the peer-token, manifest and

trust stores work you can find in other sections of

136

00:11:40,956 --> 00:11:41,751
this course.

137

00:11:42,851 --> 00:11:48,660

We add the peer-token and manifest to the /properties folder of the config-zips and the trust stores to

138

00:11:48,671 --> 00:11:52,063

the /binaries folder as you can see in the first picture.

139

00:11:53,883 --> 00:12:01,213

The second picture shows an usage example of auto resources for IPC VMs deployments in a kingsroad pipeline.

140

00:12:02,343 --> 00:12:08,350

This feature will be made opt-out in the next Kingsroad major release 9.0.0.

141

00:12:11,450 --> 00:12:17,081

If you want to learn more about these opt-in features, please see the Reference links to documentation in

142

00:12:17,090 --> 00:12:19,438

the resource section of the training course.

143

00:12:21,369 --> 00:12:26,139

This concludes the "Reliability and Resilience features in the Kingsroad" topic.

144

00:12:26,768 --> 00:12:30,426

I hope you enjoyed and I look forward to seeing you in the next video!

Apigw1 1

0:03

Let's have a closer look at the API Gateways. And remember, sometimes different terms are used for API gateways. Sometimes it's called the Engine X inbound proxy or the authenticating proxy. But anyway, the API Gateway is responsible for authentication. It will generate an access token for the TPA Mohawk applications.

0:27

Actually it's an orchestration of several APIs. It's done by the Authentication Orchestration API, the Token API and the Means API. For example for reoff applications deprecated remember or I hope even decommed. But for reoff applications they still exist, a context object is generated and that's done by the sessions API.

0:54

We have special gateways that can do the translation from access token to complex object and the other way around. We have an external API gateway and an internal API gateway. The external API gateway is used for traffic from the Internet. The internal API gateway is used for traffic coming from the intranet from an environment.

1:24

Which is governed by ING, which is controlled by ING. Let's look at the external API Gateway first. High availability between the data centers we discussed before is managed by the BGP protocol, the Border Gateway protocol and the engine X. The API Gateway itself is positioned in our DMZ Demilitarized Zone.

1:52

That ING is called the EAGN and the EAGN is protected by an external firewall and an internal firewall and it engineers. The API Gateway itself is highly available within one data center by using a F5 load balancer. An F5 load balancer is on top of the API Gateway and the traffic from the API Gateway to your API.

2:22

So passes through the internal EAGN firewall, then to your firewall of your private network and then to your machine to your API. But if you are in the war room, it's important that you know the roles and responsibilities. If you have a problem you think with the API gateway or the traffic to your API.

2:47

So first the EAGN firewalls and the five load balancers are managed by the infra network teams. The API gateway itself, the engine XS are managed by the engine X team. So also the EAGN internal firewall is managed by the infra network teams. But the firewall of your private network is managed by yourself that you can manage it using the self-service portal.

3:16

So for a working traffic flow from the Internet to your API using the API Gateway, you have several operational dependencies. First of all, the EAGN external firewall must be configured correctly. It must be opened. A VIP must exist on the LTM on the F5A valid certificate.

3:42

TLS certificate must be present on the F5 on the load balancer and then of course the engine. The API gateway must be configured correctly and then the internal EAG and firewall must be opened, must be configured correctly and then your private network. Your firewall on your private network must be open.

4:09

So for delivery for a new implementation Now keep in mind that the external and the internal EAG and firewall are not automated yet. And the infra network team, they do have to configure a VIP on the LTM, the F5 they have to open the EAGN firewalls, they have to install a TLS certificate on the F5, a certificate that you have requested.

4:37

The API Gateway must be configured correctly and you can request a configuration on gateway.ing.net and you can open the firewall of your private network using the self-service portal. But keep in mind as step one to four are not automated yet, it can take up to five weeks before it's implemented. You have to create an Azure board story for the network team.

5:06

And I didn't even mention that you need an AIR CAG, a Review Committee External connection. That's an approval document that you need that you need to have before the EAGN firewall can be opened. But not always all steps are needed if you're only going to add a new endpoint to an existing domain, only the internal EAGN firewall needs to be changed.

5:36

So I hope it's clear that you are responsible for the TLS certificate, meaning you have to request a new certificate before it expires. You have to create a story on the Azure board for the network team to reinstall this certificate. Okay, that was the external API gateway. Let's now have a look at the internal API gateway.

6:00

For the internal API gateway you need a correctly configured GTM Global Traffic manager. You need a valid certificate on the API gateway at the API gateway, the engine needs to be configured correctly and the firewall must be open from the API gateway to your private network and by the way the firewall from the user on the offices, the intranet.

6:29

To the internal API Gateway is open by default for port 443. The GTM Global Traffic Manager is managed by the infra network teams, the Engine X, the internal API Gateway is managed by the Engine X team and the private networks and the firewalls of the private networks are managed by the squads.

6:54

The delivery and the implementation of a new flow on the internal API Gateway is a little bit easier than on the external API Gateway. The configuration of the GTM is managed by the Engine X team. The Engine X team will request and manage the TLS certificate and of course the Engine X team will configure the API Gateway.

7:19

The squad only needs to request a change of the internal API gateway on gateway.ing.net, but also they have to manage their own firewall. You have to open the firewall of your private network using

the self-service portal for the network names and implicitly your firewall change. Always refer to Confluence or the Forge to get the latest and up to date network info.

7:48

The Engine X team created the page with all the network and firewall information for all the API gateways.

1

00:00:11,519 --> 00:00:16,988

Welcome to the Resilience and Performance Testing course part II, which enables you to be on top of

2

00:00:16,998 --> 00:00:19,126

the Reliability of your components.

3

00:00:19,916 --> 00:00:26,096

In this training I will guide you through the world of "Load" and "Performance" testing, as well as "resilience"

4

00:00:26,096 --> 00:00:26,626

testing.

5

00:00:28,526 --> 00:00:31,238

There are many different "performance" test types.

6

00:00:31,738 --> 00:00:36,176

Slide shows a summary of the most important ones that you need to understand.

7

00:00:36,995 --> 00:00:37,788

These are:

8

00:00:38,288 --> 00:00:39,550

Peakload test.

9

00:00:40,640 --> 00:00:45,850

In this test you test the load you expect during peak hours for a limited period of time.

10

00:00:46,951 --> 00:00:48,163

Endurance test.

11

00:00:49,293 --> 00:00:52,263

This is a test to determine the stability of your system.

12

00:00:52,893 --> 00:00:58,723
This is done by putting a high load over an extended period of time, to determine the stability of your

13
00:00:58,733 --> 00:00:59,238
system.

14
00:00:59,858 --> 00:01:05,978
Both seen from your customer perspective by looking at the latency, throughput and success rate, but also

15
00:01:05,988 --> 00:01:07,250
from a system perspective.

16
00:01:08,250 --> 00:01:09,588
Breakpoint test.

17
00:01:10,678 --> 00:01:15,288
It is great that you know that your system can handle the load for a long period of time.

18
00:01:15,878 --> 00:01:17,963
But what are the weak spots of your system?

19
00:01:18,563 --> 00:01:23,576
You want to know this to be able to run your system on production as reliable as possible.

20
00:01:24,576 --> 00:01:25,488
Combitest.

21
00:01:26,618 --> 00:01:32,268
This is a complex end to end chain performance test, where you want to see how your component runs in a

22
00:01:32,278 --> 00:01:34,300
more or less real life setup.

23
00:01:34,900 --> 00:01:39,990
Due to the complexity of our architecture, there is one thing for certain, it will be different than

24
00:01:40,000 --> 00:01:42,826
production, but it can still give you great feedback.

25

00:01:43,446 --> 00:01:49,696
You should think about joining this test in an automated
way during each workday, between five thirty and seven

26
00:01:49,705 --> 00:01:50,588
thirty AM.

27
00:01:51,728 --> 00:01:52,776
Spike test.

28
00:01:53,905 --> 00:01:59,396
This is a test that shows you what happens when there
is an unexpected rise in traffic for a short amount

29
00:01:59,405 --> 00:01:59,988
of time.

30
00:02:00,588 --> 00:02:06,600
Like after a major outage for example, or a peak
load when Ed Sheeran is selling tickets for his concert.

31
00:02:08,531 --> 00:02:14,720
This short overview gives you insight in the performance
risks you are subjected to, and the test type that

32
00:02:14,730 --> 00:02:16,213
you can validate this risk with.

33
00:02:17,343 --> 00:02:21,663
This will also help you to determine which test types
you want to execute.

34
00:02:22,783 --> 00:02:28,893
When you are, for example, worried that your application
latency falls outside the service level objective

35
00:02:28,943 --> 00:02:32,863
during expected peak load, you can validate this with
a load test.

36
00:02:33,963 --> 00:02:39,843
When you want to validate that you have a stable system
over a longer period of time, you can validate this

37
00:02:39,853 --> 00:02:41,151
with an endurance test.

38
00:02:42,270 --> 00:02:47,880
When you want to know that your system is resilient,
you can do several resilience tests to validate this

39
00:02:47,891 --> 00:02:51,676
to see if you implemented the resilience patterns
correctly and they work.

40
00:02:53,576 --> 00:02:59,515
The basic load test is the most simple form of load
testing where you evaluate if your system can handle

41
00:02:59,525 --> 00:03:03,738
the expected throughput within the desired latency
and success rate.

42
00:03:04,868 --> 00:03:11,018
I advice to automate this test and run it daily, even
if you did not change your code, so that you can gather

43
00:03:11,028 --> 00:03:14,150
a lot of feedback and better understanding about your
system.

44
00:03:15,251 --> 00:03:18,613
The most common way of doing this test is a constant
load.

45
00:03:19,223 --> 00:03:25,323
You can however also choose to have several different
load levels, which will give you insight on the scalability

46
00:03:25,333 --> 00:03:26,101
of your system.

47
00:03:26,720 --> 00:03:33,201
Also, provide insights on the effect of increasing
throughput on latency and success rate for example,

48
00:03:33,201 --> 00:03:34,876
but also available threads.

49
00:03:36,796 --> 00:03:41,488
The endurance test or soak test is all about proving
the stability of your system.

50
00:03:42,068 --> 00:03:46,538
Your system in production is running 24 hours, 7 days a week.

51
00:03:47,168 --> 00:03:53,468
This means that issues that won't show in a short load test, can show in the real world, like instability,

52
00:03:53,468 --> 00:03:56,113
due to a memory leak or thread that keep locked up.

53
00:03:56,713 --> 00:04:01,013
When you run a test longer, you will likely see these issues starting to appear.

54
00:04:01,633 --> 00:04:04,751
When I say starting, that is exactly what it means.

55
00:04:05,351 --> 00:04:11,361
There is no guarantee that it will actually break, but maybe you can see increasing memory usage or decreasing

56
00:04:11,371 --> 00:04:14,651
available threads, or you see a slope in the latency.

57
00:04:15,281 --> 00:04:18,675
These are all signs that there are issues that need to be addressed.

58
00:04:19,495 --> 00:04:25,085
Make sure your monitoring is capable of spotting these issues and your logging is sufficient to understand

59
00:04:25,095 --> 00:04:25,400
them.

60
00:04:26,220 --> 00:04:28,651
This is a test you don't have to do daily.

61
00:04:29,261 --> 00:04:35,426
Do it once in a while, preferably when you had a significant code change or architectural change.

62
00:04:37,356 --> 00:04:42,225

This is a test that gives you information about what will happen when you increase the throughput, what

63

00:04:42,235 --> 00:04:48,426

will break first and how does it affect the experience of the users, related to latency and success rate.

64

00:04:49,516 --> 00:04:54,965

It gives you valuable insights on the limits of your system and which bottlenecks need to be addressed

65

00:04:54,965 --> 00:04:55,700

over time.

66

00:04:56,830 --> 00:05:02,940

This is also a test that doesn't need to be done daily, but do it either in a regular time interval, like

67

00:05:02,950 --> 00:05:07,588

every quarter, or do it after significant changes in your code or architecture.

68

00:05:08,688 --> 00:05:14,698

Be aware that when doing this test without the isolation, you will put a lot of unexpected load on your service

69

00:05:14,708 --> 00:05:15,501

providers.

70

00:05:16,091 --> 00:05:22,031

In fact, it could make sense to do this test from time to time on a domain level, to see the impact

71

00:05:22,040 --> 00:05:23,313

it has on the services.

72

00:05:25,214 --> 00:05:31,204

A spike test gives you important information on how your system handles a sudden but short increase of

73

00:05:31,213 --> 00:05:31,639

load.

74

00:05:32,268 --> 00:05:35,176

This can happen after a major outage for example.

75
00:05:36,306 --> 00:05:41,751
The idea is to run your normal load test scenario
and plan a short burst in load.

76
00:05:42,351 --> 00:05:47,821
You want to analyze what this load does with the
latency and success rate, and you want to see that

77
00:05:47,831 --> 00:05:50,951
after the spike your system gets back to normal over
time.

78
00:05:52,871 --> 00:05:57,551
Now you have a basic understanding of the performance
test types and how to use them.

79
00:05:58,181 --> 00:06:01,151
I want to continue with the topic of resilience testing.

80
00:06:02,270 --> 00:06:07,238
Resilience means the capacity to withstand or to recover
quickly from difficulties.

81
00:06:08,368 --> 00:06:09,738
Why is this important?

82
00:06:10,838 --> 00:06:16,428
You already tested that your system is doing fine
under normal conditions, and can meet the latency

83
00:06:16,438 --> 00:06:20,651
requirements, under a certain throughput with an acceptable
success rate.

84
00:06:21,771 --> 00:06:28,931
The reason is simple, things will break, our microservices
architecture is complex, and most of the systems have

85
00:06:28,940 --> 00:06:30,501
an abundance of dependencies.

86
00:06:31,631 --> 00:06:36,613
This is also the reason we, as S R E, put focus on
resilience patterns.

87

00:06:37,233 --> 00:06:40,238
Make sure your system is as bullet proof as possible.

88
00:06:41,338 --> 00:06:47,448
There are many good examples, like when not being
able to get a response from system A, just do a request

89
00:06:47,458 --> 00:06:48,808
to the backup system B.

90
00:06:49,383 --> 00:06:55,343
When one instance of your dependency is not running,
make sure you will not query this instance for a while,

91
00:06:55,343 --> 00:07:00,301
and when it is back online, be easy on the throttle
and build your load operation gradually.

92
00:07:01,411 --> 00:07:06,976
All these resilience patterns make sure that the impact
of failure is less than without them in place.

93
00:07:08,075 --> 00:07:13,838
In order to do good resilience testing, you have to
understand your architectural overview and chain.

94
00:07:14,458 --> 00:07:19,998
What do you need, and how do you need it, which dependencies
you have and what are the resilience patterns that

95
00:07:20,008 --> 00:07:20,813
you implemented.

96
00:07:21,963 --> 00:07:24,476
Write down the scenario you want to test.

97
00:07:25,116 --> 00:07:32,366
For example, if service A goes down I expect a 100%
error rate, if service A goes up I expect the normal

98
00:07:32,376 --> 00:07:35,726
latency and success rate within 30 seconds.

99
00:07:36,336 --> 00:07:42,076
Also include how you are going to measure the success
or failure of this test scenario, what you expect

100
00:07:42,086 --> 00:07:46,363
to see in your monitoring and what you expect to see
in your logs and alerts.

101
00:07:47,493 --> 00:07:54,193
This will help you understand the impact of your dependencies,
and a forward approach of reducing the impact as much

102
00:07:54,203 --> 00:07:56,913
as possible using existing resilience patterns.

103
00:07:58,834 --> 00:08:02,626
Now, lets have a look at 2 specific important scenario's
together.

104
00:08:03,226 --> 00:08:07,226
The first one is a slow response of a specific service
that you use.

105
00:08:08,346 --> 00:08:10,588
Think about what this could do to your system.

106
00:08:11,219 --> 00:08:14,901
I would imagine response times would increase, but
by how much?

107
00:08:16,001 --> 00:08:18,951
Will they keep increasing, or is there a limit on
it?

108
00:08:20,071 --> 00:08:26,241
What happens with my available threads, and what happens
with my success rate, and what happens when this service

109
00:08:26,251 --> 00:08:27,239
recovers itself?

110
00:08:28,339 --> 00:08:32,064
Do I keep having high latency, or will this recover
as well?

111
00:08:33,184 --> 00:08:39,734
When doing this kind of resilience tests, it helps
using stubs for your dependencies, so you are flexible

112
00:08:39,824 --> 00:08:43,464
and can configure them yourself without having the
need of a third party.

113
00:08:45,364 --> 00:08:51,473
Now, lets have a look at the second specific important
scenario's together, testing what happens when a specific

114
00:08:51,483 --> 00:08:52,914
service is unavailable.

115
00:08:54,033 --> 00:08:56,276
Think about what this could do to your system.

116
00:08:56,906 --> 00:09:00,863
I would imagine when there is no alternative path,
you will see errors.

117
00:09:01,453 --> 00:09:05,251
But which error code are you seeing, and what do
you see in your logs?

118
00:09:06,361 --> 00:09:09,639
Will you receive an instant alert, or does it take
a while?

119
00:09:10,239 --> 00:09:14,063
Is the information in your monitoring sufficient
to understand the problem?

120
00:09:15,164 --> 00:09:21,313
When the service recovers, it is interesting to see
if that works under load and, if so, how long does

121
00:09:21,323 --> 00:09:23,076
it take to restore your service.

122
00:09:24,176 --> 00:09:26,713
There are many other scenario's that you can think
of.

123
00:09:27,333 --> 00:09:32,626
Maybe you have had a significant outage and you want
to test this to see it happen in real time.

124

00:09:33,226 --> 00:09:39,126
And maybe, after, you want to test the resilience fix you implemented, to mitigate this issue as well.

125
00:09:41,026 --> 00:09:44,388
It is always difficult to decide when to do which tests.

126
00:09:45,028 --> 00:09:50,578
Above is just a guideline that could help you make a high level test approach, so you know when to do

127
00:09:50,588 --> 00:09:51,363
which test.

128
00:09:52,463 --> 00:09:55,300
The diagram is based on risk-based testing.

129
00:09:55,800 --> 00:10:01,580
So, depending on the load that you expect, and the change you want to implement, you can determine which

130
00:10:01,591 --> 00:10:06,826
tests you need to do and pass, before going to production, preferably using canary releasing.

131
00:10:07,946 --> 00:10:13,236
I think it is good to use this as a reference, but use your knowledge of your system and your architecture

132
00:10:13,306 --> 00:10:15,938
as well, to come up with your test approach.

133
00:10:17,849 --> 00:10:24,208
Before we end this part of the Resilience and performance testing course, let's have a closer look at the analysis,

134
00:10:24,208 --> 00:10:25,863
reporting and evidence part.

135
00:10:26,963 --> 00:10:30,414
Lets first have a look at some "anti-patterns" you want to check.

136
00:10:31,213 --> 00:10:32,901
The first one is a slope.

137
00:10:33,531 --> 00:10:40,639
This is basically a gradually increasing value during constant load, for example, on latency or error rates.

138
00:10:41,259 --> 00:10:44,751
This indicates that your system is not stable over time.

139
00:10:45,851 --> 00:10:50,864
The second is a peak, which is a suddenly increasing value over a short amount of time.

140
00:10:51,494 --> 00:10:57,214
This could be a lot of things, and we advice to check your monitoring and logs to identify the issue.

141
00:10:58,314 --> 00:11:04,438
The last one is a dip, a suddenly decreasing value, like a lack of load over a short amount of time.

142
00:11:05,068 --> 00:11:07,626
This could indicate some issue with your throughput.

143
00:11:08,726 --> 00:11:14,001
When you have checked on the "anti-patterns", the next step is to compare the results with something.

144
00:11:14,631 --> 00:11:19,126
This is usually a previous result, but also your service level objectives.

145
00:11:19,916 --> 00:11:23,414
Did you see a significant change compared to previous runs?

146
00:11:24,044 --> 00:11:26,664
Are you still within the service level objectives?

147
00:11:27,284 --> 00:11:29,964
those are the 2 things that you need to look at here.

148
00:11:30,764 --> 00:11:33,101
The last part is about the evidencing.

149

00:11:33,691 --> 00:11:38,841
It is great that you did some load testing, but how
do you store the evidence and match it to the version

150
00:11:38,851 --> 00:11:39,801
of your component?

151
00:11:40,601 --> 00:11:42,414
It is important that this is done.

152
00:11:44,334 --> 00:11:50,088
Within ING, there are several different tools that
are available for you to do your Reliability Testing.

153
00:11:51,178 --> 00:11:53,476
Below are 3 tools that you can use.

154
00:11:54,626 --> 00:11:55,576
CTK.

155
00:11:56,696 --> 00:12:02,866
The Conformance Test kit is a CLI tool, that allows
developers to write their application and focus on

156
00:12:02,876 --> 00:12:03,789
the happy path.

157
00:12:04,598 --> 00:12:11,098
The CTK will take care of all the plumbing required
for the testing of functional requirements and non-functional

158
00:12:11,098 --> 00:12:11,964
requirements.

159
00:12:12,964 --> 00:12:19,089
MjolnirAPI is providing a REST API to run Gatling
tests in ICHP.

160
00:12:20,179 --> 00:12:26,509
K6 is an open-source load testing tool, that makes
performance testing easy and productive for engineering

161
00:12:26,519 --> 00:12:27,189
teams.

162

00:12:27,789 --> 00:12:30,851
It is free, developer-centric, and extensible.

163
00:12:31,471 --> 00:12:39,064
We provide you with the necessary docker image and
the steps required to make K6 work in INGIT environment.

164
00:12:39,664 --> 00:12:45,884
You can then adapt our example scripts to your product,
and contact us for any difficulties and questions

165
00:12:45,894 --> 00:12:46,926
that may arise.

166
00:12:48,856 --> 00:12:53,050
This concludes the part II of Resilience and Performance
testing course.

167
00:12:53,680 --> 00:12:57,338
I hope you enjoyed and I look forward to seeing you
in the next video!

1
00:00:11,519 --> 00:00:16,988
Welcome to the Resilience and Performance Testing
course part II, which enables you to be on top of

2
00:00:16,998 --> 00:00:19,126
the Reliability of your components.

3
00:00:19,916 --> 00:00:26,096
In this training I will guide you through the world
of "Load" and "Performance" testing, as well as "resilience"

4
00:00:26,096 --> 00:00:26,626
testing.

5
00:00:28,526 --> 00:00:31,238
There are many different "performance" test types.

6
00:00:31,738 --> 00:00:36,176
Slide shows a summary of the most important ones
that you need to understand.

7
00:00:36,995 --> 00:00:37,788
These are:

8
00:00:38,288 --> 00:00:39,550
Peakload test.

9
00:00:40,640 --> 00:00:45,850
In this test you test the load you expect during
peak hours for a limited period of time.

10
00:00:46,951 --> 00:00:48,163
Endurance test.

11
00:00:49,293 --> 00:00:52,263
This is a test to determine the stability of your
system.

12
00:00:52,893 --> 00:00:58,723
This is done by putting a high load over an extended
period of time, to determine the stability of your

13
00:00:58,733 --> 00:00:59,238
system.

14
00:00:59,858 --> 00:01:05,978
Both seen from your customer perspective by looking
at the latency, throughput and success rate, but also

15
00:01:05,988 --> 00:01:07,250
from a system perspective.

16
00:01:08,250 --> 00:01:09,588
Breakpoint test.

17
00:01:10,678 --> 00:01:15,288
It is great that you know that your system can handle
the load for a long period of time.

18
00:01:15,878 --> 00:01:17,963
But what are the weak spots of your system?

19
00:01:18,563 --> 00:01:23,576
You want to know this to be able to run your system
on production as reliable as possible.

20
00:01:24,576 --> 00:01:25,488
Combitest.

21
00:01:26,618 --> 00:01:32,268
This is a complex end to end chain performance test,
where you want to see how your component runs in a

22
00:01:32,278 --> 00:01:34,300
more or less real life setup.

23
00:01:34,900 --> 00:01:39,990
Due to the complexity of our architecture, there
is one thing for certain, it will be different than

24
00:01:40,000 --> 00:01:42,826
production, but it can still give you great feedback.

25
00:01:43,446 --> 00:01:49,696
You should think about joining this test in an automated
way during each workday, between five thirty and seven

26
00:01:49,705 --> 00:01:50,588
thirty AM.

27
00:01:51,728 --> 00:01:52,776
Spike test.

28
00:01:53,905 --> 00:01:59,396
This is a test that shows you what happens when there
is an unexpected rise in traffic for a short amount

29
00:01:59,405 --> 00:01:59,988
of time.

30
00:02:00,588 --> 00:02:06,600
Like after a major outage for example, or a peak
load when Ed Sheeran is selling tickets for his concert.

31
00:02:08,531 --> 00:02:14,720
This short overview gives you insight in the performance
risks you are subjected to, and the test type that

32
00:02:14,730 --> 00:02:16,213
you can validate this risk with.

33
00:02:17,343 --> 00:02:21,663
This will also help you to determine which test types

you want to execute.

34

00:02:22,783 --> 00:02:28,893

When you are, for example, worried that your application latency falls outside the service level objective

35

00:02:28,943 --> 00:02:32,863

during expected peak load, you can validate this with a load test.

36

00:02:33,963 --> 00:02:39,843

When you want to validate that you have a stable system over a longer period of time, you can validate this

37

00:02:39,853 --> 00:02:41,151

with an endurance test.

38

00:02:42,270 --> 00:02:47,880

When you want to know that your system is resilient, you can do several resilience tests to validate this

39

00:02:47,891 --> 00:02:51,676

to see if you implemented the resilience patterns correctly and they work.

40

00:02:53,576 --> 00:02:59,515

The basic load test is the most simple form of load testing where you evaluate if your system can handle

41

00:02:59,525 --> 00:03:03,738

the expected throughput within the desired latency and success rate.

42

00:03:04,868 --> 00:03:11,018

I advice to automate this test and run it daily, even if you did not change your code, so that you can gather

43

00:03:11,028 --> 00:03:14,150

a lot of feedback and better understanding about your system.

44

00:03:15,251 --> 00:03:18,613

The most common way of doing this test is a constant load.

45

00:03:19,223 --> 00:03:25,323

You can however also choose to have several different load levels, which will give you insight on the scalability

46
00:03:25,333 --> 00:03:26,101
of your system.

47
00:03:26,720 --> 00:03:33,201
Also, provide insights on the effect of increasing throughput on latency and success rate for example,

48
00:03:33,201 --> 00:03:34,876
but also available threads.

49
00:03:36,796 --> 00:03:41,488
The endurance test or soak test is all about proving the stability of your system.

50
00:03:42,068 --> 00:03:46,538
Your system in production is running 24 hours, 7 days a week.

51
00:03:47,168 --> 00:03:53,468
This means that issues that won't show in a short load test, can show in the real world, like instability,

52
00:03:53,468 --> 00:03:56,113
due to a memory leak or thread that keep locked up.

53
00:03:56,713 --> 00:04:01,013
When you run a test longer, you will likely see these issues starting to appear.

54
00:04:01,633 --> 00:04:04,751
When I say starting, that is exactly what it means.

55
00:04:05,351 --> 00:04:11,361
There is no guarantee that it will actually break, but maybe you can see increasing memory usage or decreasing

56
00:04:11,371 --> 00:04:14,651
available threads, or you see a slope in the latency.

57
00:04:15,281 --> 00:04:18,675
These are all signs that there are issues that need to be addressed.

58
00:04:19,495 --> 00:04:25,085
Make sure your monitoring is capable of spotting these issues and your logging is sufficient to understand

59
00:04:25,095 --> 00:04:25,400
them.

60
00:04:26,220 --> 00:04:28,651
This is a test you don't have to do daily.

61
00:04:29,261 --> 00:04:35,426
Do it once in a while, preferably when you had a significant code change or architectural change.

62
00:04:37,356 --> 00:04:42,225
This is a test that gives you information about what will happen when you increase the throughput, what

63
00:04:42,235 --> 00:04:48,426
will break first and how does it affect the experience of the users, related to latency and success rate.

64
00:04:49,516 --> 00:04:54,965
It gives you valuable insights on the limits of your system and which bottlenecks need to be addressed

65
00:04:54,965 --> 00:04:55,700
over time.

66
00:04:56,830 --> 00:05:02,940
This is also a test that doesn't need to be done daily, but do it either in a regular time interval, like

67
00:05:02,950 --> 00:05:07,588
every quarter, or do it after significant changes in your code or architecture.

68
00:05:08,688 --> 00:05:14,698
Be aware that when doing this test without the isolation, you will put a lot of unexpected load on your service

69
00:05:14,708 --> 00:05:15,501
providers.

70
00:05:16,091 --> 00:05:22,031

In fact, it could make sense to do this test from time to time on a domain level, to see the impact

71
00:05:22,040 --> 00:05:23,313
it has on the services.

72
00:05:25,214 --> 00:05:31,204
A spike test gives you important information on how your system handles a sudden but short increase of

73
00:05:31,213 --> 00:05:31,639
load.

74
00:05:32,268 --> 00:05:35,176
This can happen after a major outage for example.

75
00:05:36,306 --> 00:05:41,751
The idea is to run your normal load test scenario and plan a short burst in load.

76
00:05:42,351 --> 00:05:47,821
You want to analyze what this load does with the latency and success rate, and you want to see that

77
00:05:47,831 --> 00:05:50,951
after the spike your system gets back to normal over time.

78
00:05:52,871 --> 00:05:57,551
Now you have a basic understanding of the performance test types and how to use them.

79
00:05:58,181 --> 00:06:01,151
I want to continue with the topic of resilience testing.

80
00:06:02,270 --> 00:06:07,238
Resilience means the capacity to withstand or to recover quickly from difficulties.

81
00:06:08,368 --> 00:06:09,738
Why is this important?

82
00:06:10,838 --> 00:06:16,428
You already tested that your system is doing fine under normal conditions, and can meet the latency

83
00:06:16,438 --> 00:06:20,651
requirements, under a certain throughput with an acceptable success rate.

84
00:06:21,771 --> 00:06:28,931
The reason is simple, things will break, our microservices architecture is complex, and most of the systems have

85
00:06:28,940 --> 00:06:30,501
an abundance of dependencies.

86
00:06:31,631 --> 00:06:36,613
This is also the reason we, as S R E, put focus on resilience patterns.

87
00:06:37,233 --> 00:06:40,238
Make sure your system is as bullet proof as possible.

88
00:06:41,338 --> 00:06:47,448
There are many good examples, like when not being able to get a response from system A, just do a request

89
00:06:47,458 --> 00:06:48,808
to the backup system B.

90
00:06:49,383 --> 00:06:55,343
When one instance of your dependency is not running, make sure you will not query this instance for a while,

91
00:06:55,343 --> 00:07:00,301
and when it is back online, be easy on the throttle and build your load operation gradually.

92
00:07:01,411 --> 00:07:06,976
All these resilience patterns make sure that the impact of failure is less than without them in place.

93
00:07:08,075 --> 00:07:13,838
In order to do good resilience testing, you have to understand your architectural overview and chain.

94
00:07:14,458 --> 00:07:19,998
What do you need, and how do you need it, which dependencies you have and what are the resilience patterns that

95

00:07:20,008 --> 00:07:20,813
you implemented.

96
00:07:21,963 --> 00:07:24,476
Write down the scenario you want to test.

97
00:07:25,116 --> 00:07:32,366
For example, if service A goes down I expect a 100%
error rate, if service A goes up I expect the normal

98
00:07:32,376 --> 00:07:35,726
latency and success rate within 30 seconds.

99
00:07:36,336 --> 00:07:42,076
Also include how you are going to measure the success
or failure of this test scenario, what you expect

100
00:07:42,086 --> 00:07:46,363
to see in your monitoring and what you expect to see
in your logs and alerts.

101
00:07:47,493 --> 00:07:54,193
This will help you understand the impact of your dependencies,
and a forward approach of reducing the impact as much

102
00:07:54,203 --> 00:07:56,913
as possible using existing resilience patterns.

103
00:07:58,834 --> 00:08:02,626
Now, lets have a look at 2 specific important scenario's
together.

104
00:08:03,226 --> 00:08:07,226
The first one is a slow response of a specific service
that you use.

105
00:08:08,346 --> 00:08:10,588
Think about what this could do to your system.

106
00:08:11,219 --> 00:08:14,901
I would imagine response times would increase, but
by how much?

107
00:08:16,001 --> 00:08:18,951
Will they keep increasing, or is there a limit on
it?

108
00:08:20,071 --> 00:08:26,241
What happens with my available threads, and what happens
with my success rate, and what happens when this service

109
00:08:26,251 --> 00:08:27,239
recovers itself?

110
00:08:28,339 --> 00:08:32,064
Do I keep having high latency, or will this recover
as well?

111
00:08:33,184 --> 00:08:39,734
When doing this kind of resilience tests, it helps
using stubs for your dependencies, so you are flexible

112
00:08:39,824 --> 00:08:43,464
and can configure them yourself without having the
need of a third party.

113
00:08:45,364 --> 00:08:51,473
Now, lets have a look at the second specific important
scenario's together, testing what happens when a specific

114
00:08:51,483 --> 00:08:52,914
service is unavailable.

115
00:08:54,033 --> 00:08:56,276
Think about what this could do to your system.

116
00:08:56,906 --> 00:09:00,863
I would imagine when there is no alternative path,
you will see errors.

117
00:09:01,453 --> 00:09:05,251
But which error code are you seeing, and what do
you see in your logs?

118
00:09:06,361 --> 00:09:09,639
Will you receive an instant alert, or does it take
a while?

119
00:09:10,239 --> 00:09:14,063
Is the information in your monitoring sufficient
to understand the problem?

120
00:09:15,164 --> 00:09:21,313
When the service recovers, it is interesting to see if that works under load and, if so, how long does

121
00:09:21,323 --> 00:09:23,076
it take to restore your service.

122
00:09:24,176 --> 00:09:26,713
There are many other scenario's that you can think of.

123
00:09:27,333 --> 00:09:32,626
Maybe you have had a significant outage and you want to test this to see it happen in real time.

124
00:09:33,226 --> 00:09:39,126
And maybe, after, you want to test the resilience fix you implemented, to mitigate this issue as well.

125
00:09:41,026 --> 00:09:44,388
It is always difficult to decide when to do which tests.

126
00:09:45,028 --> 00:09:50,578
Above is just a guideline that could help you make a high level test approach, so you know when to do

127
00:09:50,588 --> 00:09:51,363
which test.

128
00:09:52,463 --> 00:09:55,300
The diagram is based on risk-based testing.

129
00:09:55,800 --> 00:10:01,580
So, depending on the load that you expect, and the change you want to implement, you can determine which

130
00:10:01,591 --> 00:10:06,826
tests you need to do and pass, before going to production, preferably using canary releasing.

131
00:10:07,946 --> 00:10:13,236
I think it is good to use this as a reference, but use your knowledge of your system and your architecture

132

00:10:13,306 --> 00:10:15,938
as well, to come up with your test approach.

133
00:10:17,849 --> 00:10:24,208
Before we end this part of the Resilience and performance testing course, let's have a closer look at the analysis,

134
00:10:24,208 --> 00:10:25,863
reporting and evidence part.

135
00:10:26,963 --> 00:10:30,414
Lets first have a look at some "anti-patterns" you want to check.

136
00:10:31,213 --> 00:10:32,901
The first one is a slope.

137
00:10:33,531 --> 00:10:40,639
This is basically a gradually increasing value during constant load, for example, on latency or error rates.

138
00:10:41,259 --> 00:10:44,751
This indicates that your system is not stable over time.

139
00:10:45,851 --> 00:10:50,864
The second is a peak, which is a suddenly increasing value over a short amount of time.

140
00:10:51,494 --> 00:10:57,214
This could be a lot of things, and we advice to check your monitoring and logs to identify the issue.

141
00:10:58,314 --> 00:11:04,438
The last one is a dip, a suddenly decreasing value, like a lack of load over a short amount of time.

142
00:11:05,068 --> 00:11:07,626
This could indicate some issue with your throughput.

143
00:11:08,726 --> 00:11:14,001
When you have checked on the "anti-patterns", the next step is to compare the results with something.

144
00:11:14,631 --> 00:11:19,126
This is usually a previous result, but also your

service level objectives.

145
00:11:19,916 --> 00:11:23,414
Did you see a significant change compared to previous runs?

146
00:11:24,044 --> 00:11:26,664
Are you still within the service level objectives?

147
00:11:27,284 --> 00:11:29,964
those are the 2 things that you need to look at here.

148
00:11:30,764 --> 00:11:33,101
The last part is about the evidencing.

149
00:11:33,691 --> 00:11:38,841
It is great that you did some load testing, but how do you store the evidence and match it to the version

150
00:11:38,851 --> 00:11:39,801
of your component?

151
00:11:40,601 --> 00:11:42,414
It is important that this is done.

152
00:11:44,334 --> 00:11:50,088
Within ING, there are several different tools that are available for you to do your Reliability Testing.

153
00:11:51,178 --> 00:11:53,476
Below are 3 tools that you can use.

154
00:11:54,626 --> 00:11:55,576
CTK.

155
00:11:56,696 --> 00:12:02,866
The Conformance Test kit is a CLI tool, that allows developers to write their application and focus on

156
00:12:02,876 --> 00:12:03,789
the happy path.

157
00:12:04,598 --> 00:12:11,098
The CTK will take care of all the plumbing required for the testing of functional requirements and non-functional

158
00:12:11,098 --> 00:12:11,964
requirements.

159
00:12:12,964 --> 00:12:19,089
MjolnirAPI is providing a REST API to run Gatling
tests in ICHP.

160
00:12:20,179 --> 00:12:26,509
K6 is an open-source load testing tool, that makes
performance testing easy and productive for engineering

161
00:12:26,519 --> 00:12:27,189
teams.

162
00:12:27,789 --> 00:12:30,851
It is free, developer-centric, and extensible.

163
00:12:31,471 --> 00:12:39,064
We provide you with the necessary docker image and
the steps required to make K6 work in INGIT environment.

164
00:12:39,664 --> 00:12:45,884
You can then adapt our example scripts to your product,
and contact us for any difficulties and questions

165
00:12:45,894 --> 00:12:46,926
that may arise.

166
00:12:48,856 --> 00:12:53,050
This concludes the part II of Resilience and Performance
testing course.

167
00:12:53,680 --> 00:12:57,338
I hope you enjoyed and I look forward to seeing you
in the next video!

Observ1a

0:03
Observability is very important for site reliability engineering, and successfully operating a service
entails a wide range of activities.

0:13

And maybe you know the Site Reliability Pyramid, also known as the Dickerson pyramid that provides a set of principles that an organization can use to define and improve reliability to promote engineering excellence.

0:29

It builds upon the principles of DevOps to bring an engineering led approach to IT operations.

0:35

SLA uses software to automate system operation, identify problems and implement resolutions.

0:43

And this concept of SRE is developed at Google.

0:46

And in SRE we use the terms SLI, SLO and SLA.

0:52

The SLI you probably heard about, that's an agreement between the service provider and the customers about the service deliverables.

1:01

And SLI is a measurement of the characteristics of a service.

1:06

The most common SLIs are the four golden signals we will discuss later.

1:11

And the service level objective specifies a target level for the reliability of your service.

1:18

Because SLIs are key to making data-driven decisions about reliability, they are at the core of SRE practices.

1:26

An example for an SLI could be the proportion of successful requests as measured from the load balancer metrics and SLO could then be 97% success.

1:39

Or another example, the proportion of sufficiently fast requests as measured from the load balancer metrics and as low could be 90% of the request should be below 400 milliseconds.

1:55

And for SLI implementation, it's important you find a measurable implementation of your service.

outcome that you think matters to the users and make sure the specification includes an event success criteria and where and how it is measured.

2:14

Use white box monitoring to collect SLI metrics from the various components of your application and we'll discuss white box monitoring later.

2:24

The SLI is best expressed as an equation between good events that fulfill success criteria and all valid events.

2:33

So an SLO sets a target level of reliability for your service customers.

2:39

Above this threshold, almost all users should be happy with your service.

2:44

Below this threshold, users are likely to start complaining, and that's what we call the error Bridget.

2:52

The error Bridget is the amount of errors that your service can accumulate over a period of time before your users start being unhappy.

3:02

You can think of it as a pain tolerance for your users, but apply to a certain dimension of your service availability, latency and so forth.

3:13

So the error Bridget gives the number of allowed bad events and the error rate is the ratio of bad events to the total events.

3:23

You need to turn the AS allows in actionable alerting rules, but the goal is to be notified when there is a significant event, an event where a large portion of the error Bridget is consumed.

3:37

Consider the following attributes when evaluating an alerting strategy.

3:42

The precision the proportion of events detected that were significant.

3:47

Recall the proportion of significant events that were detected.

3:52

Detection time, how long it takes to send notifications in various conditions, and reset time how long alerts fire after an issue is resolved.

4:04

Observability is the ability to measure a system's current state based on its logs metrics and traces.

4:12

It allows teams to solve problems proactively.

4:17

Observability tools collect and analyze data, user experience, infrastructure, and network telemetry that will resolve issues before they impact business key Performance indicators.

4:31

And the more observable system, the more quickly and accurately you can navigate from an identified performance problem to its root cause without additional testing or coding.

4:43

The architects and developers who create the software must design it to be observed.

4:50

Observability and monitoring are not the same.

4:53

Monitoring tells you when something is wrong.

4:57

Observability tells you what's happening, why it is happening, and how to fix it.

5:03

Observability provides context around the problem, while monitoring just gives information about the existence of a problem.

5:13

Monitoring helps engineers determine that a problem exists.

5:17

Observability goes further by allowing them to understand why it exists.

5:23

And after you've made the system observable and you've collected the data using a monitoring tool, you must perform analysis either manually or automatically.

5:34

The better your analysis capabilities are, the more valuable your investments in observability and monitoring the call.

5:42

And why this need for observability?

5:45

Because of the increased complexity that the move from monolithic applications to microservices and containers that we need more context, the shift left the shifting responsibility.

5:57

We have to build in observability early in the process.

6:02

And because of the shifting responsibility, the cognitive load of the teams has increased.

6:07

Cloud and virtualization have made troubleshooting also more complex and the agile way of work increased pressure for delivery.

6:16

Sometimes getting priority for operational tasks is difficult and we want an immutable infrastructure.

6:25

Everything must be recreated using a script, no manual intervention.

6:31

This makes troubleshooting also more difficult and the liability is more important than ever.

6:38

And we need to be always on.

6:40

And there's a need for instant gratification.

6:44

And we have an increase in changes in the application, in the infrastructure, and we focus now on the liability.

6:51

But from a security point of view, we have advanced threats, advanced persistent threats.

6:58

So we must be able to react quickly to those threats like ransomware, for example.

7:04

And because of these rapid changes in applications and infrastructure capacity, management is much more difficult.

7:12

You have to anticipate quickly in a change of demand of hardware, for example.

7:19

So we have to change from reactive to proactive.

7:24

At the most basic level, monitoring allows you to gain visibility into your system, which is a core requirement for judging service health and diagnosing your service.

7:35

When things go wrong, your monitoring system should address 2 questions, What's broken and why.

7:41

The what's broken indicates the symptom, the why indicates A cause.

7:47

In other words, monitoring is finding the symptoms and causes, and the objectives of monitoring are to alert on conditions that require attention to investigate and diagnose those issues, to display information about the system visually, to gain insight into trends in resource usage or service health for long term planning, and to compare the behavior of the system before and after the change and for observability.

8:20

We have the three pillars of observability metrics loss and traces.

8:27

To support a data-driven resolution, we need loss, a record of what's happening within your application, and we need metrics the numerical assessment of application performance and resource utilization.

8:43

And we need traces how operations move throughout the system from one node to another, from one API to another.

8:52

And as a read, we have the four golden signals.

8:55

These signals define what it means for the system to be healthy.

8:59

The four golden signals are the basic essential building blocks for any effective monotonic strategy.

9:06

And those signals are first of all latency at the time taken to serve a request traffic that the stress from demand on the system.

9:16

The more traffic, the more stress errors or error rate, the rate of requests that are failing and saturation overall capacity of the service.

9:29

The saturation is a high level overview of the utilization of the system.

9:35

Establish benchmarks for each metric, showing when the system is healthy.

9:39

Ensuring positive customer experience in a timely and good Observability is actionable, meaning when an engineer is alerted, he is able to take some action immediately, and the four golden signals serve as an excellent starting point for actionable monitoring.

9:58

Observability requires that data from multiple sources is connected, optimized, and enhanced.

10:05

For context, you have to instrument in a way that it creates an understanding of relationships and dependencies between your services.

10:15

In a distributed environment, you have to create a single pane of glass into the health of all these services.

10:22

Building observable systems requires being able to understand the failure domain proactively, and that's a tall order.

Observ1a

0:03

Observability is very important for site reliability engineering, and successfully operating a service entails a wide range of activities.

0:13

And maybe you know the Site Reliability Pyramid, also known as the Dickerson pyramid that provides a set of principles that an organization can use to define and improve reliability to promote engineering excellence.

0:29

It builds upon the principles of DevOps to bring an engineering led approach to IT operations.

0:35

SLA uses software to automate system operation, identify problems and implement resolutions.

0:43

And this concept of SRE is developed at Google.

0:46

And in SRE we use the terms SLI, SLO and SLA.

0:52

The SLI you probably heard about, that's an agreement between the service provider and the customers about the service deliverables.

1:01

And SLI is a measurement of the characteristics of a service.

1:06

The most common Slis are the four golden signals we will discuss later.

1:11

And the service level objective specifies a target level for the reliability of your service.

1:18

Because as allows are key to making data-driven decisions about reliability, they are at the core of Sri practices.

1:26

An example for an SLI could be at the proportion of successful requests as measured from the load balancer metrics and SLO could then be 97% success.

1:39

Or another example, the proportion of sufficiently fast requests as measured from the load balancer metrics and as low could be 90% of the request should be below 400 milliseconds.

1:55

And for SLI implementation, it's important you find a measurable implementation of your service outcome that you think matters to the users and make sure the specification includes an event success criteria and where and how it is measured.

2:14

Use white box monitoring to collect SLI metrics from the various components of your application and we'll discuss white box monitoring later.

2:24

The SLI is best expressed as an equation between good events that fulfill success criteria and all valid events.

2:33

So an SLO sets a target level of reliability for your service customers.

2:39

Above this threshold, almost all users should be happy with your service.

2:44

Below this threshold, users are likely to start complaining, and that's what we call the error Bridget.

2:52

The error Bridget is the amount of errors that your service can accumulate over a period of time before your users start being unhappy.

3:02

You can think of it as a pain tolerance for your users, but apply to a certain dimension of your service availability, latency and so forth.

3:13

So the error Bridget gives the number of allowed bad events and the error rate is the ratio of bad events to the total events.

3:23

You need to turn the AS allows in actionable alerting rules, but the goal is to be notified when there is a significant event, an event where a large portion of the error Bridget is consumed.

3:37

Consider the following attributes when evaluating an alerting strategy.

3:42

The precision the proportion of events detected that were significant.

3:47

Recall the proportion of significant events that were detected.

3:52

Detection time, how long it takes to send notifications in various conditions, and reset time how long alerts fire after an issue is resolved.

4:04

Observability is the ability to measure a system's current state based on its log metrics and traces.

4:12

It allows teams to solve problems proactively.

4:17

Observability tools collect and analyze data, user experience, infrastructure, and network telemetry that will resolve issues before they impact business key Performance indicators.

4:31

And the more observable system, the more quickly and accurately you can navigate from an identified performance problem to its root cause without additional testing or coding.

4:43

The architects and developers who create the software must design it to be observed.

4:50

Observability and monitoring are not the same.

4:53

Monitoring tells you when something is wrong.

4:57

Observability tells you what's happening, why it is happening, and how to fix it.

5:03

Observability provides context around the problem, while monitoring just gives information about the existence of a problem.

5:13

Monitoring helps engineers determine that a problem exists.

5:17

Observability goes further by allowing them to understand why it exists.

5:23

And after you've made the system observable and you've collected the data using a monitoring tool, you must perform analysis either manually or automatically.

5:34

The better your analysis capabilities are, the more valuable your investments in observability and monitoring the call.

5:42

And why this need for observability?

5:45

Because of the increased complexity that the move from monolithic applications to microservices and containers that we need more context, the shift left the shifting responsibility.

5:57

We have to build in observability early in the process.

6:02

And because of the shifting responsibility, the cognitive load of the teams has increased.

6:07

Cloud and virtualization have made troubleshooting also more complex and the agile way of work increased pressure for delivery.

6:16

Sometimes getting priority for operational tasks is difficult and we want an immutable infrastructure.

6:25

Everything must be recreated using a script, no manual intervention.

6:31

This makes troubleshooting also more difficult and the liability is more important than ever.

6:38

And we need to be always on.

6:40

And there's a need for instant gratification.

6:44

And we have an increase in changes in the application, in the infrastructure, and we focus now on the liability.

6:51

But from a security point of view, we have advanced threats, advanced persistent threats.

6:58

So we must be able to react quickly to those threats like ransomware, for example.

7:04

And because of these rapid changes in applications and infrastructure capacity, management is much more difficult.

7:12

You have to anticipate quickly in a change of demand of hardware, for example.

7:19

So we have to change from reactive to proactive.

7:24

At the most basic level, monitoring allows you to gain visibility into your system, which is a core requirement for judging service health and diagnosing your service.

7:35

When things go wrong, your monitoring system should address 2 questions, What's broken and why.

7:41

The what's broken indicates the symptom, the why indicates A cause.

7:47

In other words, monitoring is finding the symptoms and causes, and the objectives of monitoring are to alert on conditions that require attention to investigate and diagnose those issues, to display information about the system visually, to gain insight into trends in resource usage or service health for long term planning, and to compare the behavior of the system before and after the change and for observability.

8:20

We have the three pillars of observability metrics loss and traces.

8:27

To support a data-driven resolution, we need logs, a record of what's happening within your application, and we need metrics the numerical assessment of application performance and resource utilization.

8:43

And we need traces how operations move throughout the system from one node to another, from one API to another.

8:52

And as a read, we have the four golden signals.

8:55

These signals define what it means for the system to be healthy.

8:59

The four golden signals are the basic essential building blocks for any effective monotonic strategy.

9:06

And those signals are first of all latency at the time taken to serve a request traffic that the stress from demand on the system.

9:16

The more traffic, the more stress errors or error rate, the rate of requests that are failing and saturation overall capacity of the service.

9:29

The saturation is a high level overview of the utilization of the system.

9:35

Establish benchmarks for each metric, showing when the system is healthy.

9:39

Ensuring positive customer experience in a timely and good Observability is actionable, meaning when an engineer is alerted, he is able to take some action immediately, and the four golden signals serve as an excellent starting point for actionable monitoring.

9:58

Observability requires that data from multiple sources is connected, optimized, and enhanced.

10:05

For context, you have to instrument in a way that it creates an understanding of relationships and dependencies between your services.

10:15

In a distributed environment, you have to create a single pane of glass into the health of all these services.

10:22

Building observable systems requires being able to understand the failure domain proactively, and that's a tall order.

Observing 1d

0:03

So let's have a look at observability and monitoring tools at ING and we have many tools.

0:09

We give you an overview of the preferred tools and always check the latest on the Forge as these tools change rapidly.

0:18

Then of course monitoring is mandatory from a risk perspective.

0:23

There is an OSB 2 control for this, but it's not very smart.

0:28

So ING has redefined some minimum standards, minimum standards for access to application loss, inciting system metrics, configurable dashboarding and filterable alerting to engineers.

0:44

So first access to application loss.

0:47

Every application should on some level what it's doing and what is going wrong, but this is traditionally done in log file and sending your log files to the Elastic stack is a very good way to do this and we'll see.

1:01

We have several solutions using Elastic Second insight in system metrics.

1:08

The metrics are simply the measurements of a specific variable over time.

1:13

And remember the four golden signals of latency, traffic, errors, saturation.

1:21

The Prometheus can scrape these metrics if you have the appropriate collectors, then configurable dashboarding next to data collection you need to be able to access it, organize it.

1:34

A good monitoring solution should have some options to put your data in graphs and this capability is imperative.

1:42

All elastic stacks come with dashboarding capabilities.

1:46

That's Kibana as part of the ELK stack.

1:49

But for metrics the most obvious choice is Graphana.

1:54

This is an example of Kibana.

1:56

In Elastic you can see a selection of your log file entries and you can use several selection criteria.

2:04

And this is an example of Graphana where you can see your metrics, your CPU, your memory and then filterable alerting to engineers.

2:15

Your system needs to be able to tell something is wrong and to be able to reach the correct person.

2:21

Your alerting system must be able to filter less important issues or route them to less intrusive channels.

2:29

A combination of Prometheus and the ING alerting transporter is a very good way to achieve this.

2:36

And before we take a look at the tools at ING, another guideline, never store data with a

confidentiality rating higher than two in operational loss and never store customer or payments data in loss.

2:51

Audit loss are an exception, but those are not operational loss.

2:57

And maybe you've heard of MDPL, the monitoring data pipeline.

3:01

The monitoring data pipeline is a set of tools, applications and processes to help users get control of their application data.

3:11

MDPL delivers IT monitoring as a service at a global level in ING.

3:17

And let's have a quick look at the concept of the monitoring data pipeline.

3:21

It's comprised of deep functional areas, collection, processing and delivery.

3:27

It supports of course data ingestion of four different streams of monitoring data, logs, metrics, events and traces.

3:37

And this data is coming from different source domains, normal cloud, the IPC, the ING private cloud and the public cloud.

3:46

And it must be able to process the collected data and deliver the resulting information to the consumers.

3:53

And the goal is to support the decrease of MTTR mean time to repair.

4:02

And ultimately, by using machine learning and big data analysis, they can help to move from being reactive to being proactive.

4:13

And on the Forge, several monitoring journeys are available for logging and for observability.

4:21

Please check the links provided in the notes of this slide.

4:24

So let's have a closer look at lock file monitoring metrics and traces.

4:29

ELK is a well known standard to parse arbitrary lock files and send them to the analytics engine for processing and visualization.

4:38

And the goal of ELK is to facilitate investigation of your application for known patterns as well as detect anomalies.

4:47

And if you want to use ELK at ING we have a couple of different flavors.

4:51

Elkas ELK as a service a self-service offering to build your own elastic search cluster.

5:00

But for this solution you need quite some knowledge and understanding of ELK.

5:05

And then we have law for all.

5:07

That's the ELK offering that supports application loss as well as native loss from Linux servers and therefore allow you to perform queries that include system and application information.

5:19

And then we have another elastic solution which also provides alerting and application monitoring.

5:26

It's provided by the Global Monitoring Squad in Belgium.

5:30

And for metrics, we have RTK too go with metrics.

5:34

You can see things like the utilization of your IT infrastructure, how many users are connected, what's the number of concurrent connections.

5:45

RTK 2 is based upon open source tools like Prometheus and Graphana.

5:51

With Prometheus, you can bring together metrics for all your applications and infrastructure.

5:56

And with Graphana, you're able to visualize the health of these applications and infrastructure components.

6:03

Where RTK is able to store all metrics for at most 32 days, the long term metrics store will keep metrics much longer.

6:13

That enables you to detect trends for traces.

6:17

We use tracing at ING.

6:19

Not only does tracing show your chain or what your chain looks like, it also reports whether errors have occurred.

6:27

And tracing is an implementation of the open Tracing IO standard for track and trace.

6:33

And we have a couple of variations of tracing.

6:36

Trace view is an extension of tracing, it gives you a higher overview, you can see the relations between components on a higher level.

6:45

And now we're very interesting and I think important Tool 1 dashboard, a chain monitoring tool.

6:52

It offers a real time visualization of a user defined chain based on dynamic chain data.

7:01

It can be used for monitoring.

7:03

It can show the dependencies between your APIs.

7:06

It shows the chain for example the ideal flow or the global NLBE logging flow and the landing flow.

7:16

All components using tracing or creating trace events are shown in this tool.

7:21

So middleware and databases day bus are not shown, but they are not generating trace events but in an API chain.

7:31

The consumer and provider create trace events and you can define your chain using a query language and you see an example here.

7:41

But one day's word is unique because it can visualize chain data and show only the components that make up your chain.

7:49

It shows only the request relevant for your chain.

7:54

The whole chain is shown by default and that can be an issue for very large chains.

8:00

So you can use filters and create anchors so you can filter and specify what is visualized.

8:08

For example, if you want to see your consumers, your dependencies and if the consumers are shown, maybe you don't want to see the dependencies of that consumer.

8:21

And if you click on a component you get a detailed view that you can see your error, success request, your endpoints, your consumers.

8:31

Not only that, in the detailed view you can click on a button CMDB and it shows you for example the team supporting a component in the CMDB.

8:45

So 1 dashboard shows the current health in real time with second granularity.

8:51

That changes can be monitored in real time.

8:54

Mitigating actions during incidents.

8:56

Monitor regular changes deployments.

8:59

For example, during a major incident you are able to perform mitigating actions more quickly.

9:05

You can see the results very quickly.

9:09

You can quickly rollback the actions if they had a negative effect.

9:13

I hope I already convinced you, but why should you use 1 dashboard?

9:18

It offers zero on boarding.

9:21

It's very easy to find an accessible for everyone within ING.

9:26

Also for our service providers like infra.

9:30

So if there is a firewall or network switch or whatever change in the infra, an infra engineer can see the results, can see the effects in one dashboard.

9:43

It's easy to understand.

9:44

You don't need to create your own dashboard.

9:47

You can configure your own chain using a query.

9:51

It has several use cases to show the integration and dependencies between APIs for debugging issues, for major incident handling and it uses one data source.

10:04

It uses tracing as a data source.

10:07

You don't need to hook up your own data source and a new tool Single view will be available soon.

10:14

It has some similar functions as one dashboard.

10:18

Single view will ultimately allow you to tie several MDPL products into one single view.

10:25

And on the subject of discovering relations and dependencies, there's also a promising tool called igraph.

10:33

Igraph tries to combine and visualize several important IT data sources.

10:38

It enables you to quickly visualize dependencies, find responsible teams and persons, and observe an entire application landscape.

Observing2c

0:03

IAT is the IMG Alert Transporter.

0:06

It automates the alerting process and speeds up alert notifications.

0:11

It ensures the right people are notified using flexible notifications, standby schedules, escalations and it supports multiple outgoing channels like phone calls, emails, SMS or method.

0:26

Most and IIT is integrated with Law for All and Prometheus and here you see an overview of Law for All including alerting with IAT so you can produce data, send data to Elastic using a CAF Carpenter or foul beat, a loft that is used to combine join a filter.

0:48

Analytics Search is your monitoring engine.

0:51

Analytics engine where Kibana can visualize your data and alerts are sent out using IAT.

1:01

For black box monitoring we use two external tools at ING, Splunk Synthetics, formerly known as Rigor and WTSS.

1:12

WTSS enables synthetic monitoring of the digital channel through robots.

1:17

These robots simulate multiple customer journeys on a regular basis, so WTSS offers black box monitoring of your applications.

1:27

It provides a report of journeys availability and reliability via E Reporter and you can access the WTSS raw data via Kibana.

1:38

And there is a relatively new tool, Splunk Synthetic, which you can use for black box monitoring.

1:44

So let's compare WTSS and Splunk Synthetic for configuration For Splunk you need to create your own scripts for WTSS that you have to request configuration at a central team.

1:59

They both support black box monitoring of your Internet web apps.

2:04

But support for black box monitoring of your mobile apps is only supported by WTSS support for your internal intranet web apps.

2:15

Then you have to use Splunk and not WTSS and you can use Splunk for testing of your internal APIs.

2:23

Now both are integrated with IIT and both are integrated with E Reporter that we will discuss soon.

2:31

And here you see some screenshots from Splunk Synthetic.

2:37

E Reporter is an internally developed tool which reports availability of your applications and business services.

2:44

An E reporter shows, for example, the uptime, latency, success rate and it can retrieve this data from various monitoring sources.

2:55

And also worth mentioning is SLIM.

2:58

SLIM is a tool to report your service level objectives and indicators.

3:03

While most monitoring tools in the ING have a data persistency of two months, Slim offers the ability to persist data for much longer.

3:12

So we can look back on your SLO history for a longer period.

3:17

On the related subject of incident management, we have a tool called Everbridge.

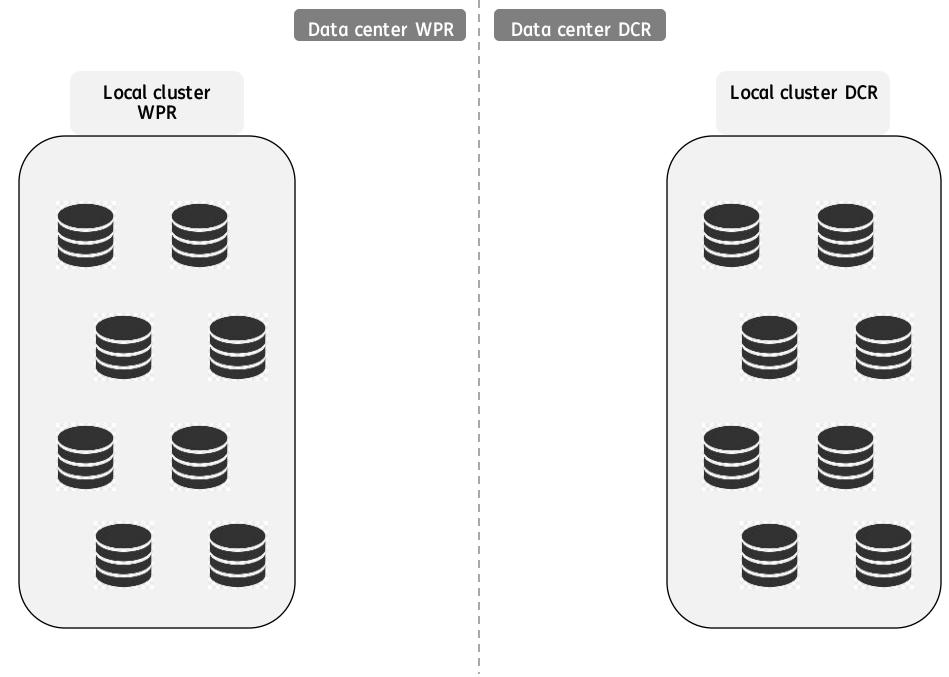
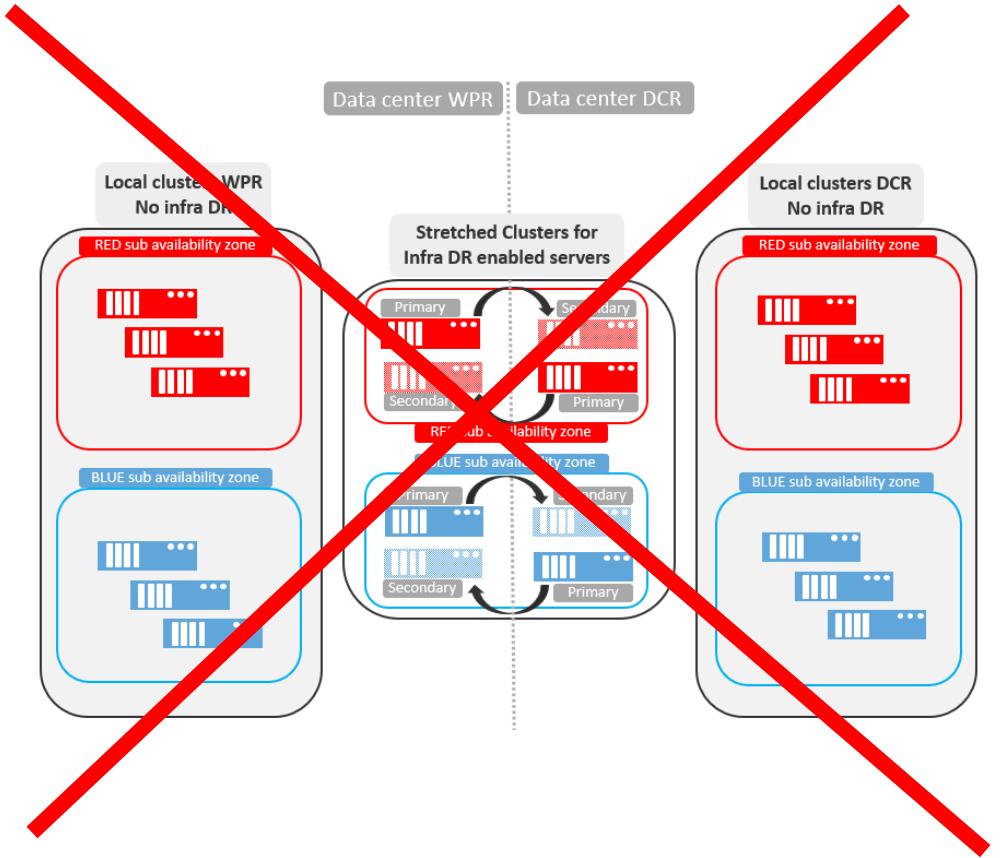
3:23

Everbridge helps ING to effectively communicate and respond to critical events.

3:30

It enables DevOps teams and departments to send out notifications in a standard and compliant way.

Correct use of sub-availability zones



Configuration of your database

Capacity management

Maximum shape limit				
Service level/Cloud type	IPC	POD: Romania (RO)	POD: Poland (PL)	Big Data
Thin	32	24	24	-
Normal	64	24	16	128

[Catalog](#) [Deployments](#) [Inbox](#)Deployments 1 item Search for deployments by name, description, IP address, resource name or machine status

Sort: Created Date (descending)



Created 22 minute... Never expires

ACTIONS ▾

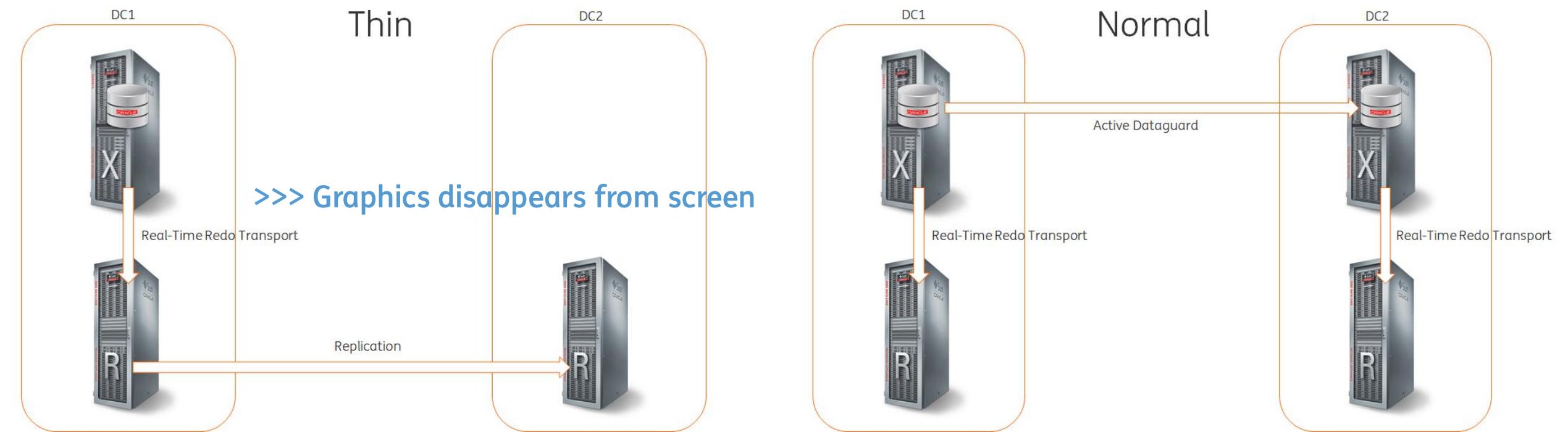
- Add additional access from P...
- Change Owner Oracle Databa...
- Copy Oracle Database
- Create DBFS on Oracle Databa...
- Delete DBFS on Oracle Databa...
- Patch Oracle Database
- Remove Oracle Database
- Resize Oracle Database**
- Restore Oracle Database
- Start Oracle Database
- Stop Oracle Database
- Unlock pdbadmin for Oracle D...
- View Details



Performance monitoring

Metric	Comparison Operator	Warning Threshold	Critical Threshold	Corrective Actions	Collection Schedule	Prevent Template Override	Edit
▽ TH307WCM_D00C390_341							
▷ Database Job Status							
▷ Failed Logins							
▷ Replication Processes Status							
▷ Response							
▽ Tablespaces Full							
Tablespace Space Used (%)	>=	85	97	None		<input checked="" type="checkbox"/>	

Back-ups and restores



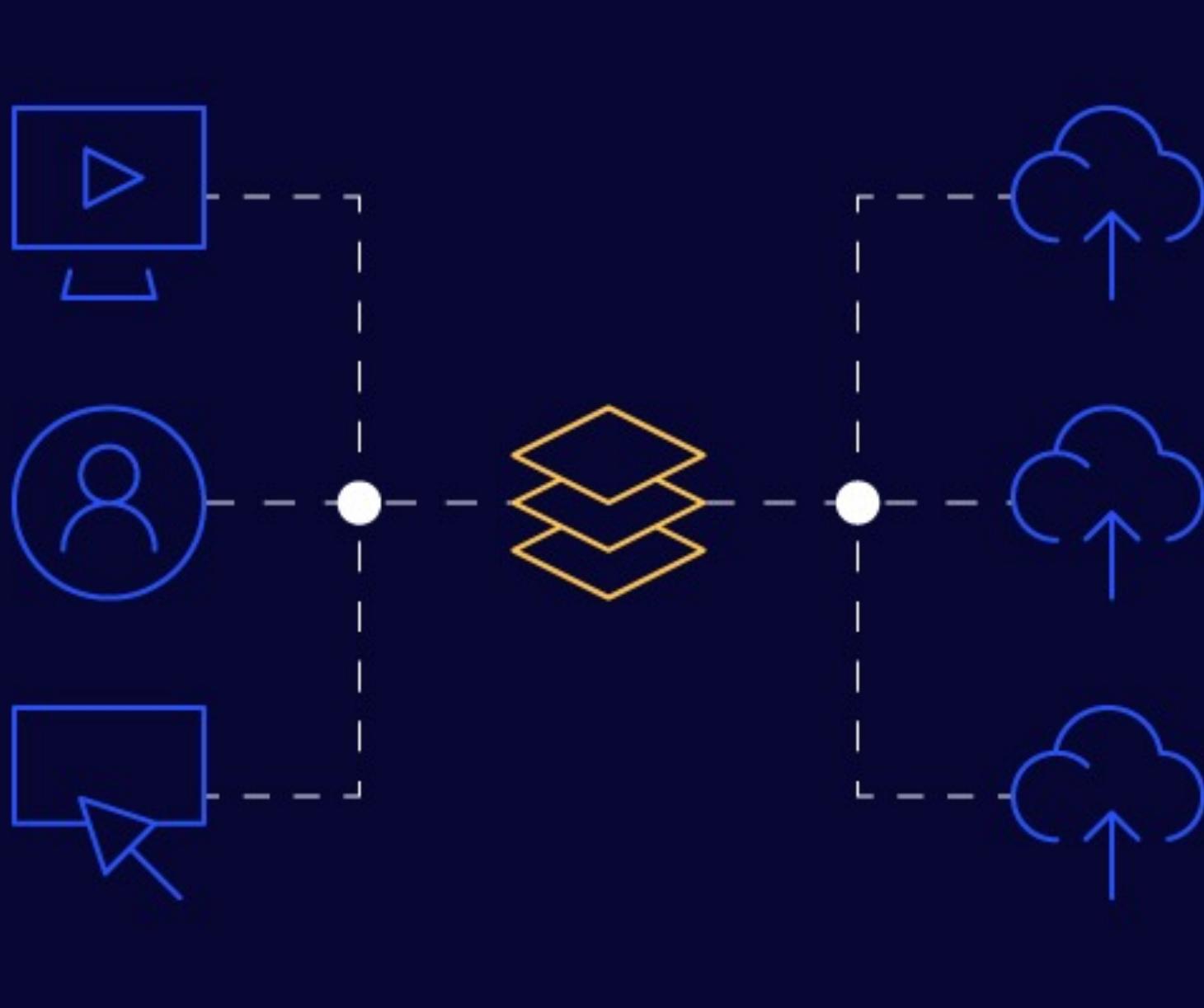
Patches and change strategy using IPC tools

Sort: Created Date (descending) 

ACTIONS 

- Add additional access from P...
- Change Owner Oracle Database...
- Change Patch Ring Oracle Dat...**
- Copy Oracle Database
- Create DBFS on Oracle Database...
- Delete DBFS on Oracle Database...
- Patch Oracle Database
- Remove Oracle Database
- Resize Oracle Database
- Restore Oracle Database
- Start Oracle Database
- Stop Oracle Database
- Unlock pdbadmin for Oracle Database...
- View Details

Disaster recovery scenario's



API Gateways

API gateways / Authenticating proxies / NGINX inbound proxy

Authentication is executed in the API gateway.

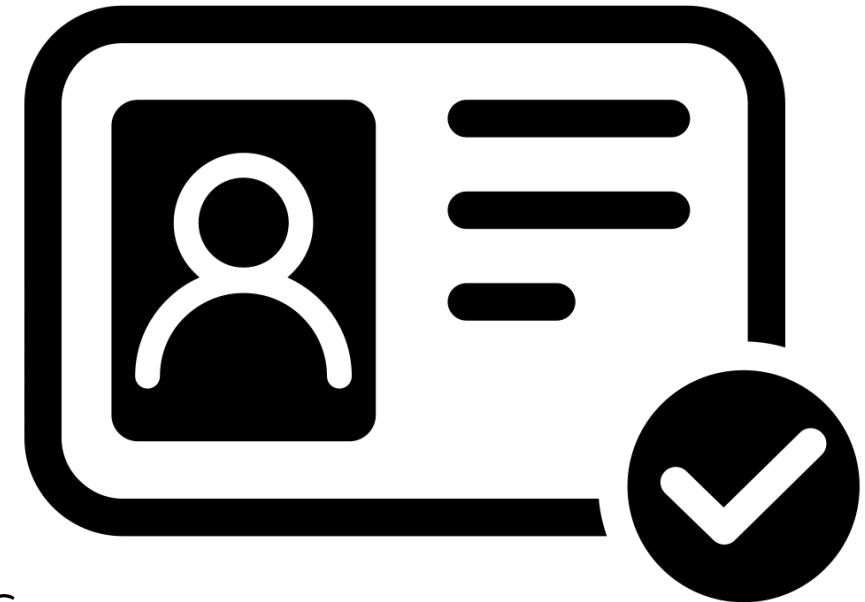
For TPA authentication (generating the access token) is an orchestration of several APIs:

- AuthenticationOrchestrationAPI
- TokenAPI
- Means API's (per means of authentication)

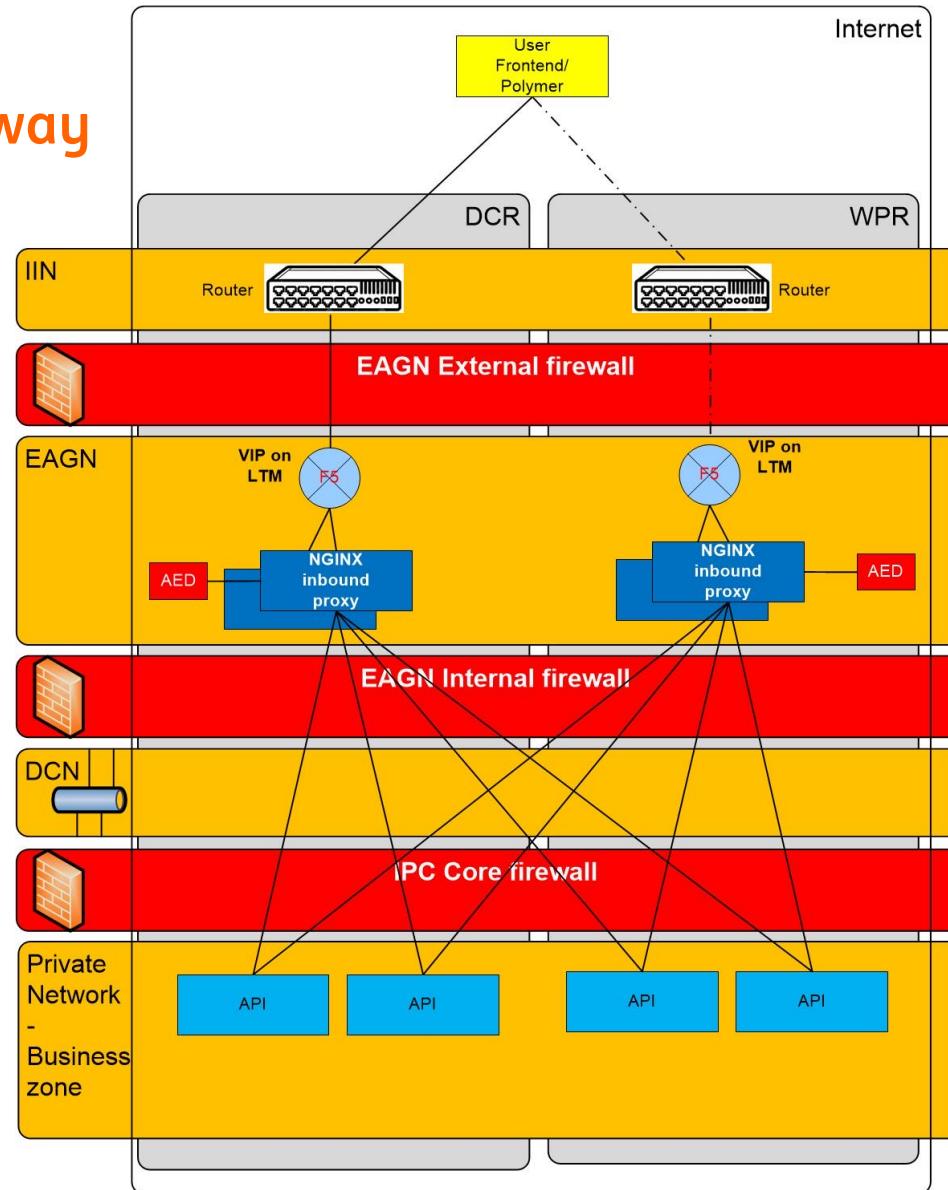
For RIAF (deprecated/decommmed) the SessionsApi is responsible for authentication

We have several gateways:

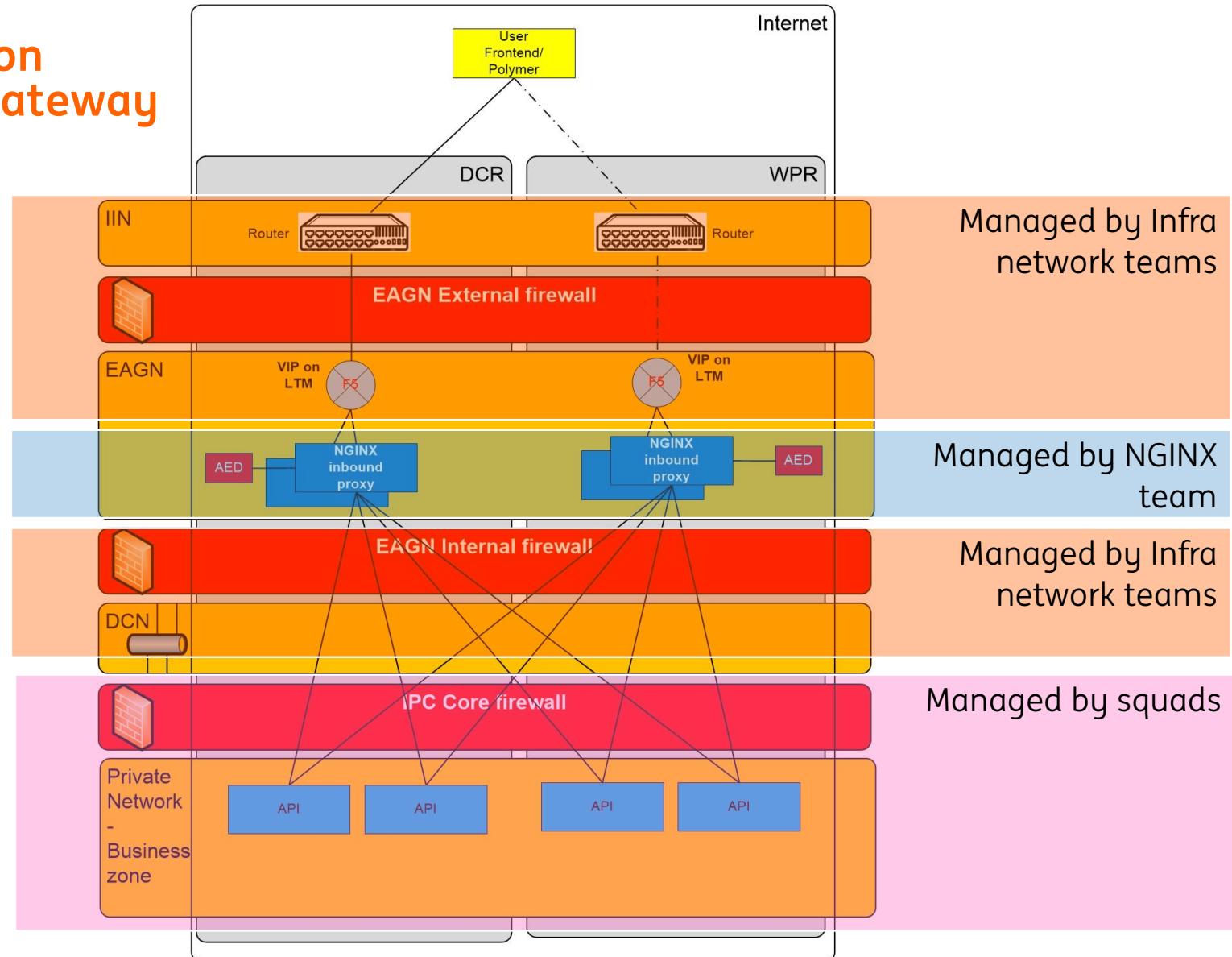
- **external API Gateway** for traffic coming from the internet
- **internal API Gateway** for traffic from an ING governed environment (ING offices, ING intranet etc.)
- **Gateways for translation** of the access token to context-object and vice versa



Implementation External API Gateway



Implementation External API Gateway



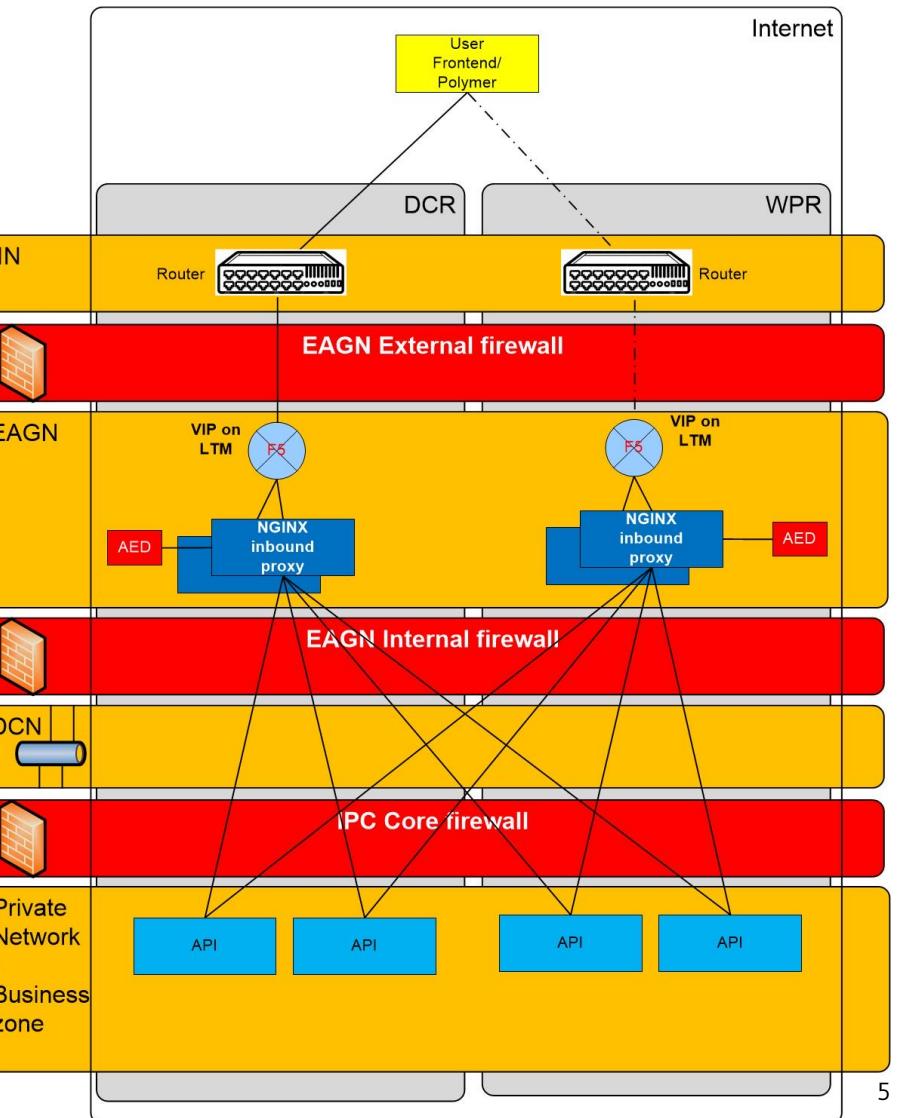
Implementation external API Gateway- NGINX Inbound

Operational dependencies

1. The **EAGN external** Firewall must be open
2. a **VIP** must be present on the LTM
3. A valid TLS certificate must be present on the LTM
4. NGINX must be configured correctly
5. The **EAGN internal** Firewall must be open
6. The firewall of the business zone (your PN or of ICHP) must be open
7. A TLS certificate must be present in the API

Delivery:

- 1, 2, 3, 4 are implemented by an **Azure board story** for the network teams. **Squads need to request (and manage!) a TLS certificate.**
- 5 is implemented by the NGINX team (with your request on **gateway.ing.net**)
- 6 is implemented by the squads (using the IPC / GSO portal)



Implementation external API Gateway- NGINX Inbound

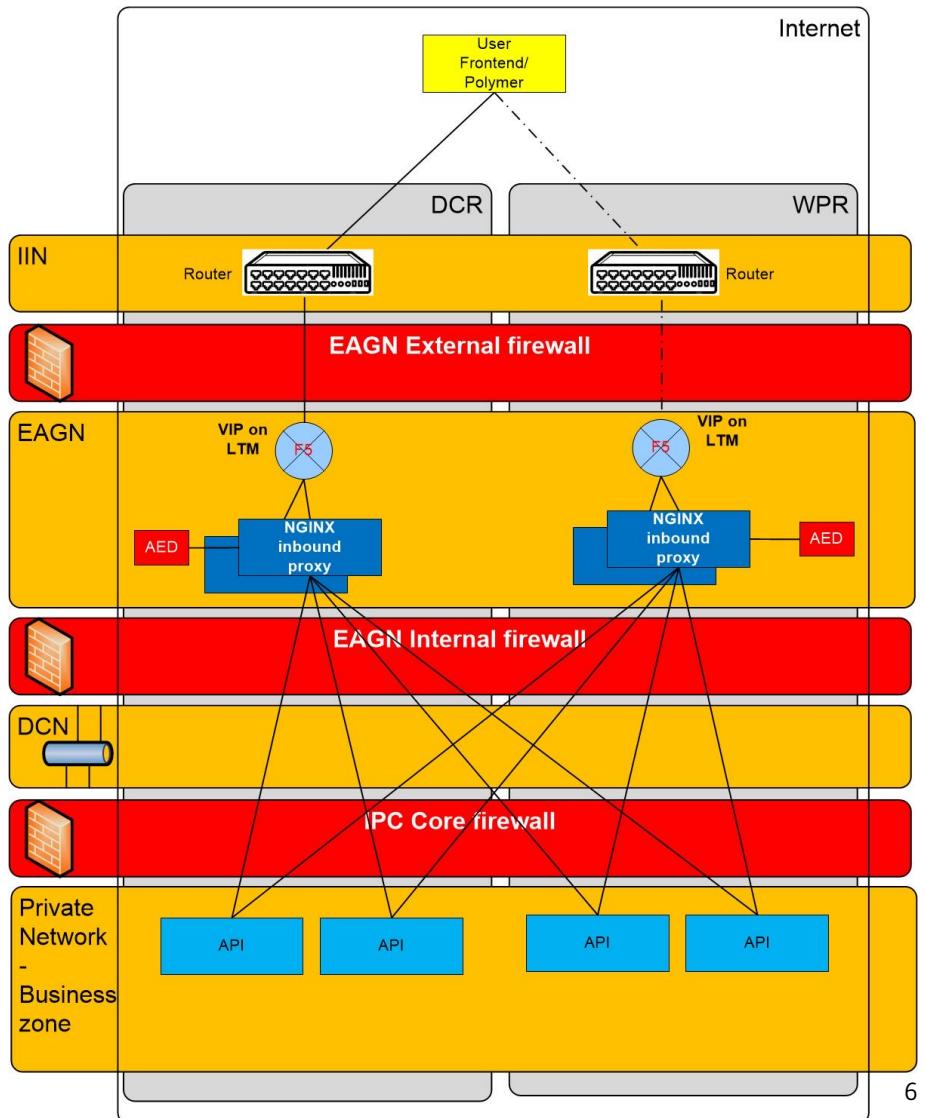
Operational dependencies

1. The **EAGN external** Firewall must be open
2. a **VIP** must be present on the LTM
3. A valid TLS certificate must be present on the LTM
4. NGINX must be configured correctly
5. The **EAGN internal** Firewall must be open
6. The firewall of the business zone (your PN or of ICHP) must be open
7. A TLS certificate must be present in the API

Delivery:

- 1, 2, 3, 4 are implemented by an Azure board story for the network teams. Squads need to request (and manage!) a TLS certificate.

Implementation is NOT yet automated,
can take up to 5 weeks



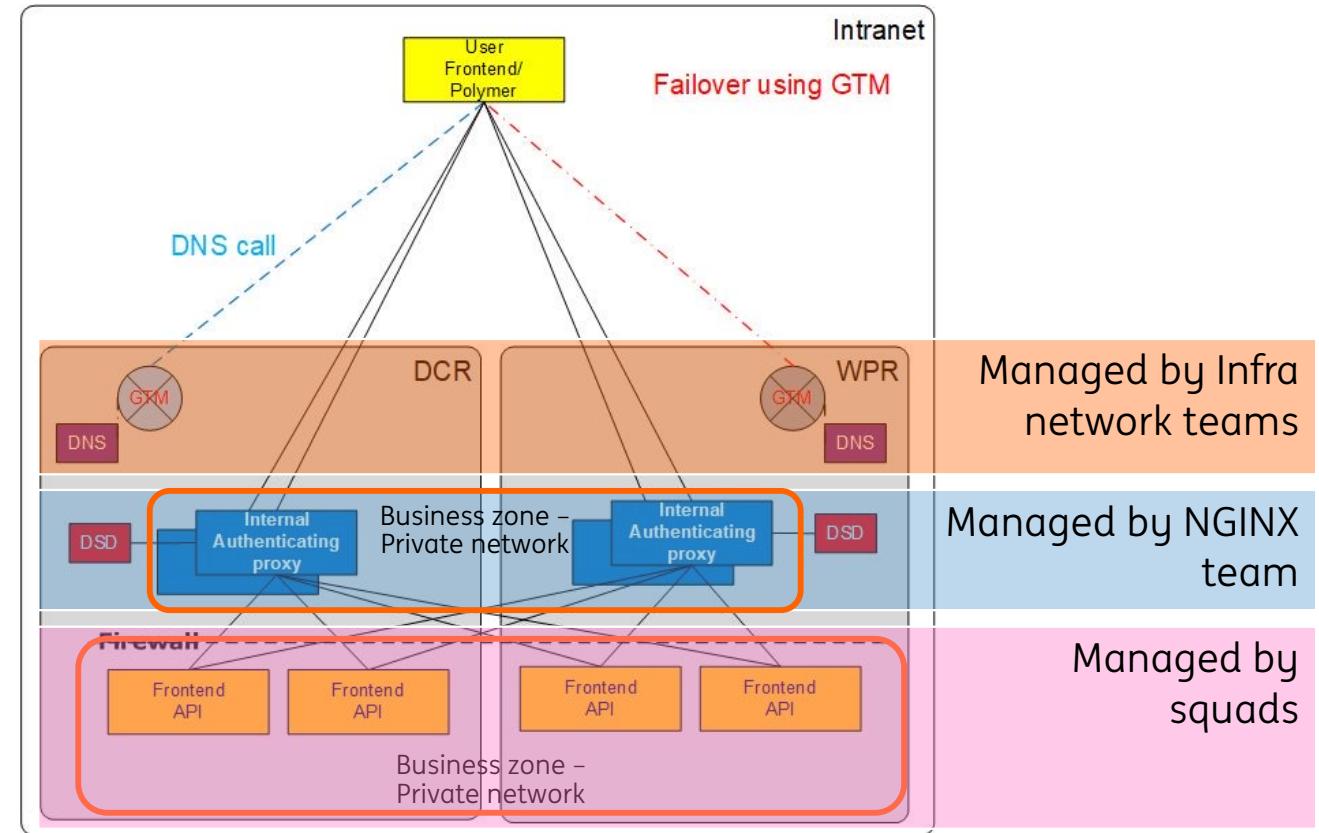
Implementation internal API Gateway

Operational dependencies

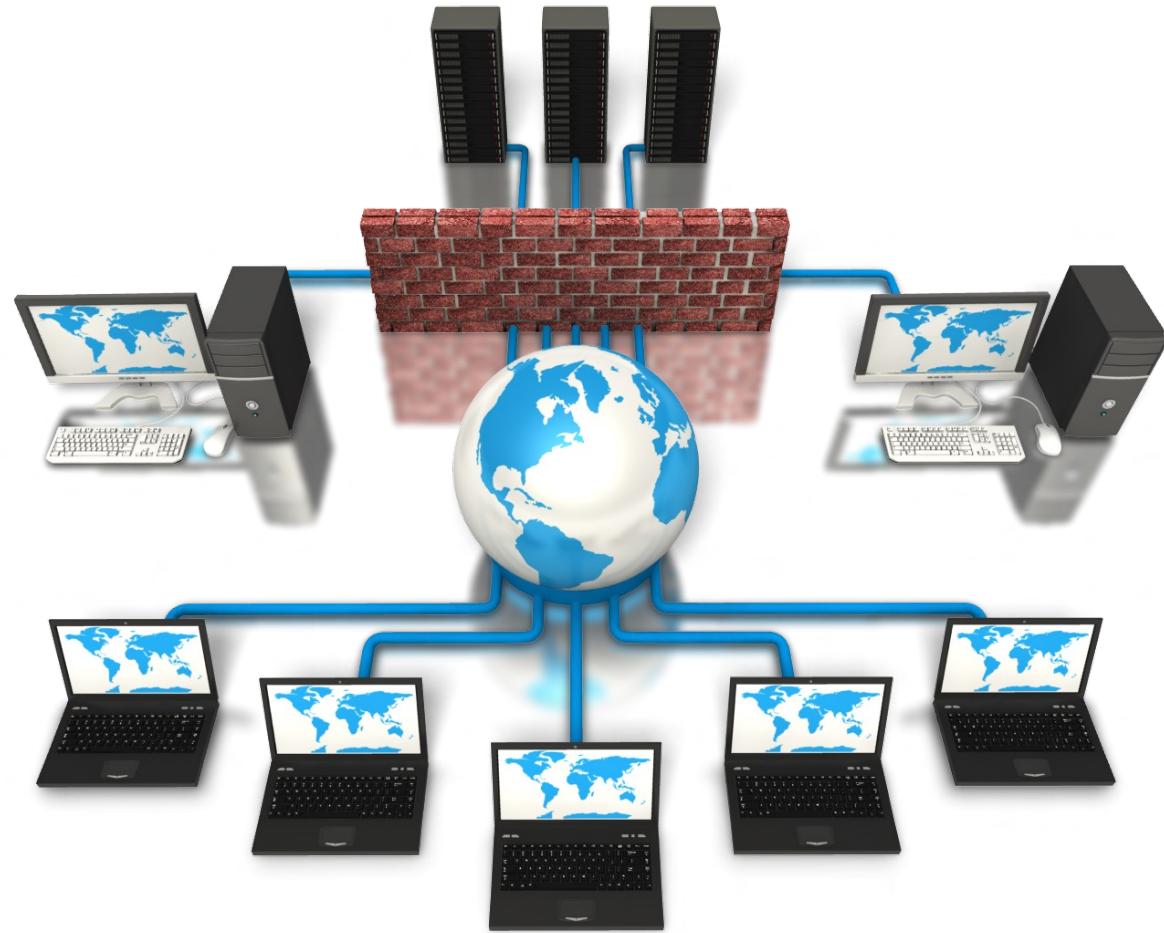
1. GTM configured correctly
2. A valid TLS certificate must be present on NGINX
3. NGINX must be configured correctly
4. Firewall must be opened from Gateway* to your PN
(firewall from userlan to internal API Gateway is default open on 443)

Delivery:

- 1, 2, 3: The NGINX team will request the GTM configuration **AND manage the TLS certificate**. Squad has to request the Gateway change on **gateway.ing.net**
- 4 is implemented by the squads (using the IPC / GSO portal)



*Please refer to the network & firewall page of the NGINX team for up to date network information



Network basics IPC and ICHP



Network zone – Private network

TCP/IP networks are the most common type of network today. With such a network, a number of computers or nodes can communicate with each other. Each TCP/IP packet transferred over the network contains a source and destination **IP address**.

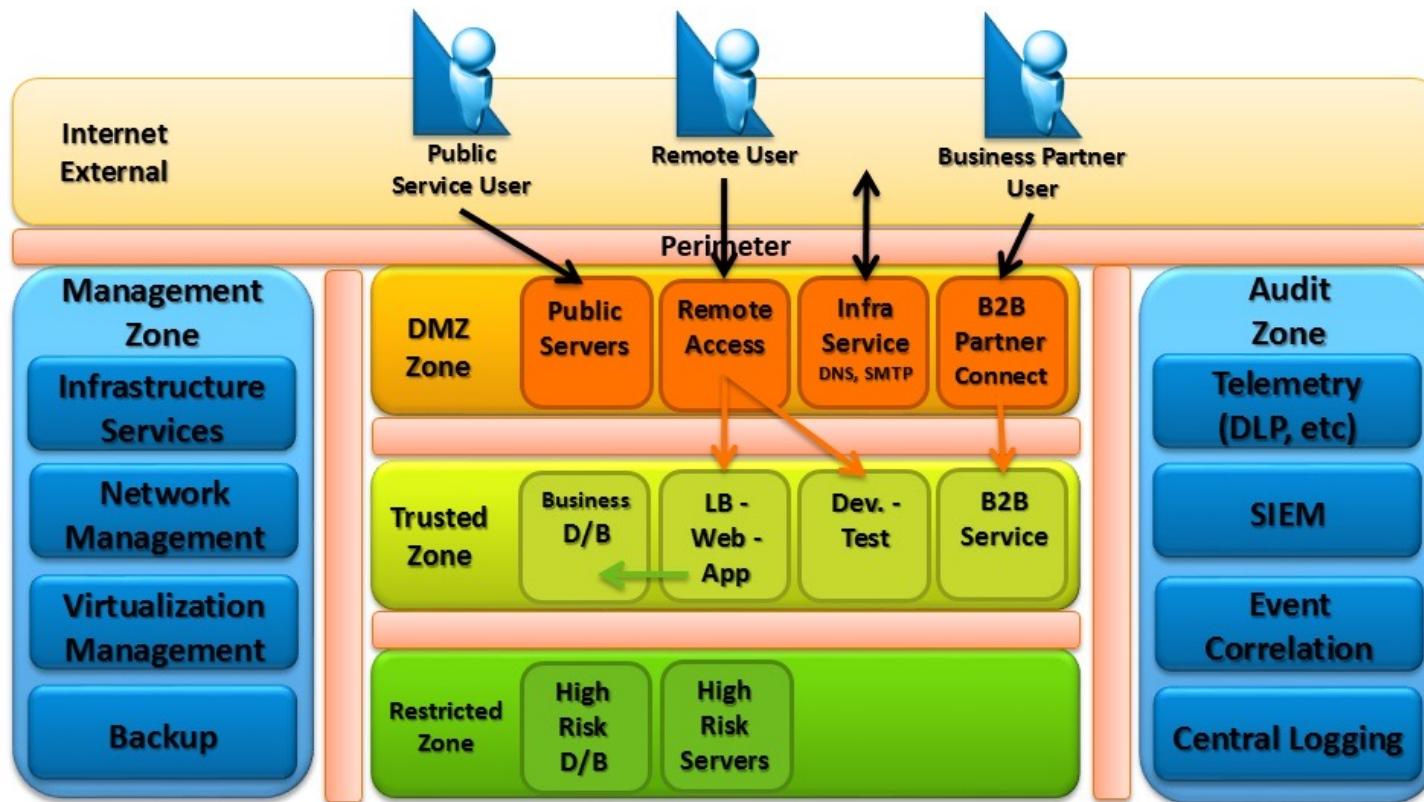
LANs (Local Area Networks) are networks usually confined to a **geographic area**, such as a single building or a college campus. A **VLAN** has the same attributes as a physical local area network (LAN), but it allows for end stations to be grouped together even if they are not located on the same network switch.

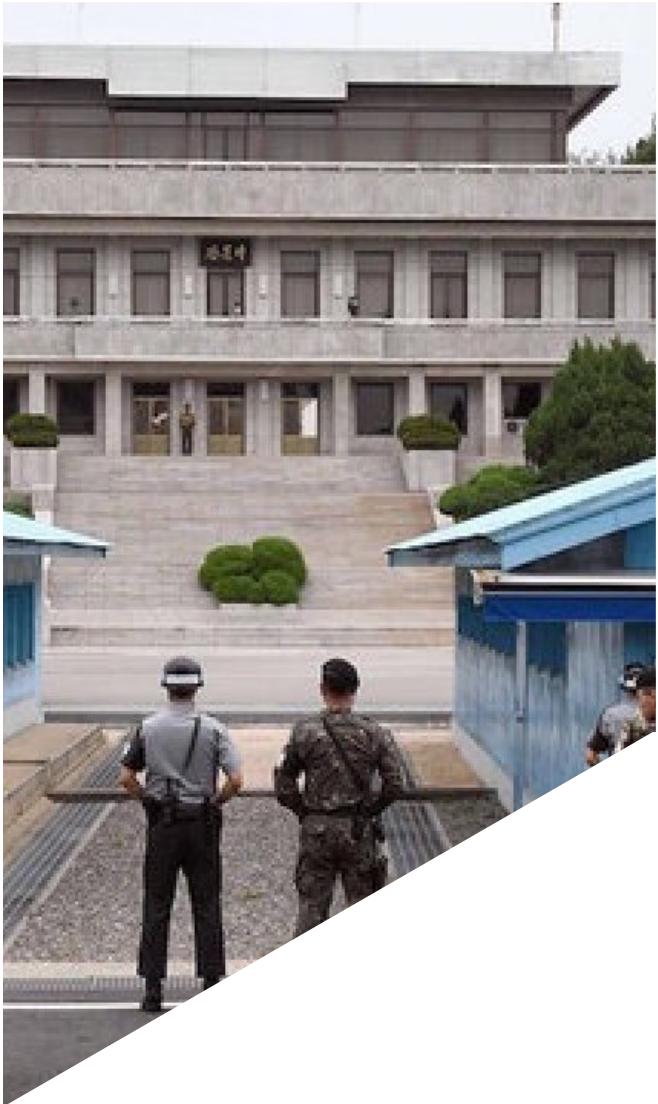
A logical grouping of nodes (computers) is also called a **network zone** or a **network segment**.

A **Private Network** in the IPC is a **network zone**.

Each network zone has a firewall to filter in- and outgoing traffic. WITHIN a network zone traffic is NOT filtered.

The perimeter / castle-wall model





Demilitarized Zone (DMZ)

A special **secured network zone**, connected with the **outside world**: the internet.

All **external facing** applications must connect through this zone (using a proxy like the API Gateway, XFB Gateway, Giba proxy etc.).

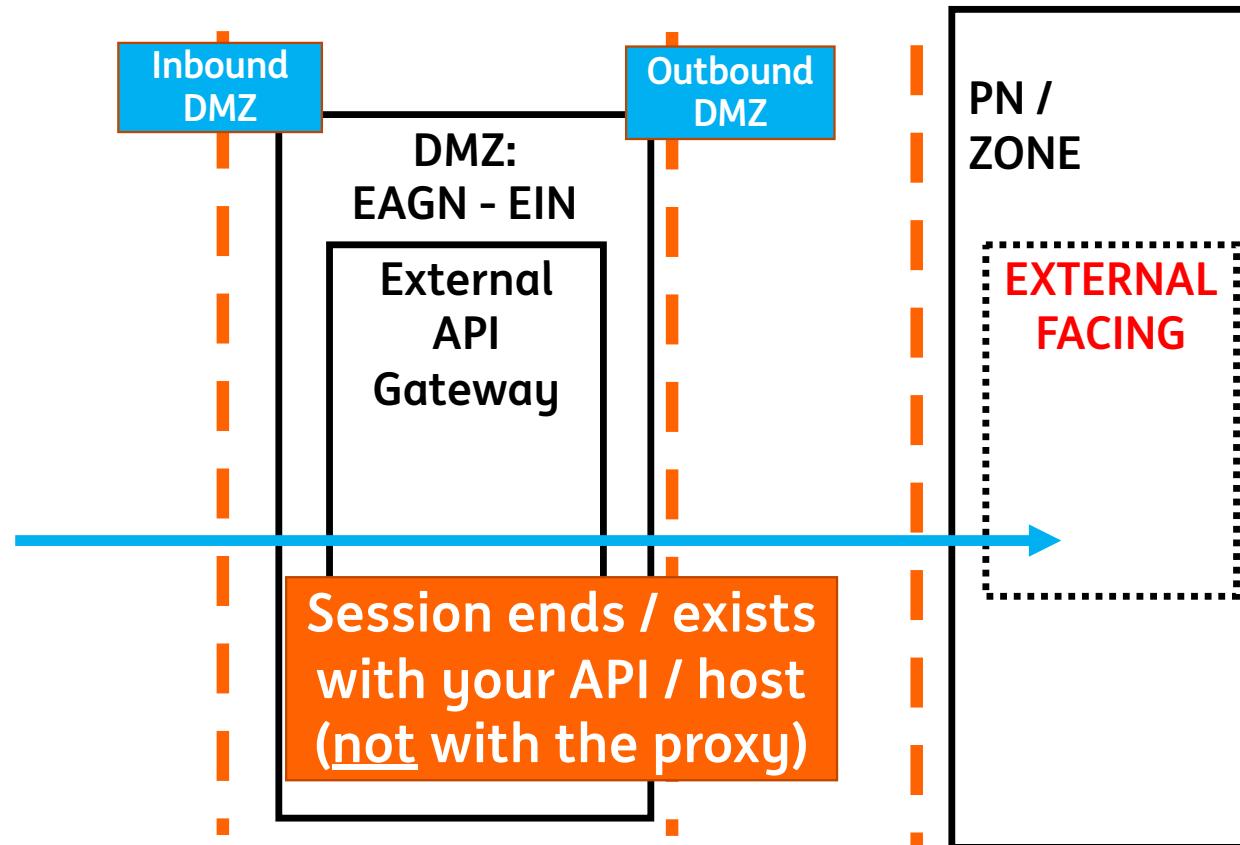
Machines/servers within the DMZ are **hardened**, meaning these machines are protected with special measures like security monitoring tools, intrusion detection tools etc.

What is external facing?

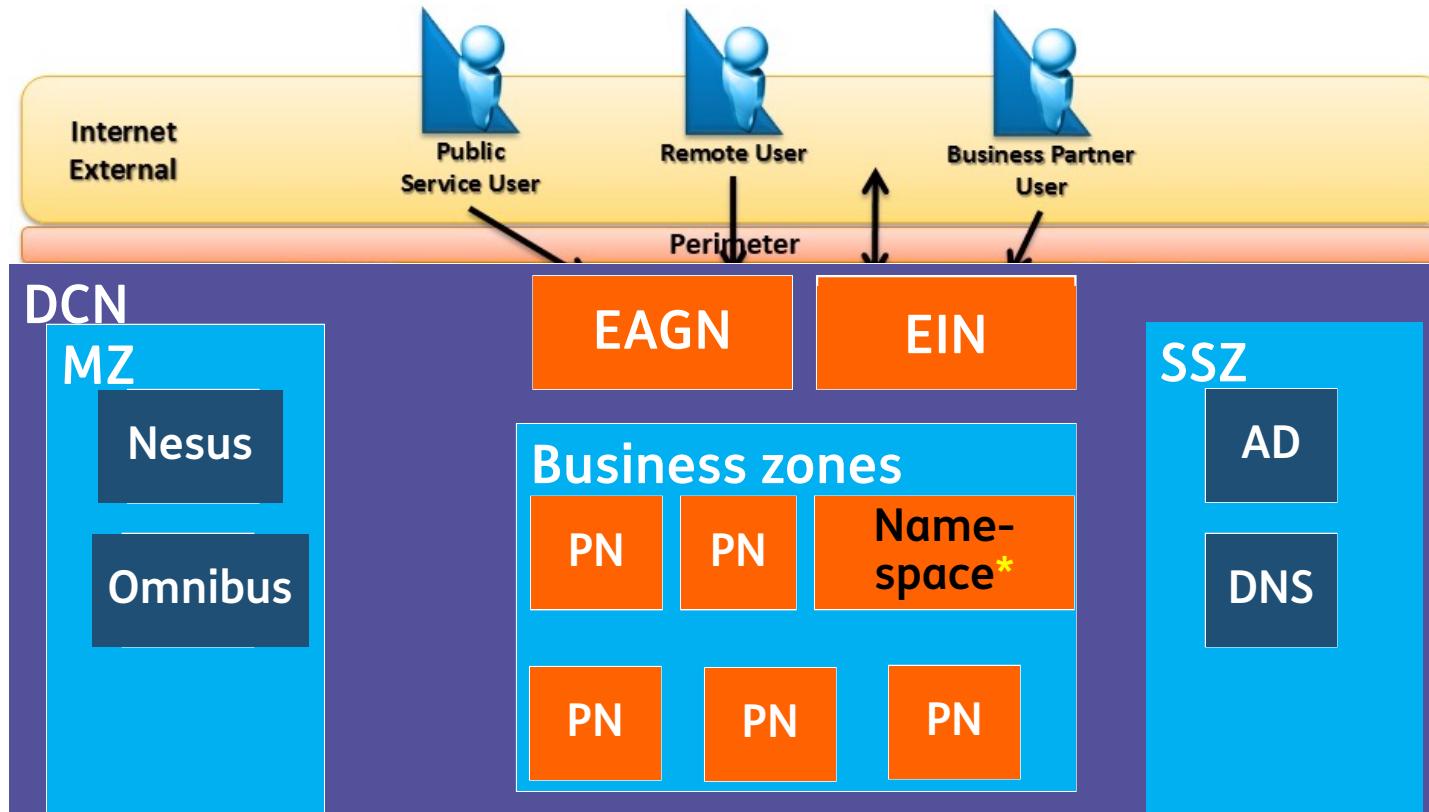
A connection between a non-ING governed IT environment (read: external party) and ING internal network is considered external facing.
This traffic must go through the DMZ: **EAGN** and **EIN**.

Traffic coming from ING access zones (offices, ING intranet and ING user lan) are NOT considered external facing

Session between external and your API



The perimeter / castle-wall model



* A Namespace in ICHP is NOT a traditional network zone as such, with a traditional firewall. However it is a grouping of apps with a single purpose, it is security boundary

ING Network zones

Management Zone (MZ)	Supporting Services Zone(SSZ)	External Application Gateway Network (EAGN)	External Interface Network (EIN)
<ul style="list-style-type: none">MZ hosts solutions aimed at <u>managing the ING IT-estate</u> for example with the purpose of vulnerability scanning.There is a general MZ firewall policy and a specific firewall policy to the solution can be implemented.	<ul style="list-style-type: none">SSZ hosts servers and appliances to <u>support ING applications</u> (AD, LDAP, DNS, DHCP, etc.)Every application hosted in the BZ should be able to establish a connection to the SSZ: <u>the firewall is default open for BZ</u>.	<ul style="list-style-type: none">Offers intermediary function between the internal and external network and the Internet.Typical DMZ workloads are reverse proxies (e.g. NGINX) with a north bound physical firewall and a south bound physical firewall.<u>You need a RCEC to open the firewalls</u>	<ul style="list-style-type: none">Used to connect to known Business Partners. The "known" means ING has some contractual arrangements in place to guarantee "good behavior".The connectivity is built using point to point connection using lines or VPN across the Internet.<u>You need a RCEC to open the firewalls</u>

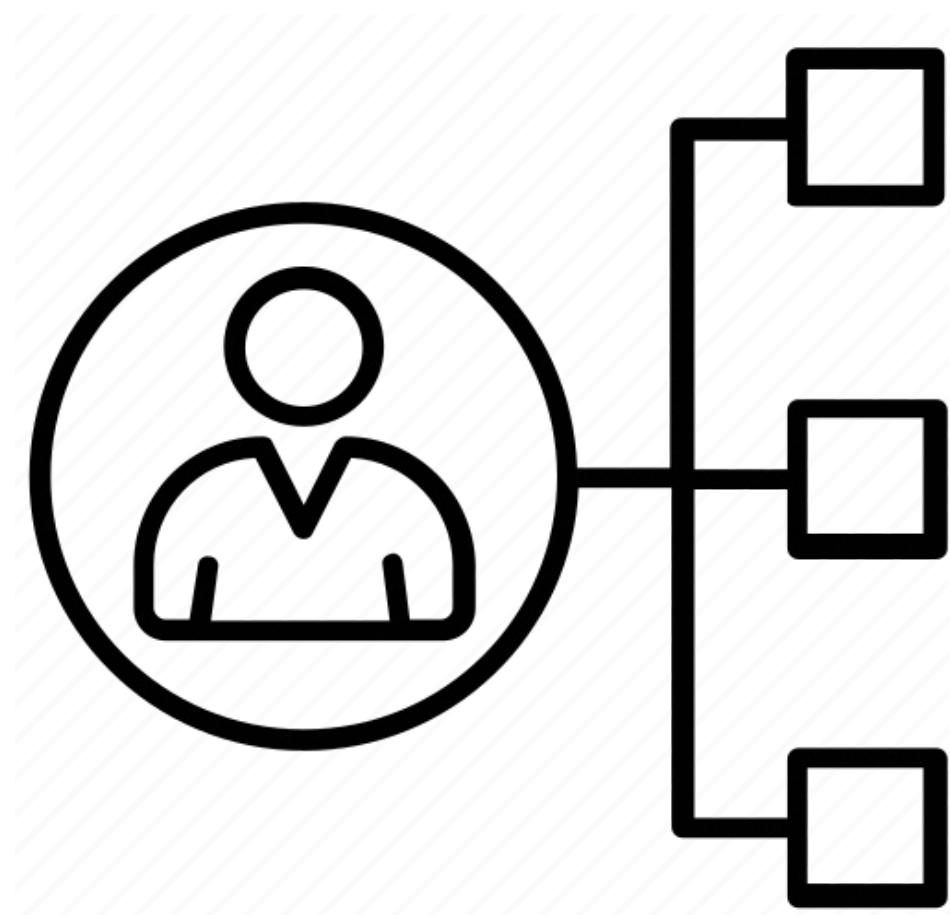
Business Zones (BZ)

Any zone where a DevOps team can deploy an application.

in IPC a business zone is called a **Private Network (PN)**, managed by the self-service portal.



Access networks (AN) - Userlan



Access networks are oftentimes called **Userlans (Local Area Networks)** in network architectures and are the network implementation at an ING owned or rented location(s).

A location could be an ING office, branch office, Point of Sale (POS) or an ATM.

We have different AN's depending on connection type (Wifi, Gras, LAN etc.) and location (country, branche etc.).



Firewall Basics

A Firewall acts as a gatekeeper and inspects traditionally the **5-tuple**:

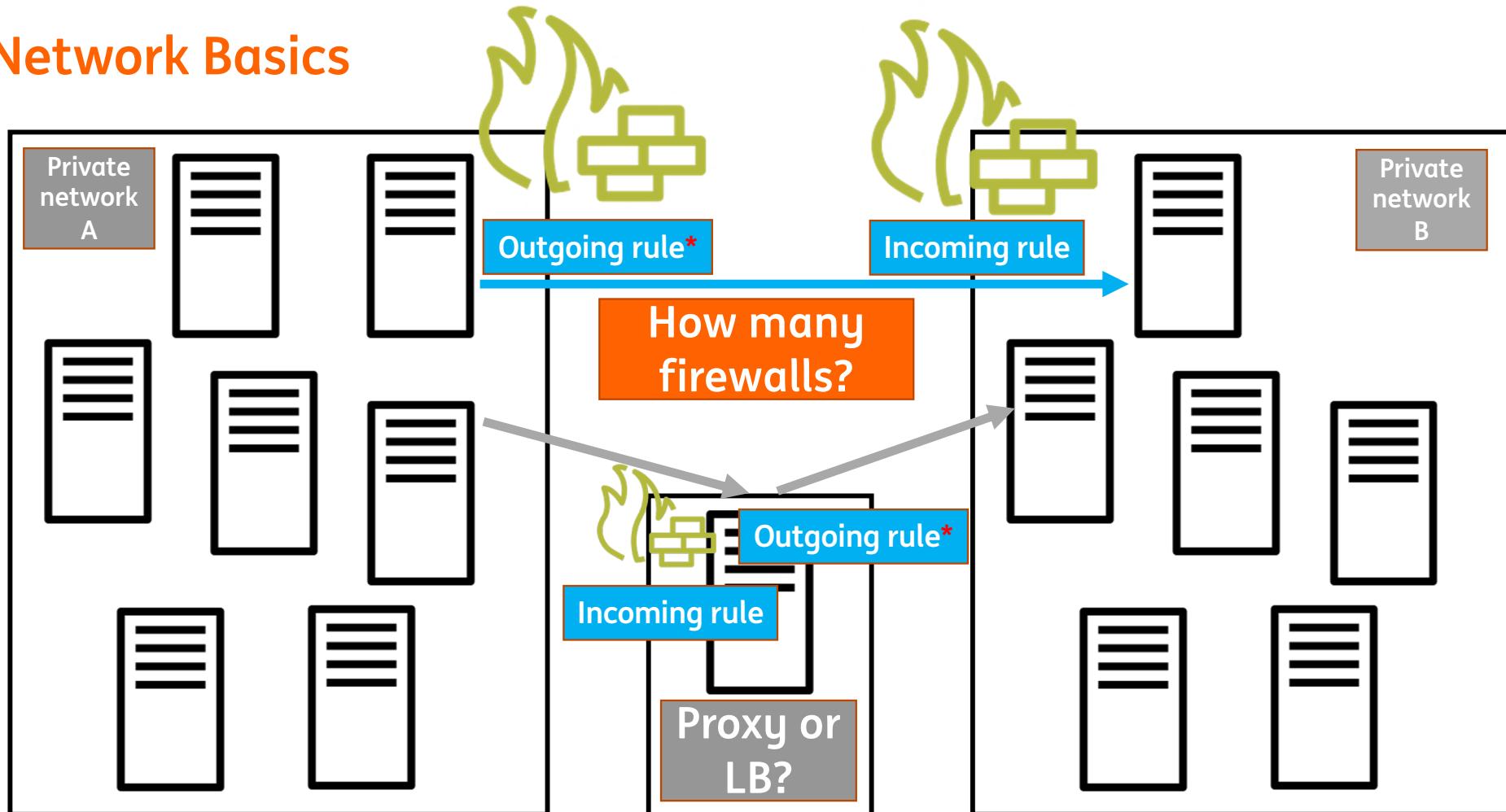
- source IP address/port number
- destination IP address/port number
- protocol

Firewalls can filter on **outgoing** and **incoming** traffic of a network zone.

Traffic within a zone is NOT filtered.

The firewall is using a **default deny policy** meaning if no **firewall rule** exists the communication is rejected.

Network Basics



* In IPC, at the moment, outbound filtering is disabled by default. It can be enabled in the Self-Service portal. It must be enabled for some external connections.

How many firewall rules?

Even though it is one click or one request in the self-service portal of IPC, multiple firewall rules might be created



Application aware firewall

The virtual firewalls within IPC have Application Based Security i.s.o IP based security, meaning the firewalls are “**application aware**”.

An application aware firewall or next-generation firewall (**NGFW**) detects and blocks malicious traffic previous generations could not; it can decrypt, adds context (is **application** and user **aware**) and uses **DPI** (deep packet Inspection).

For example, if you want to block Oracle traffic in the previous generation firewall you had to block **port 1521**. In a NGFW you can define a rule to block **Oracle**:

It won't be fooled by using a different port number, it recognizes the application!

Namespaces ICHP - Ingress

Network traffic to and from an ICHP namespace works a little bit different:

ALL incoming traffic (called **Ingress**) to ICHP has an Infra node (also known as ingress routers) as destination.

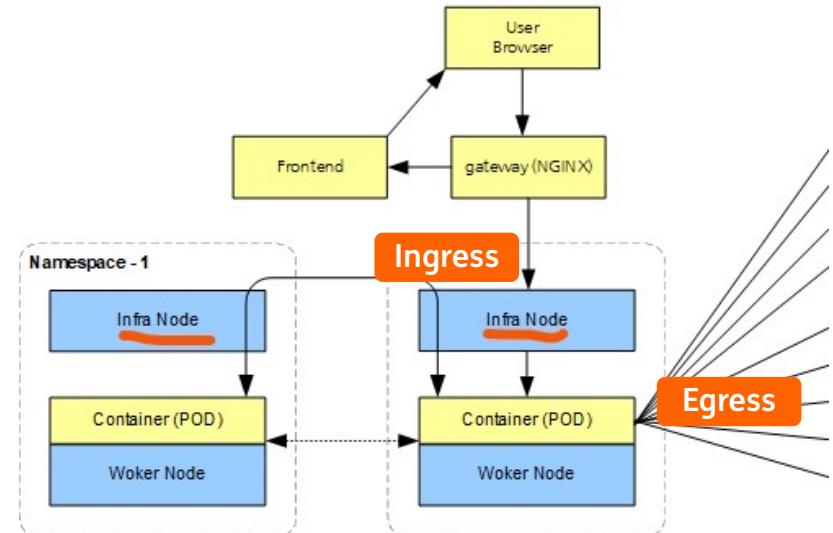
Namespaces in ICHP are not traditional network zones, ie. they do NOT have a firewall.

Traffic **within** ICHP is managed by **routes** (created by the Kingsroad pipeline for TPA applications).

For **traffic** from a network zone / a PN outside ICHP to a namespace within ICHP you need to configure **the ingress routers as destination** in the self-service portal, for example destination:

OpenShift Production on BMaaS DCR (new)

(please check the latest ICHP network info on confluence!)



Namespaces ICHP - Egress

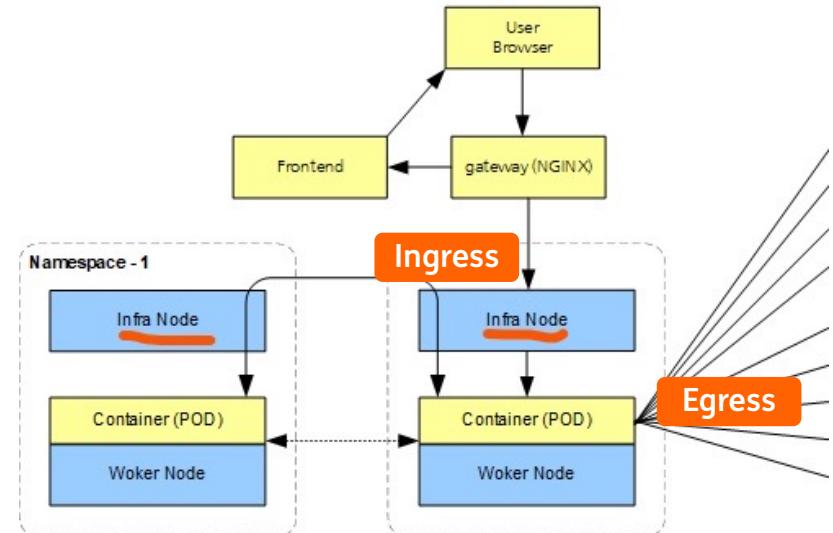
To enable automation of outgoing traffic flows from an ICHP namespace to a private network in IPC:

- namespaces are defined as Private Networks in the self-service portal.
- **the IP addresses of your namespace (pods) are NAT-ted**: your outgoing IP-address is changed!

It is NOT 172.X.X.X
But 10.X.X.X

Do a NSLOOKUP, for example:
`nslookup aonlapi-prd.pod.dcr.prod.ichp.ing.net`

The ICHPv2 DTA egress IP's resolve to DNS entries:
<namespace>-<environment>-1-<cluster number>.egress.ichp.ing.net



ICHP outgoing connectivity issues

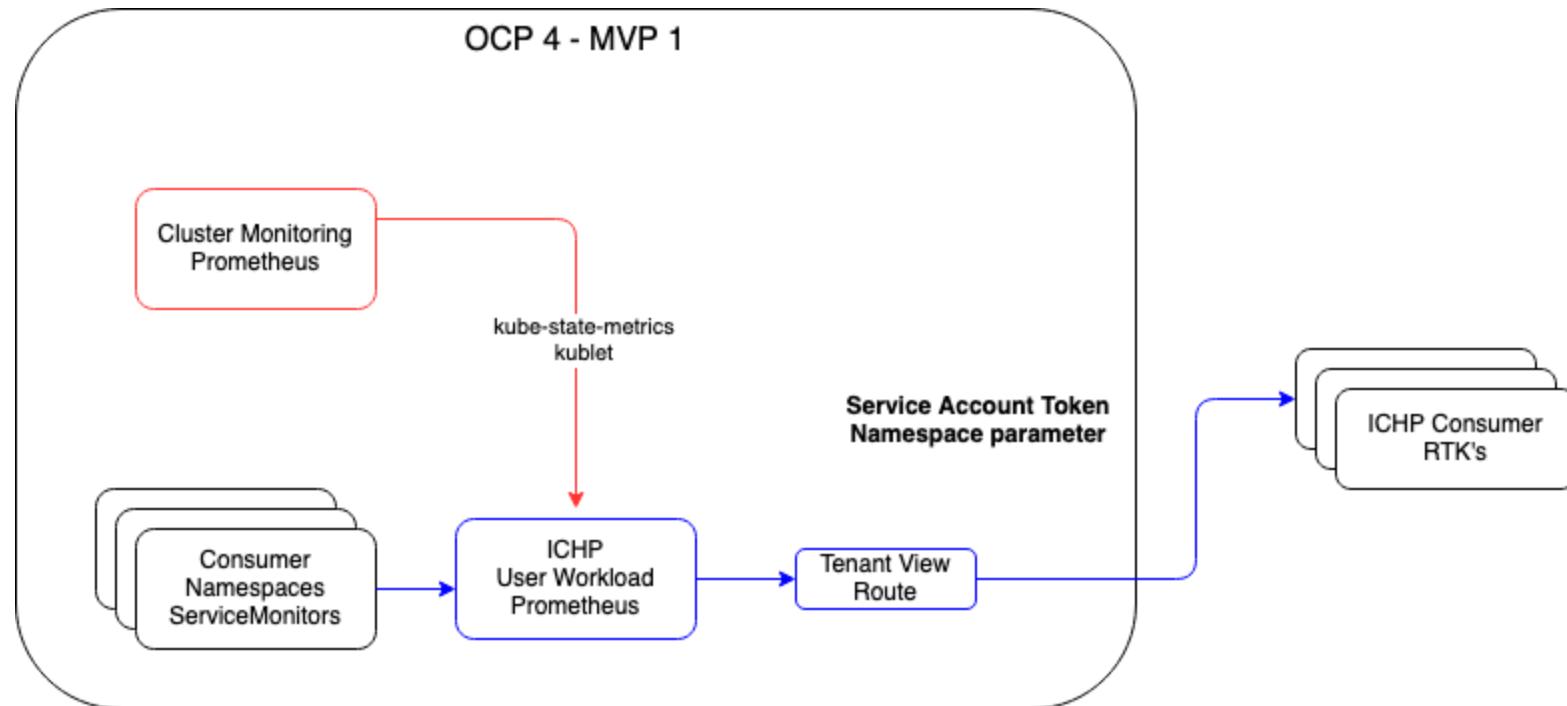
Use the correct source IP addresses (for ICHP the NAT-ted ones) and destination IP-addresses in communication with network engineers when discussing ICHP outgoing connectivity issues

Capacity management

```
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: nginx-stable
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: nginx-stable
  minReplicas: 2
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
    target:
      type: Utilization
      averageUtilization: 50
```

Container management

Performance monitoring



Back-ups and restores

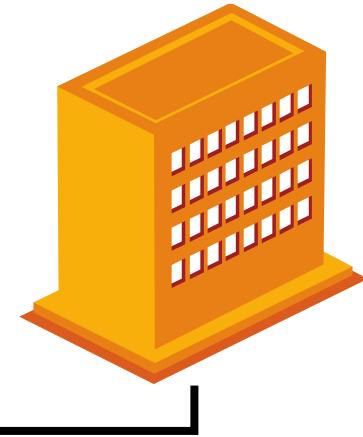
Patches and change strategy using IPC tools

IPC Datacenters and availability zones

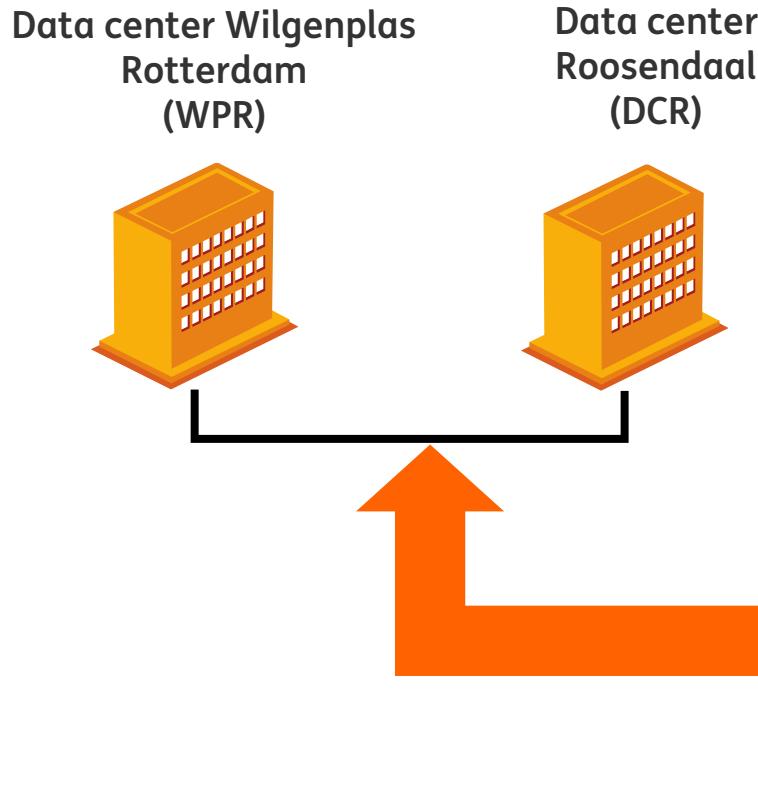
Data center Wilgenplas Rotterdam
(WPR)



Data center Roosendaal
(DCR)



IPC Datacenters and availability zones



Catalog Deployments Inbox

8 RHEL8 Native OS (1.0.0) | Business group P00415_CloudBootcamp1_bkif

Service Description Deployment Type **Deployment Target** Deployment Size Backup Pricing

Deployment target

The deployment target page contains all the information necessary for selecting the placement in terms of location, availability and network. The Compute Cloud is the first choice to determine placement. Based on the resources that are linked to the Compute Cloud you can have Disaster Recovery (DR). Based on the selected Compute Cloud and DR the Availability Zones are populated. Based on the selected Availability Zone the corresponding Sub Availability Zones are populated. The Search Private Network input box allows you to filter the Private Network list by inserting a text and pressing enter.

Compute

* Select Compute Cloud: ipc-default-cloud

* Select Disaster Recovery: No

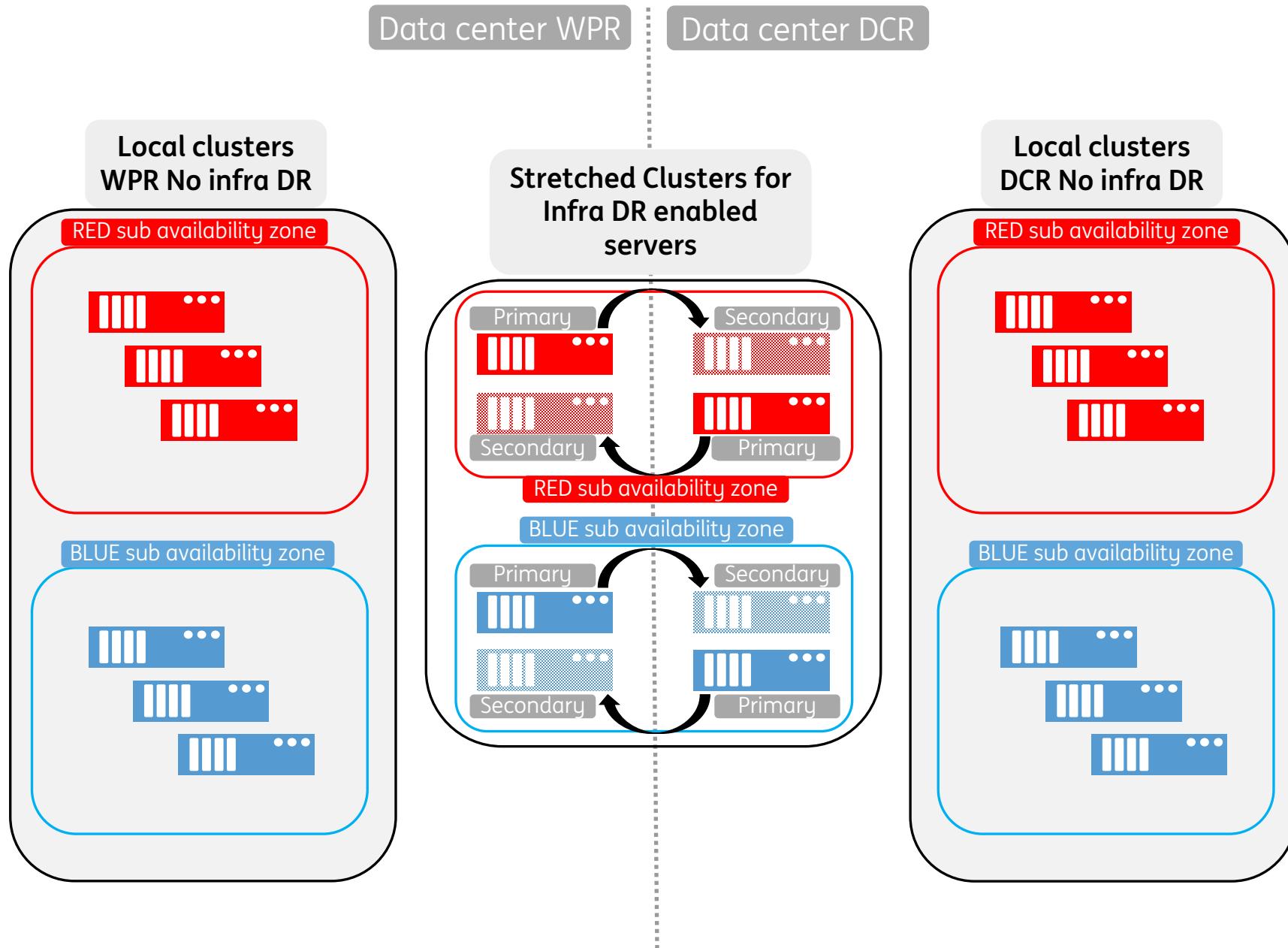
* Availability Zone: <None>

* Sub Availability Zone:

- <None>
- No preference
- DCR
- WPR

Search Private Network:

* Private Network:



Deployment target

The deployment target page contains all the information necessary for selecting the placement in terms of location, availability and network. The Compute Cloud is the first choice to determine placement. Based on the resources that are linked to the Compute Cloud you can have Disaster Recovery (DR). Based on the selected Compute Cloud and DR the Availability Zones are populated. Based on the selected Availability Zone the corresponding Sub Availability Zones are populated. The Search Private Network input box allows you to filter the Private Network list by inserting a text and pressing enter.

Compute

* Select Compute Cloud: ipc-default-cloud

* Select Disaster Recovery: No

* Availability Zone: WPR

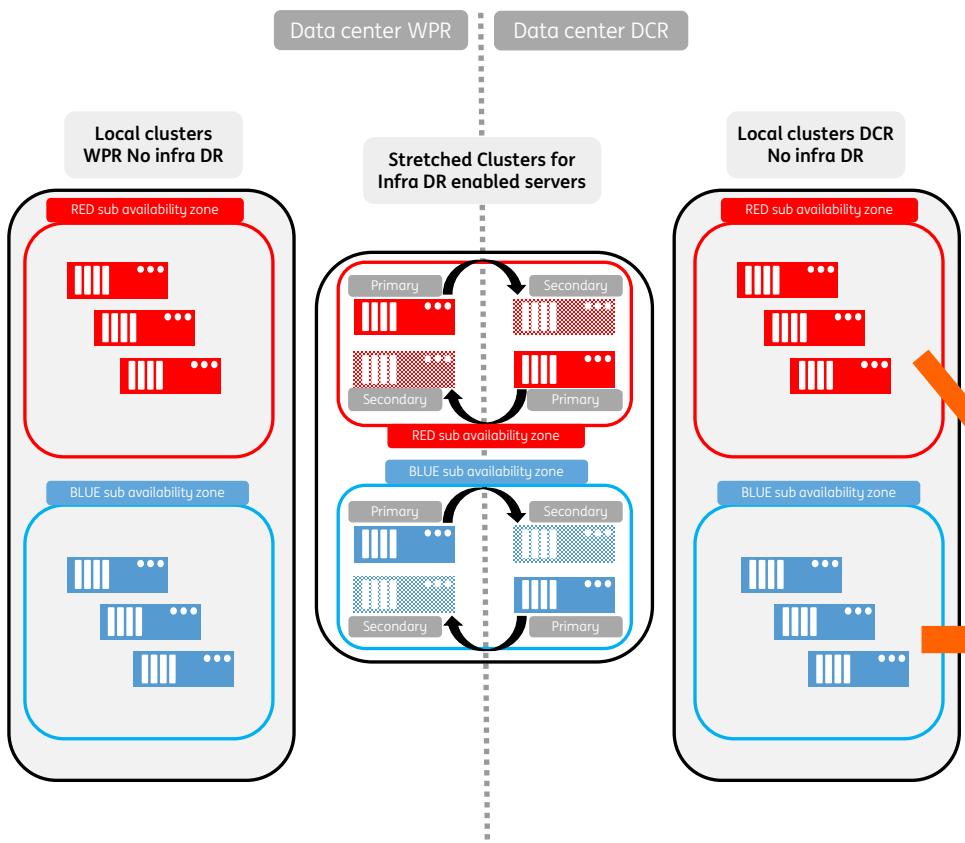
* Sub Availability Zone:

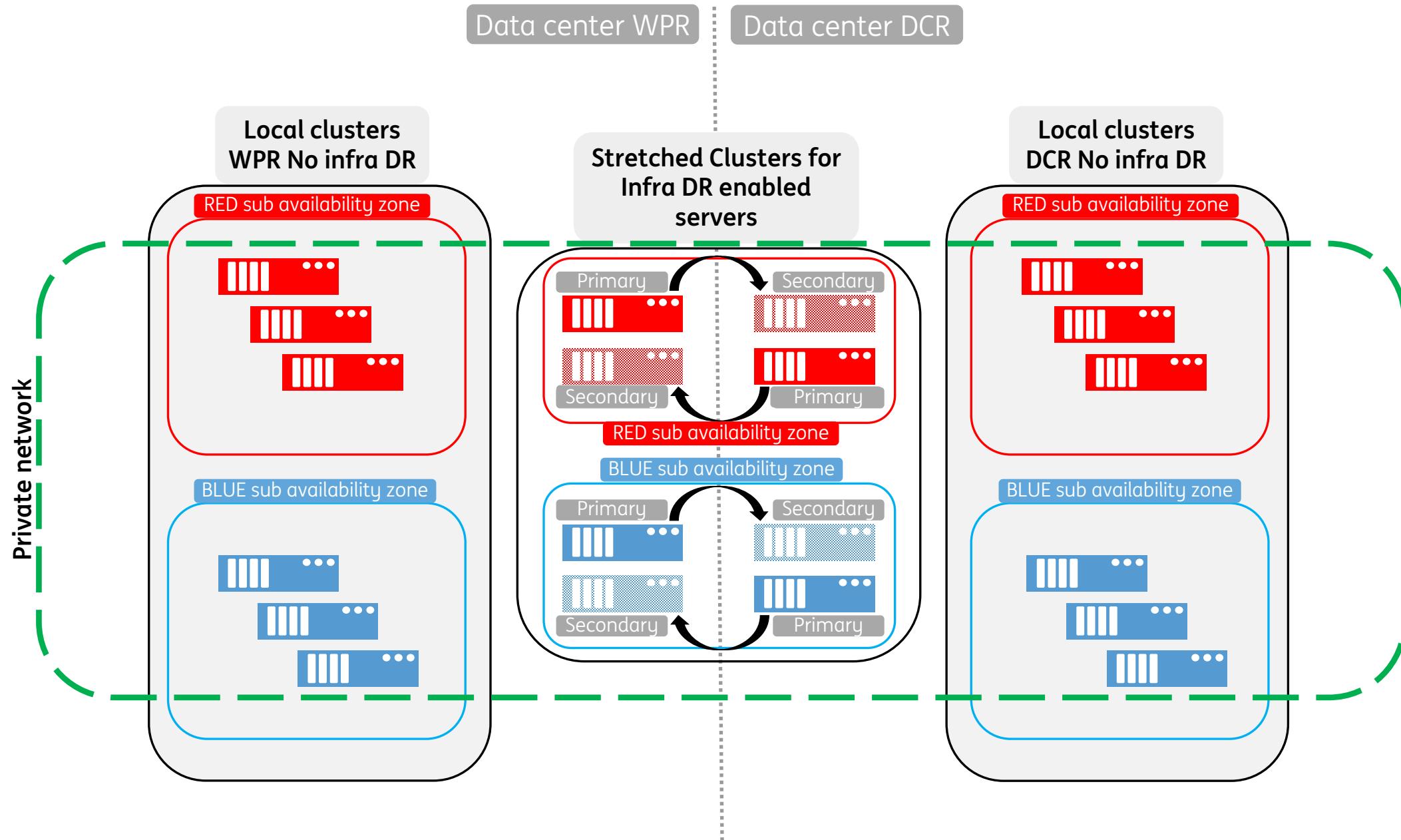
<None>

No preference

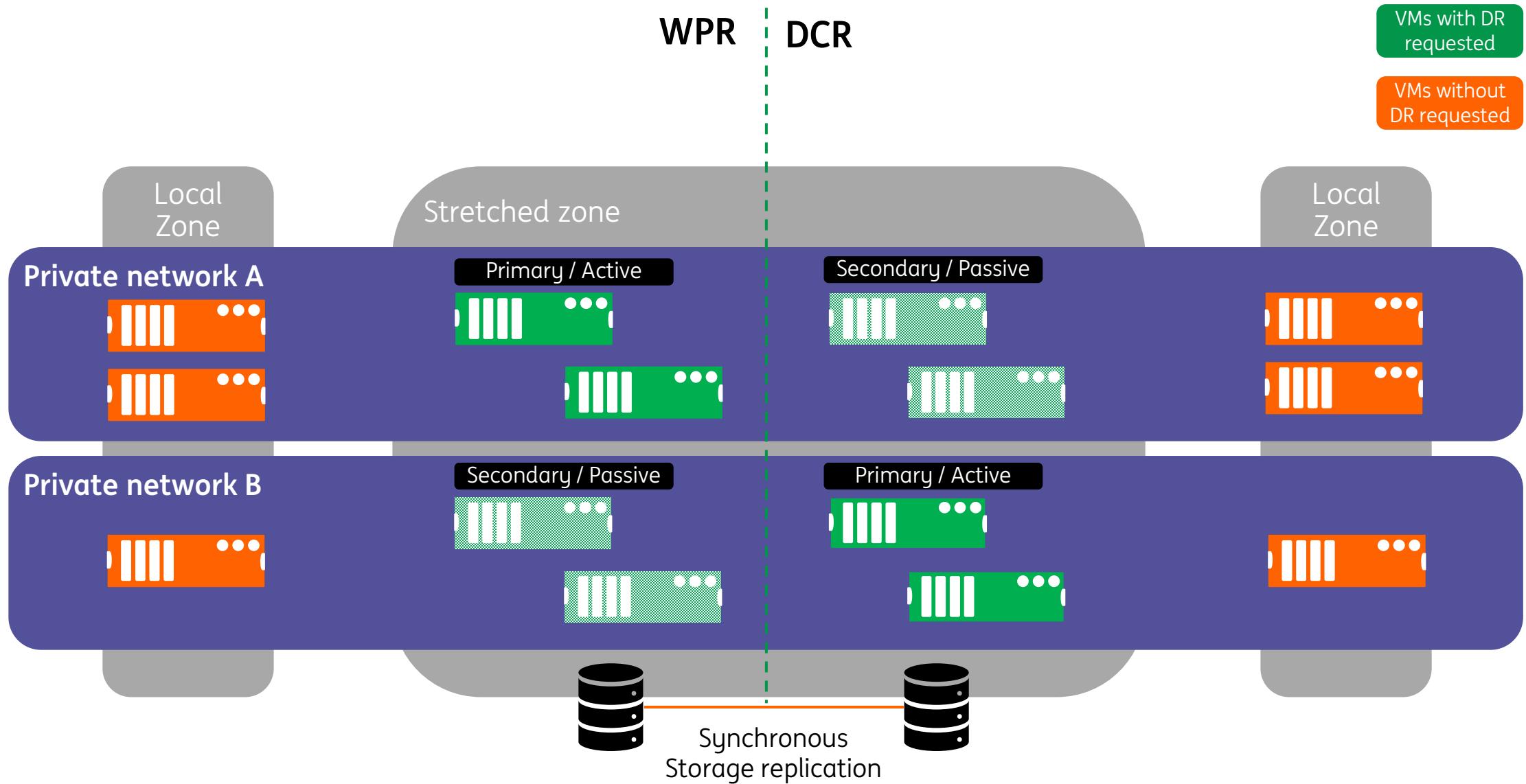
BLUE

RED

Network



IPC infrastructure Disaster recovery



Primary location for DR enabled workloads

Compute

① * Select Compute Cloud: ipc-default-cloud

② * Select Disaster Recovery: No

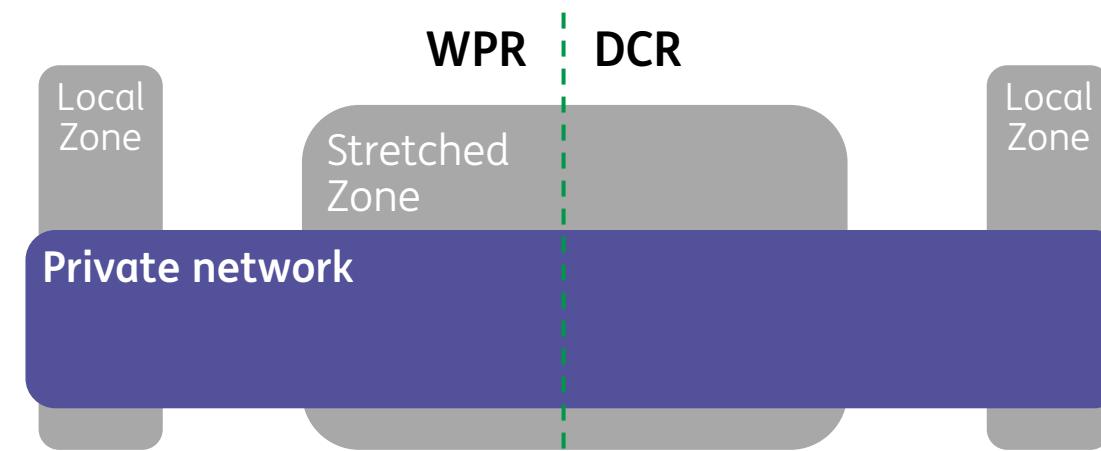
③ * Availability Zone:

④ * Sub Availability Zone:

⑤ * Select Compute Cloud: ipc-default-cloud

⑥ * Select Disaster Recovery: Yes

⑦ * Sub Availability Zone:



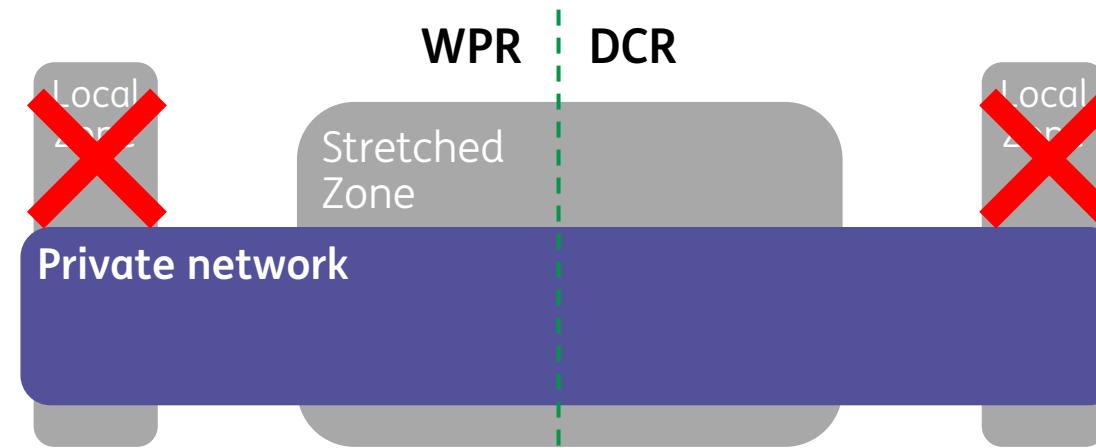
Primary location for DR enabled workloads

Compute

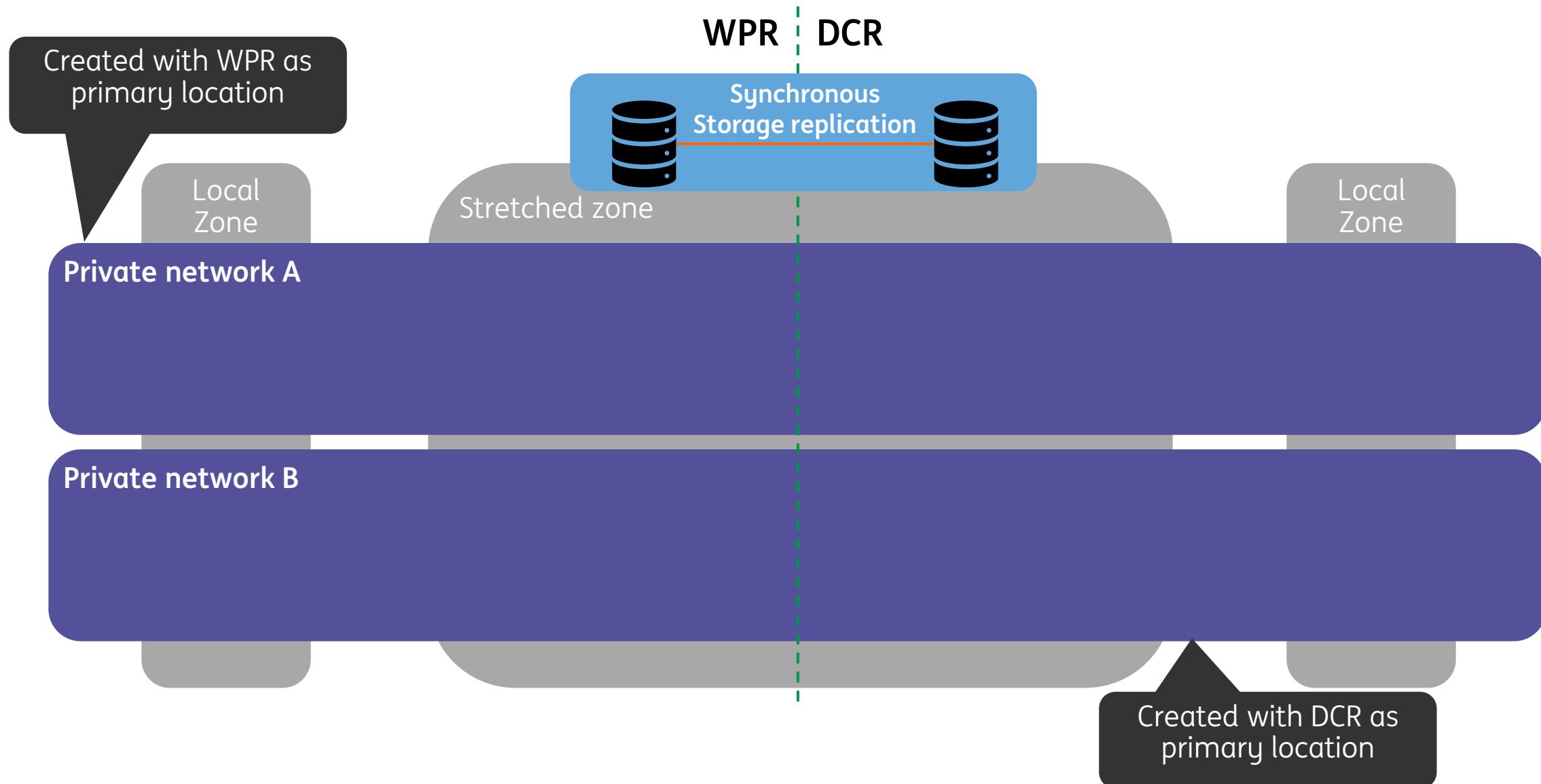
• * Select Compute Cloud: ipc-default-cloud

• * Select Disaster Recovery: Yes

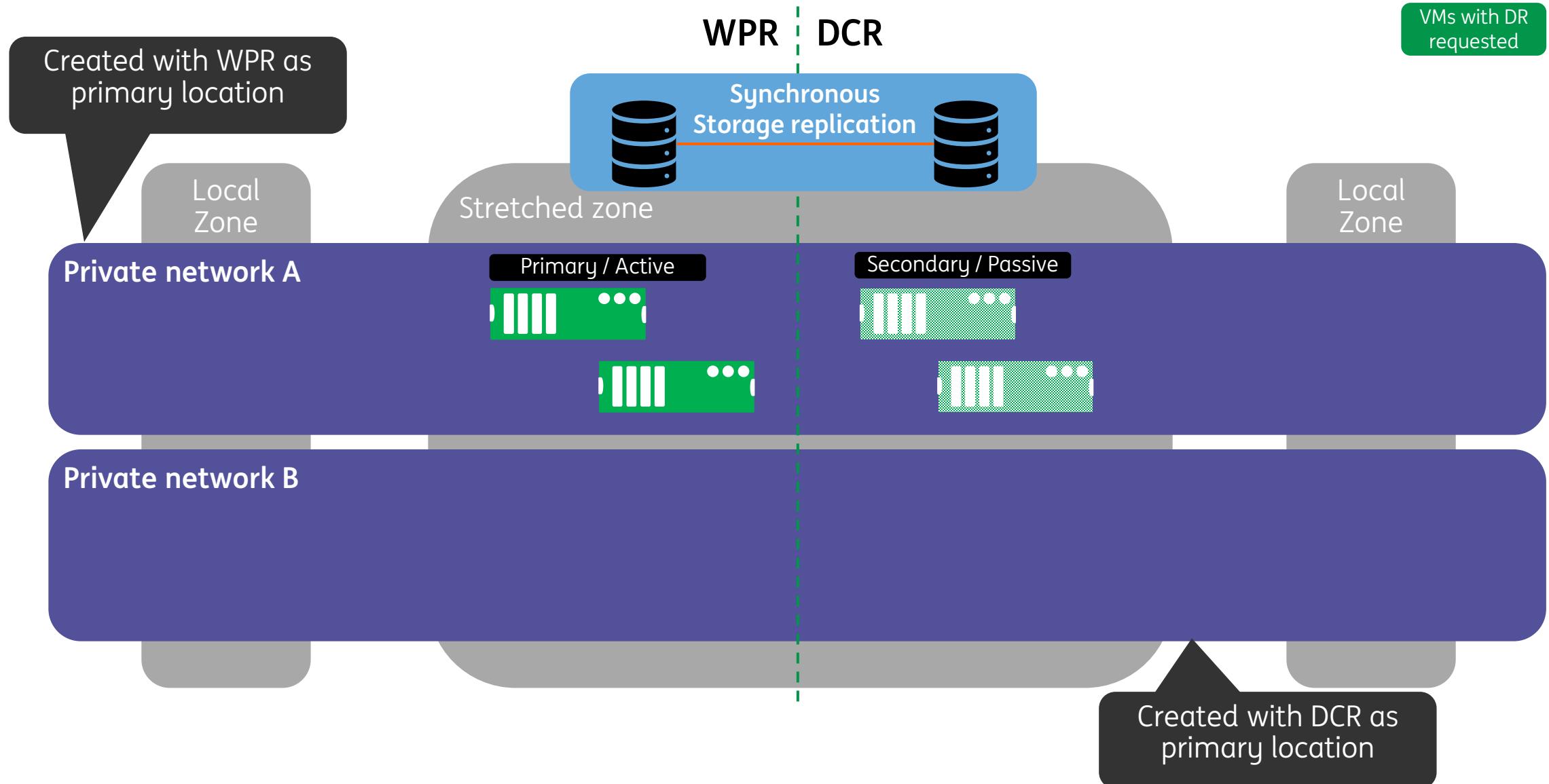
• * Sub Availability Zone:



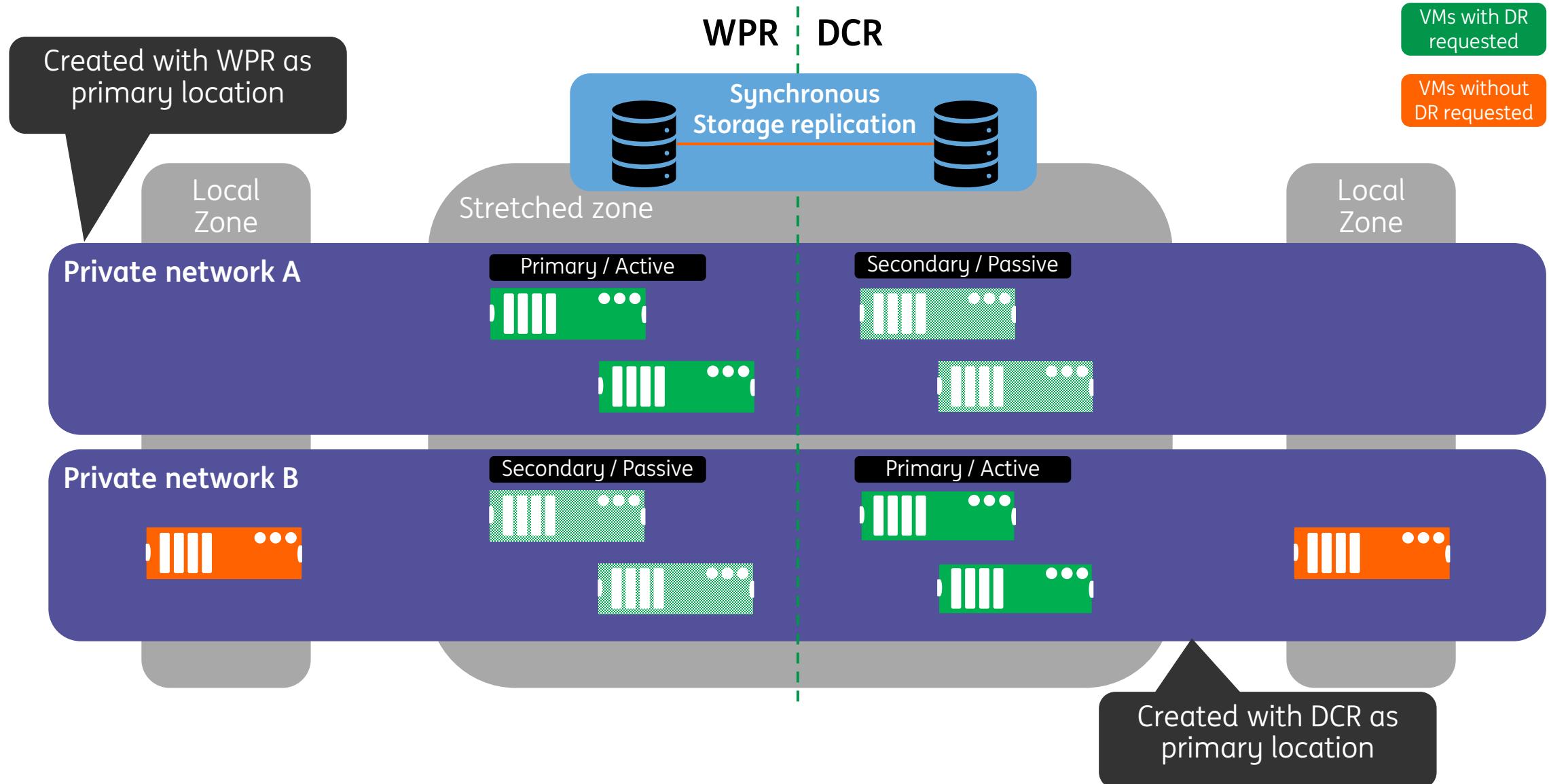
Primary location for DR enabled workloads



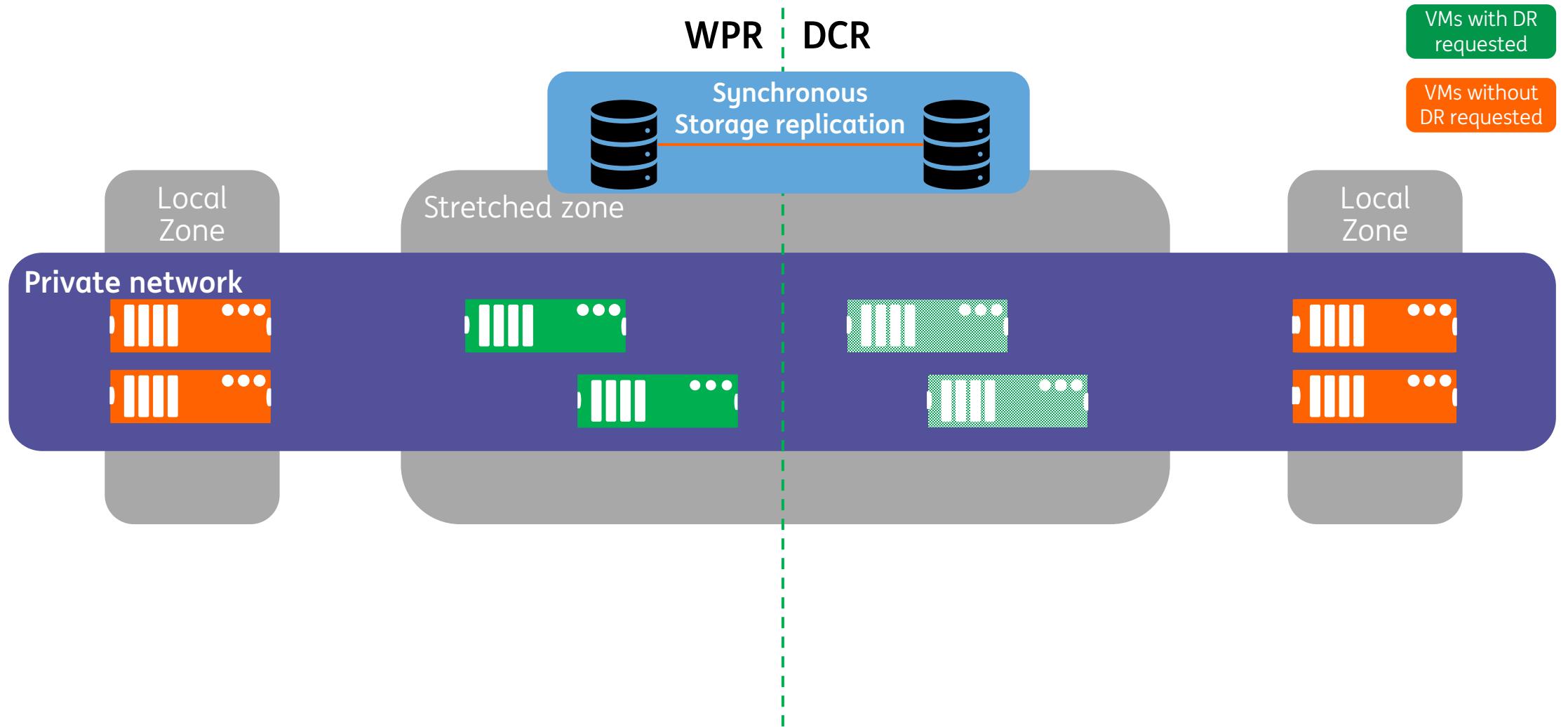
Primary location for DR enabled workloads



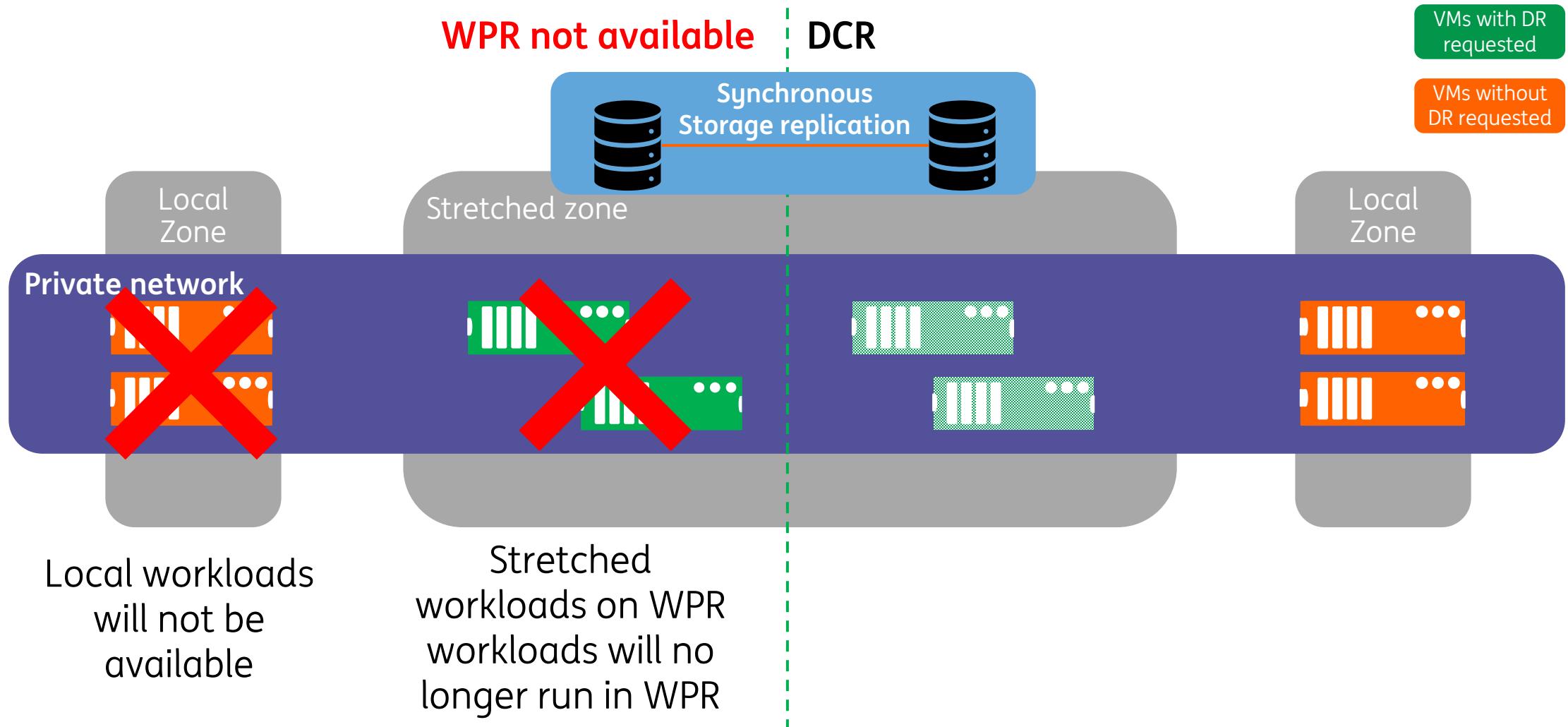
Primary location for DR enabled workloads



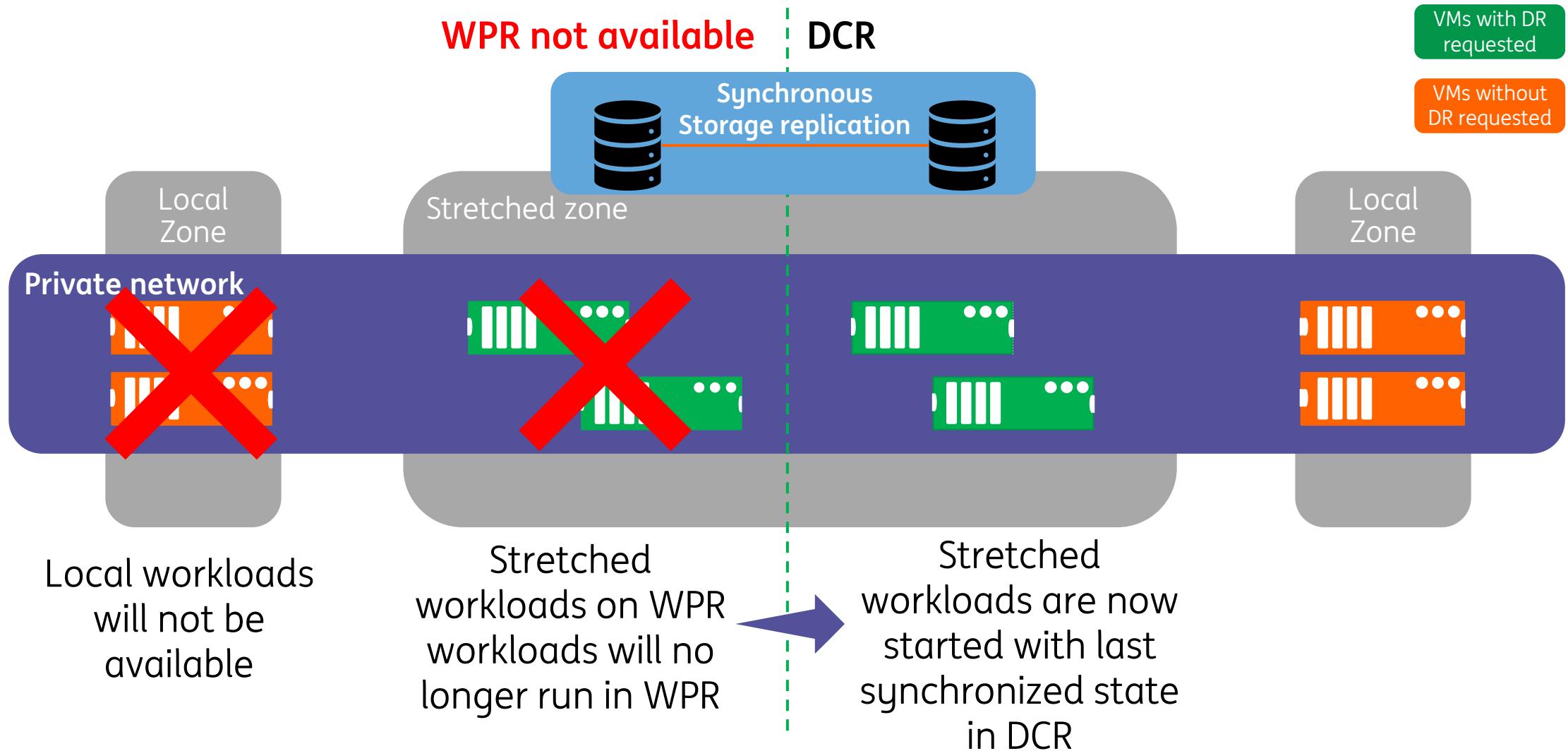
In case of data center outage



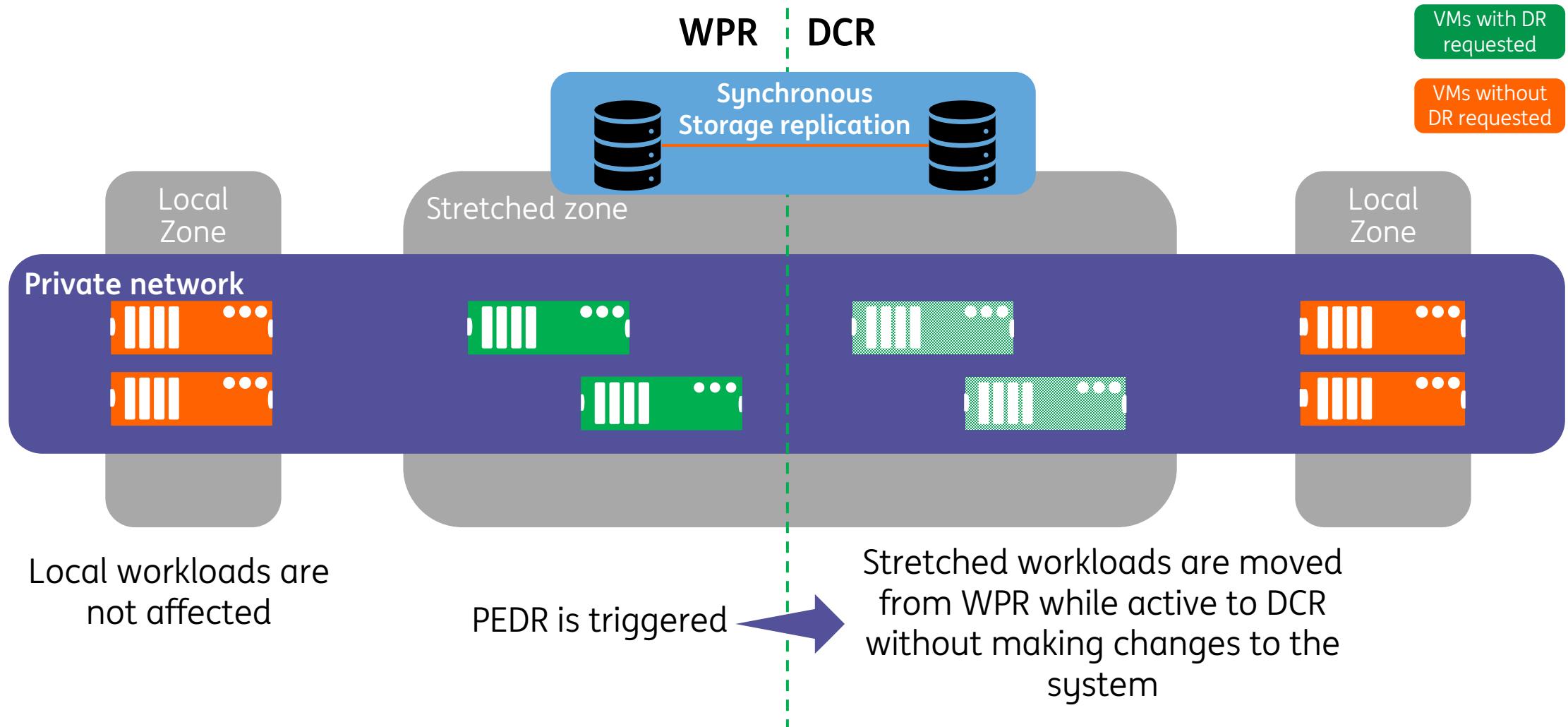
In case of data center outage



In case of data center outage

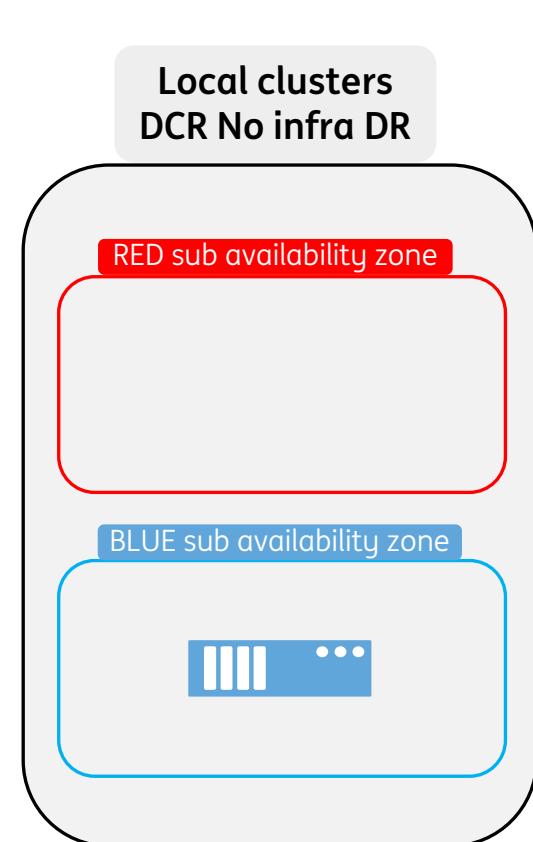
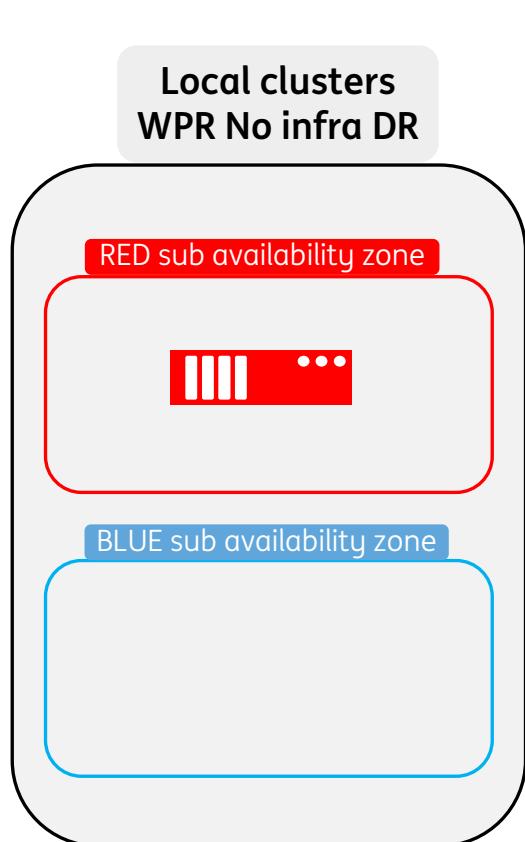


Production Exercise Disaster Recovery (PEDR)

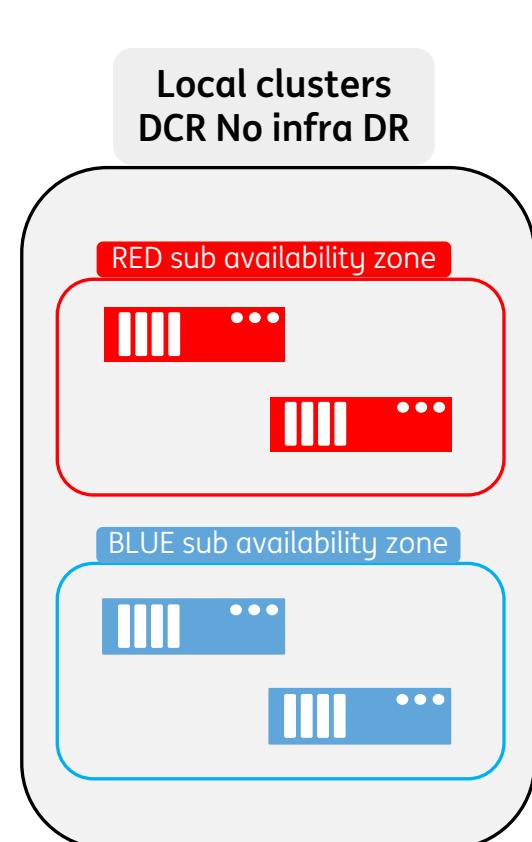
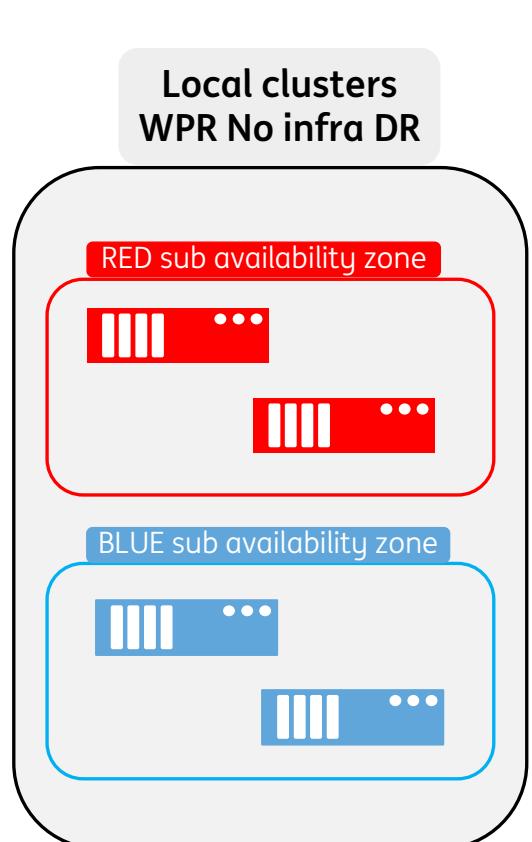


Distribute your infra across availability zones and failure domains

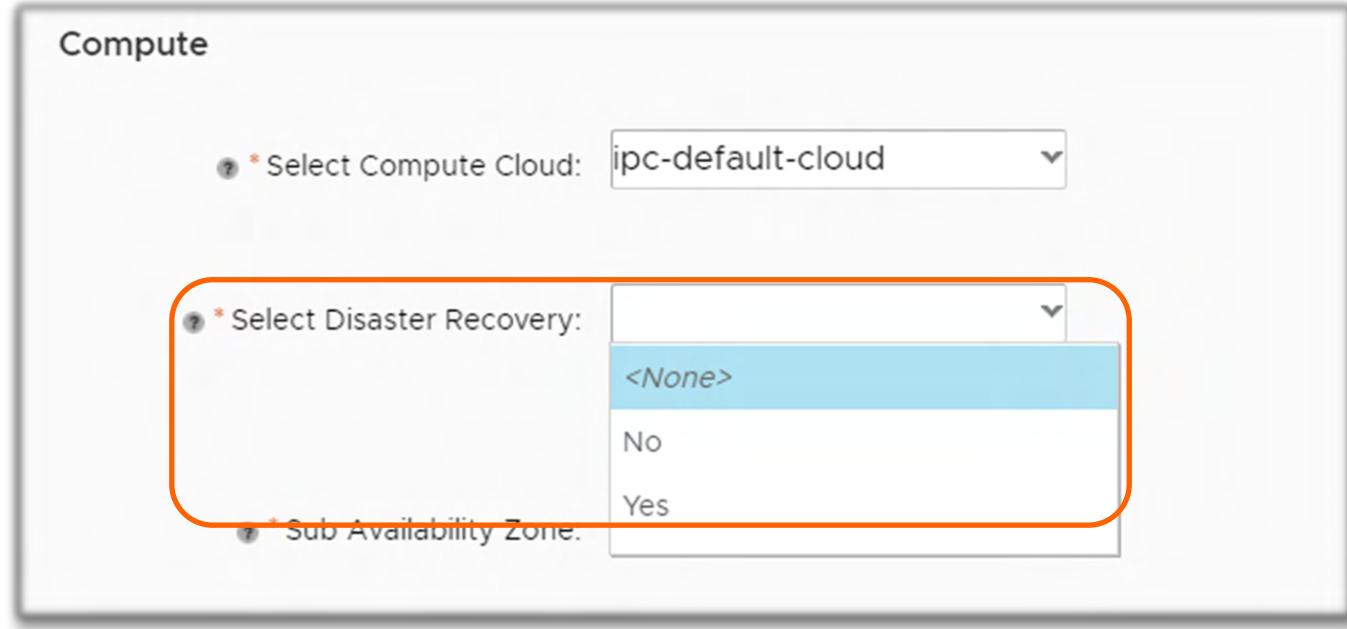
Distribute your infra across availability zones and failure domains cross datacenter



Distribute your infra using all failure domains across both datacenters

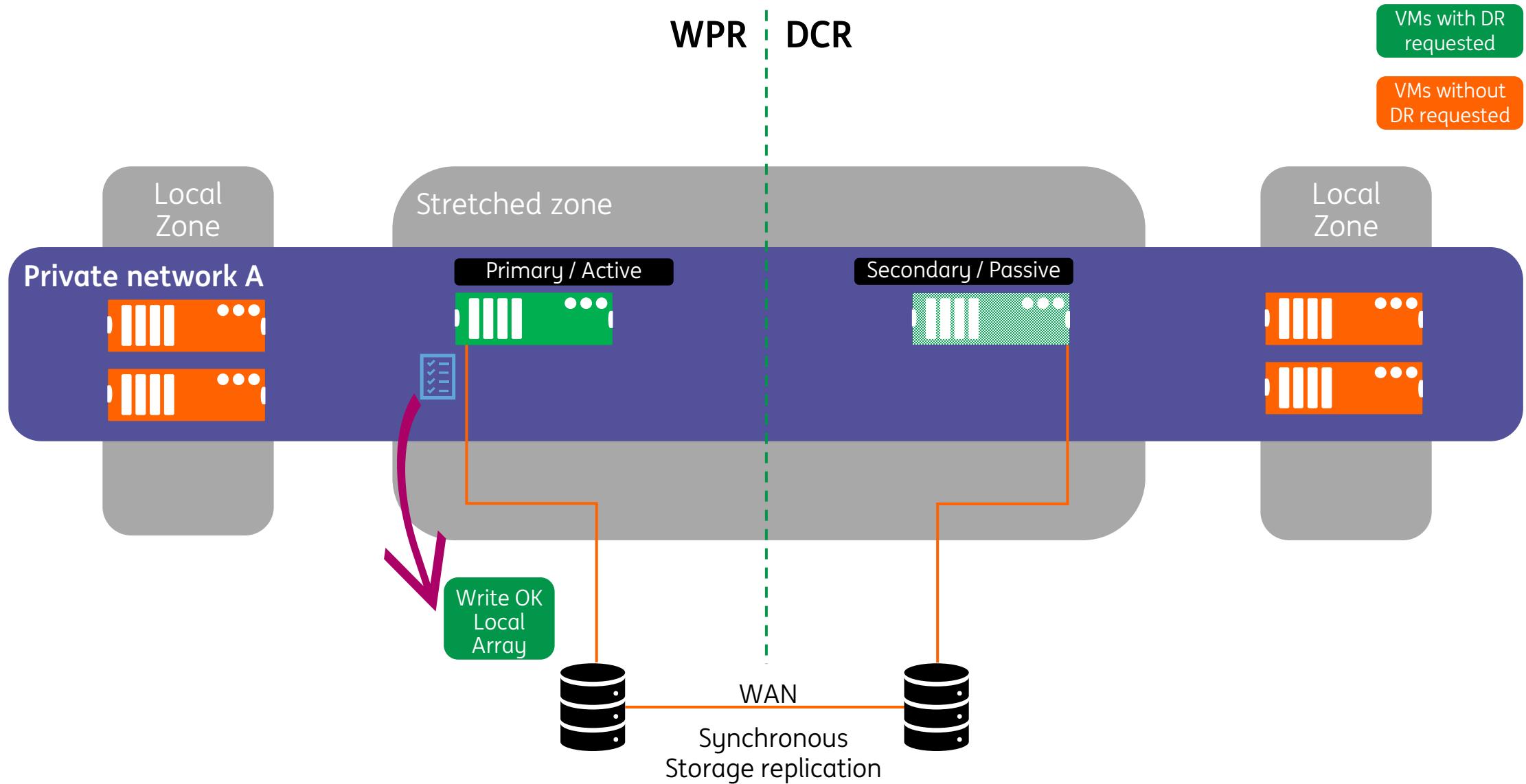


Considerations stretched vs local

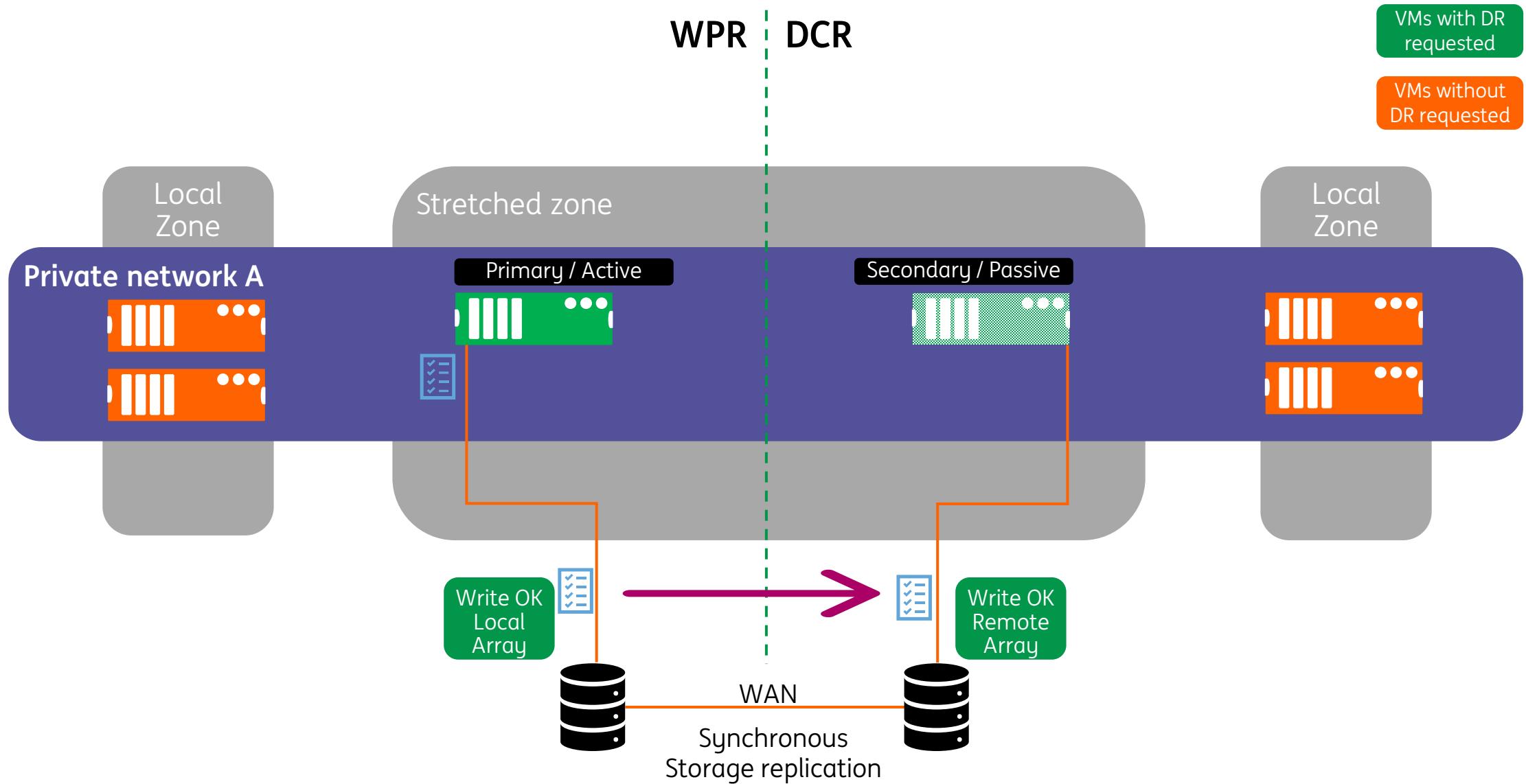


What you select in this drop-down has the most effect on the IO performance of your system.

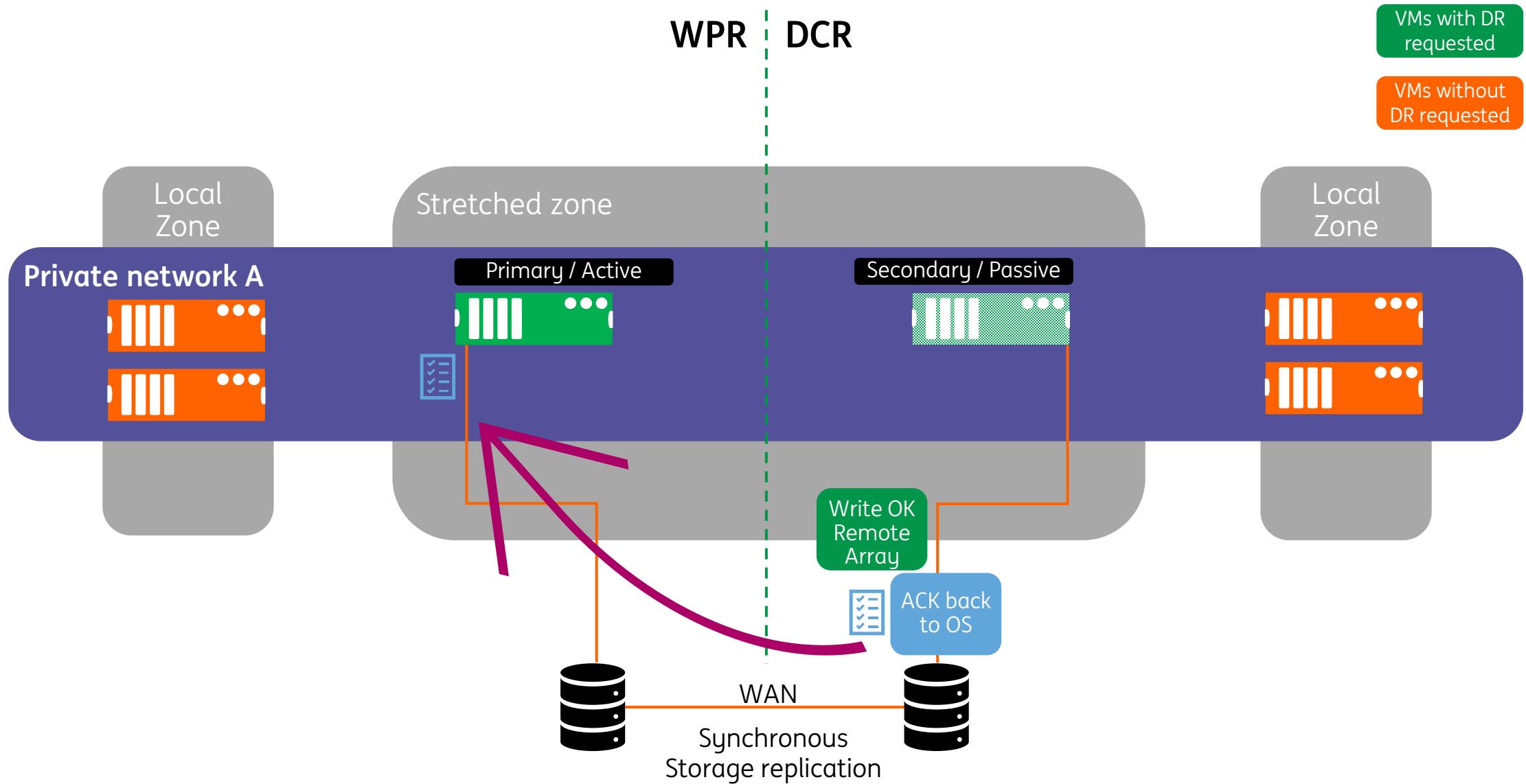
Write actions on stretched zone



Write actions on stretched zone



Write actions on stretched zone



Keyspace as a Service (KaaS)

Datastax Cassandra

DataStax Cassandra

Distributed NoSQL database

Leaderless topology

No single point of failure

Schema-less and can handle unstructured data

Developers can use a query language CQL

Onboarding process with the KaaS team (Minions Squad) is required to use KaaS.

Keyspace

Top level Database object
That controls the replication for the object at each datacenter

Contains tables, user defined types, functions and aggregates.

Typically a cluster has one Keyspace per application

A cluster can be stretched across multiple datacenters

The keyspace replication strategy and replication factor control data availability for a set of tables in each datacenter of the cluster.

Keyspace minimum config

2 datacenters per cluster

At least 5 nodes per datacenter

Network Topology Strategy
Applies the replication setting per datacenter (default 3)

3 Seed nodes per datacenter

Cluster configuration depends on the use-case where the final decision is made by Minions squad.

The application teams are responsible for data consistency in Cassandra.

Types of environments

Development and Test

Sandbox environment is open to experiment

Development is done locally on a laptop

Test is created in the IPC shared virtual infrastructure

Acceptance and Production

Is provided on dedicated infrastructure, based on Stand-Alone platform (BMaaS).

Development limitations	Test limitations
100 TPS for read	1000 TPS for read
100 TPS for write	5000 TPS for write
1GB of disk space	1GB of disk space

Configuration of your Keyspace

Cassandra has no single point of failure

Cross datacenter Active-Active configuration

For acceptance and Production environments, Cassandra knowledge is mandatory in the DevOps team

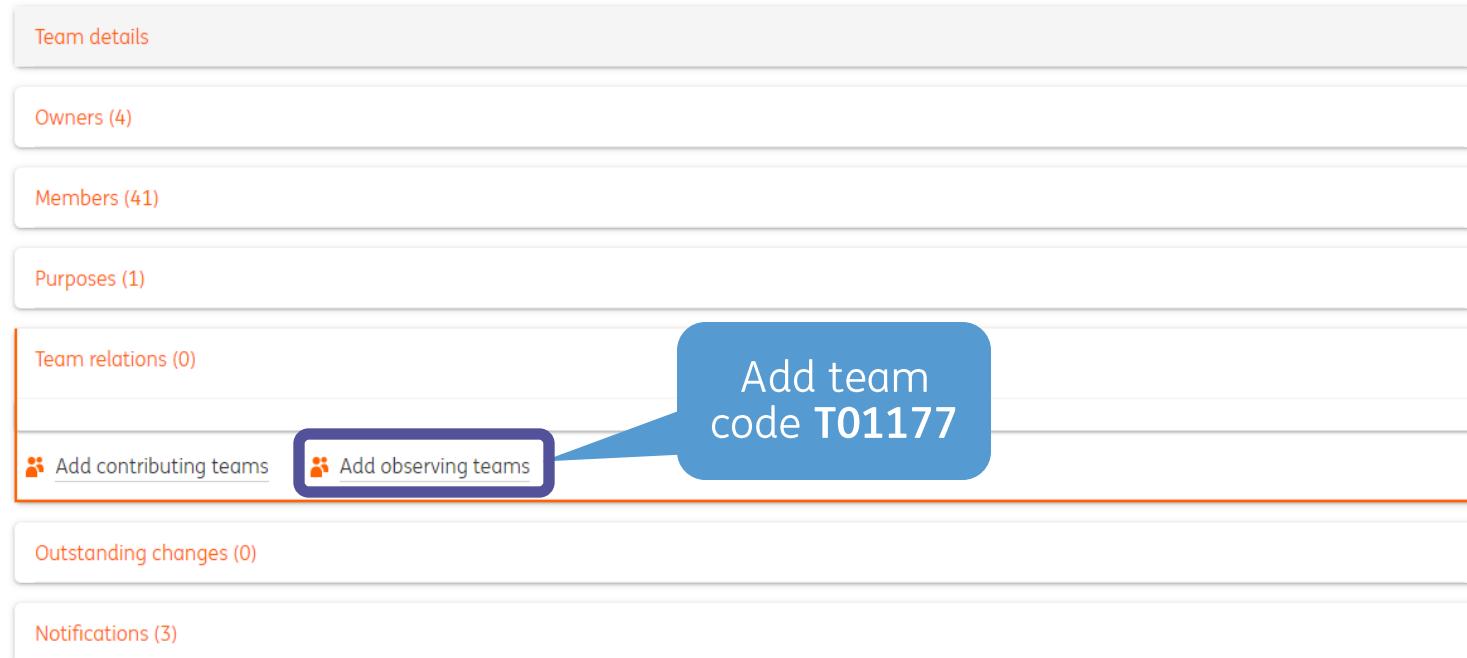
DevOps teams can change the consistency level (CL) as per application requirement

Start your knowledge journey at DataStax Academy
<https://academy.datastax.com>

Monitoring

Monitoring can be configured by sending the metrics for the DataStax driver to a team's own RTK2 Instance.

A mandatory requirement is to add the Minions squad as an observing team for your Ace team. Please add the following team code as Observer in Ace **T01177**



As part of the 'how to documentation', steps and examples are provided by The Minions squad.

Back-ups and restores

Cassandra or RDBMS

Cassandra positioned for **session data** and **cache use cases**

Traditional Relational database management systems to be used for **System of Records**

Application teams are responsible for data consistency in Cassandra

Knowledge of Cassandra
‘nice to have’ for Development & Test environments

Knowledge of Cassandra
‘must have’ for Acceptance & Production environments

Backup and restores

Cassandra can offer Snapshots based on the use-case.

No point-in-time recovery available

Keyspace users do not have control of the snapshot creation and/or restore

For Acceptance and Production auto-snapshot is in place to protect against user error

Cassandra team creates snapshots before running system upgrades

Backup use cases

Session data

Session information is typically marked as short-lived-data.

Therefore a snapshot is not required/**out of scope**.

Cache

When required, can be recreated by requesting a new feed from the source of the data.

Therefore a snapshot is not required/**out of scope**.

Persistence of derived data

When required, can be recreated by requesting a new feed from the source of the data

Therefore a snapshot is not required/**out of scope**.

System of Records

Automated snapshot creation, to be used as mitigation in case of data integrity issue.

Max. 4 days snapshot retention

**Availability, capacity, performance,
patches and Lifecycle Management responsibility**

Availability, capacity, performance, patches and Lifecycle Management responsibility

Responsibilities of the service provider

OS Patch Management
Cassandra upgrades
Code Red Patches

Ensure functional correctness of
Cluster instance

Monitor and act upon Cluster:

- availability
- capacity
- performance

Responsibilities of Service user

Ensure functional correctness of
Keyspace instance and related data

Monitor and act upon Keyspace:

- availability
- capacity
- performance

DevOps teams can participate daily with a performance test and later evaluate
the result with the KaaS team.

Disaster recovery

&

Disaster recovery exercises

Clusters in both datacenters
WPR & DCR

Built on highly redundant
infrastructure

Data will still accessible in
case of failure of one
Datacenter

**Application team
responsibility to implement
connections to both
datacenters**

Semi Annual DR test

Organized by BCM department

For Production

Weekly DR test

Organized by Minions

for Test and Acceptance

Soft DR

Native transport is turned off



Active-passive: stop serving queries on one datacenter, but keep replication cross-datacenter

Real DR

Cassandra service is turned off



Active-really down: Cassandra stop all the nodes on one datacenter

Link resources:

The Forge: <https://theforge.ing.net/product/33362/details/>

Confluence: <https://confluence.ing.net/display/mns/Minions+Squad>

Community: <https://confluence.ing.net/display/CC/Cassandra+Community+Home>

Onboarding: <https://confluence.ing.net/display/CC/Onboarding+for+a+keyspace>

Minions 'How to' main page: <https://confluence.ing.net/display/CC/How-to>

Mattermost: <https://mattermost.ing.net/cassandra-kaas/channels/town-square>

Performance test: <https://confluence.ing.net/display/CC/How-to%3A+Performance+test>

Official Datastax Driver metrics documentation: <https://docs.datastax.com/en/developer/java-driver/4.14/manual/core/metrics/>

Link resources:

Datastax Academy: <https://academy.datastax.com/>

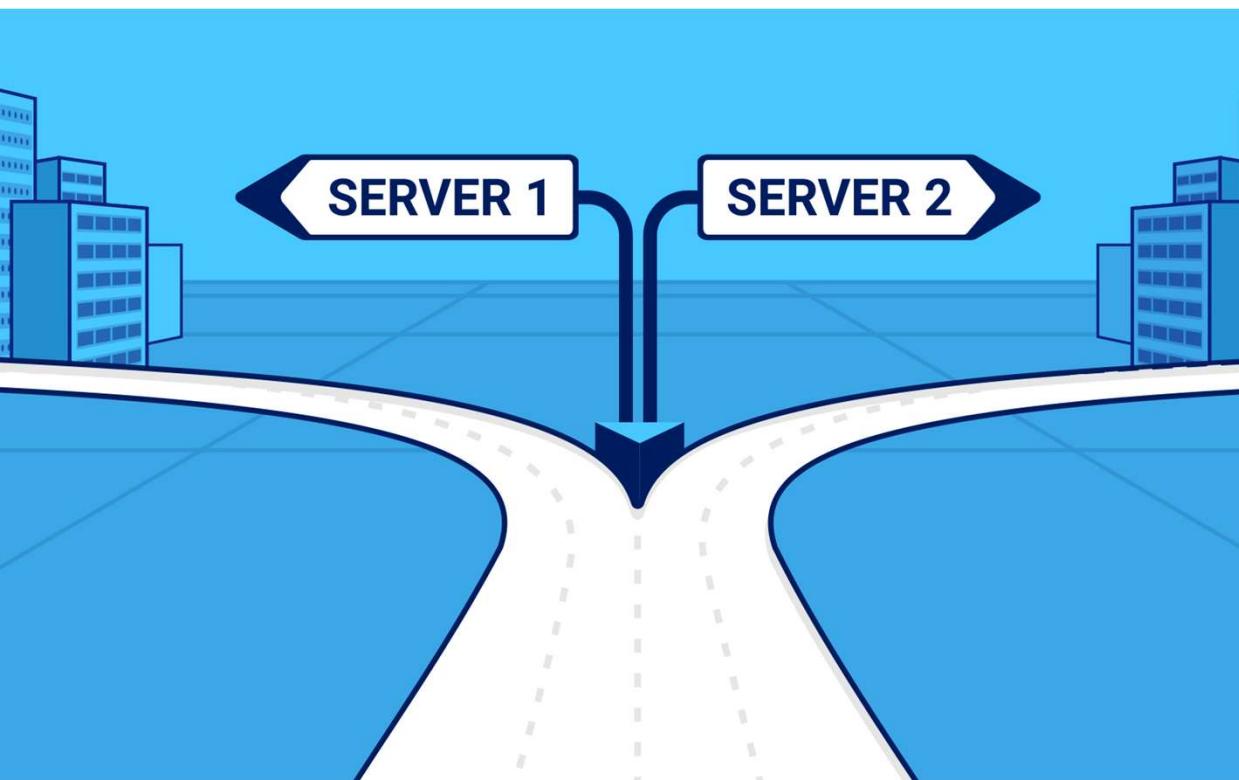
Mandatory training path to start with Cassandra to update your knowledge:

<http://www.datastax.com/dev/blog/java-driver-async-queries>

<http://www.datastax.com/dev/blog/cassandra-error-handling-done-right>

<http://www.datastax.com/dev/blog/java-driver-2-1-2-native-protocol-v3>

<http://www.datastax.com/dev/blog/ccm-2-0-datastax-enterprise-support>



Load Balancing
(LTM & GTM)
@ING

Load Balancing options

Load Balancing can be offered by:

- Software (for example NGINX)
- An appliance/hardware (for example F5).
- A software library used by the consumer/requester(for example the ***Finagle*** client) = **clientside load balancing = PREFERRED**

Why load balancing?

Availability

Scalability

NGINX at ING

We have multiple services offered by NGINX software at ING:

1. As a Security gateway/authenticating proxy/API Gateway with load balancing (internal / external / BE instances)
2. Forwarding proxy
3. Solely for Load Balancing (in IPC)

Be careful and check which NGINX you are using or want to use!

Security gateway / Authenticating proxy / API Gateway / NGINX ?



Load Balancing

A local Traffic Manager (**LTM**) is used for load balancing within one DC.

A Global Traffic Manager (**GTM**) can be used for load balancing over multiple DC's

Software solutions like the NGINX and the Finagle client can do both: load balancing **over multiple instances in multiple DC's**

A LTM and a NGINX can also be extended with security functions like **SSL Offloading**.

As non-encrypted traffic is only allowed within your networkzone it depends on your solution if SSL offloading is an option.

GTM - LTM

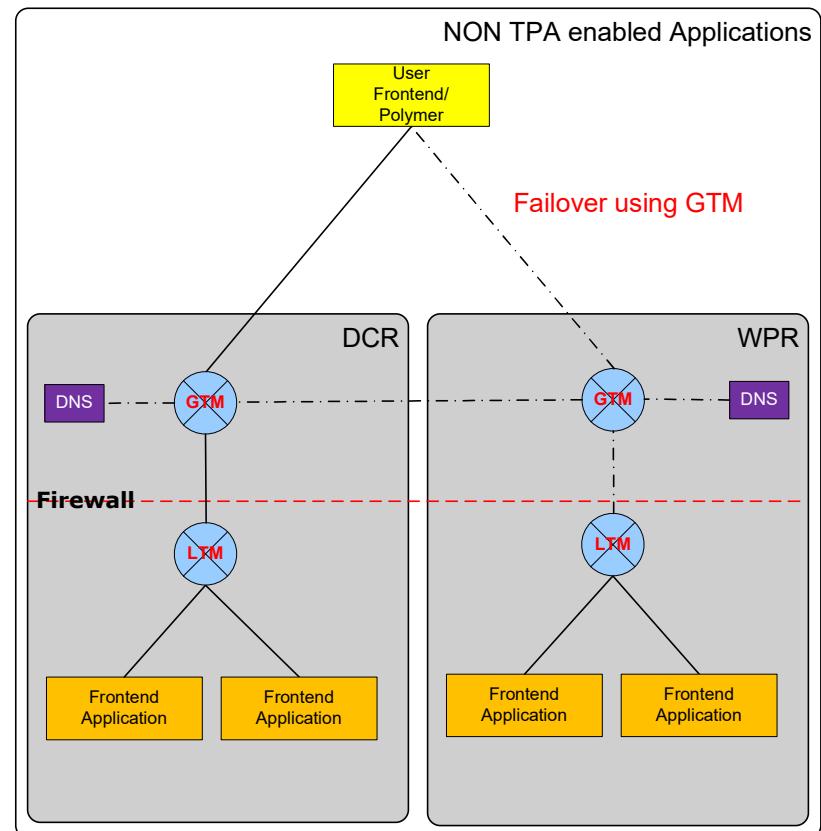
A **GTM** is an extension of **DNS**.

You can configure it **ACTIVE-PASSIVE** and **ACTIVE-ACTIVE**.

In case of a DC failure DNS will point to the remaining DC.

Failover depends on Time To Live (TTL) settings and can take several minutes or more!

The **LTM** can be your own provisioned load balancer: a NGINX in the IPC



Load balancing in IPC

In the IPC there is no generic LTM (F5) available.

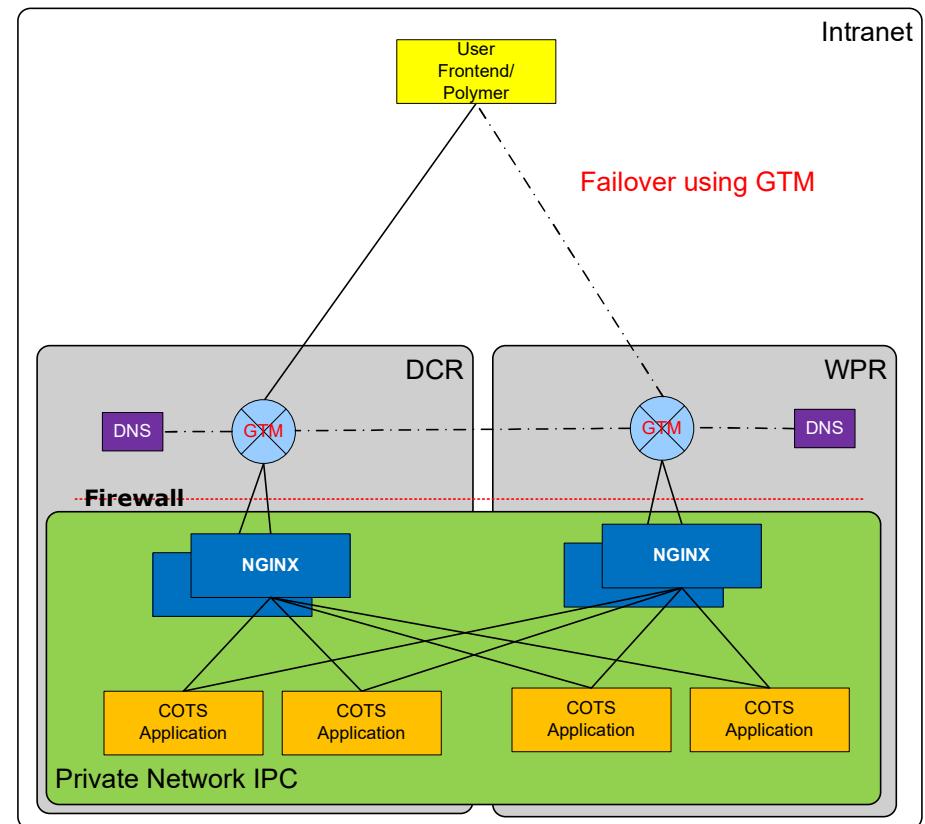
You can provision a **NGINX** in your own private network.

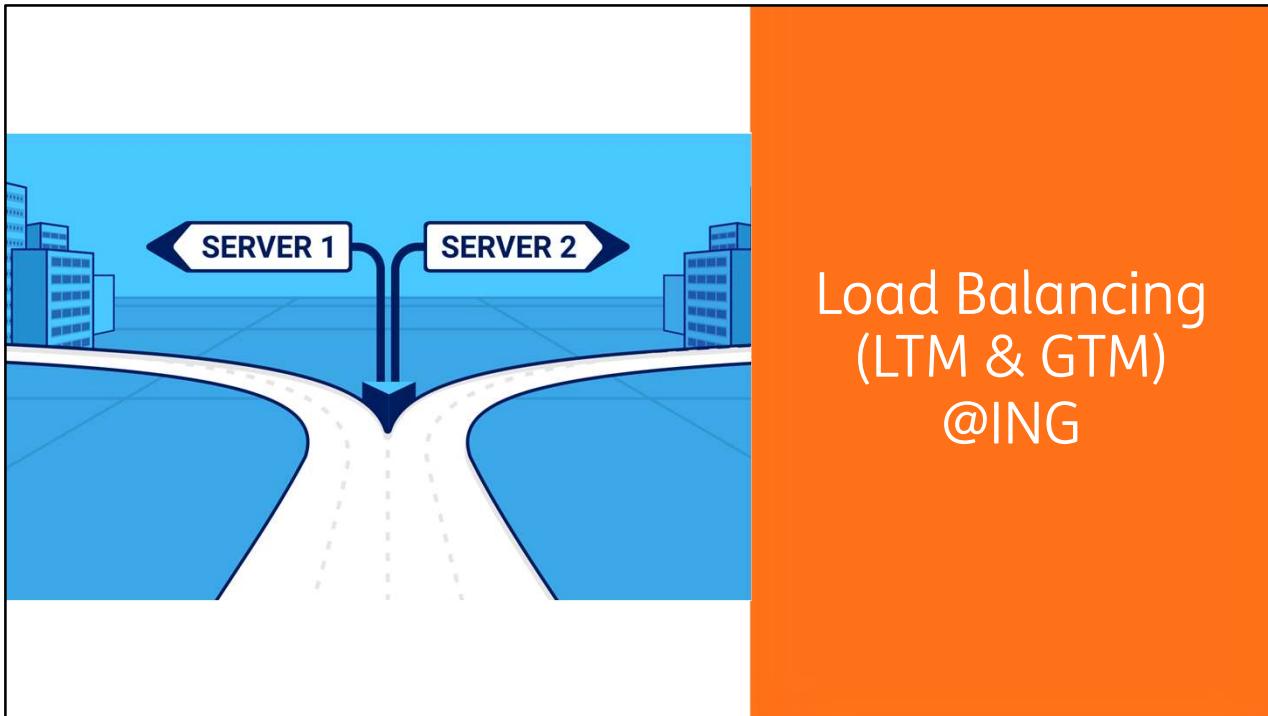
The IPC offers two NGINX services:

1. Single instance
2. High available (with keepalive)

A GTM can be used for failover over the DCs (active-passive: **not preferred**).

And as discussed before: **DR capable is not an option.**





Load Balancing
(LTM & GTM)
@ING

Clientside load balancing at ING

API's using the **Merak** framework include libraries for using the **Dynamic Service Discovery(DSD)** and **Clientside load balancing**.

TPA also offers a **Touchpoint Mesh**: you can implement a **Sidecar** next to your application which enables your application to use Dynamic Service Discovery(DSD) and Clientside load balancing (and more!). This is an implementation of a **Cloud Native Service Mesh**.

It transforms your (COTS) application to a **TPA enabled** API with the TPA security framework (using **mTLS** for example).

CNCF Cloud Native Definition v1.0

*Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. **Containers, service meshes, microservices, immutable infrastructure**, and declarative APIs exemplify this approach.*

*These techniques enable **loosely coupled** systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.*

What is a Service mesh

Service Mesh is an **inter-service communication** infrastructure

Service Mesh is a network communication infrastructure which allows you to decouple and offload the control and monitoring of internal, **service-to-service traffic (east-west)** from your microservices applications to a **sidecar**.

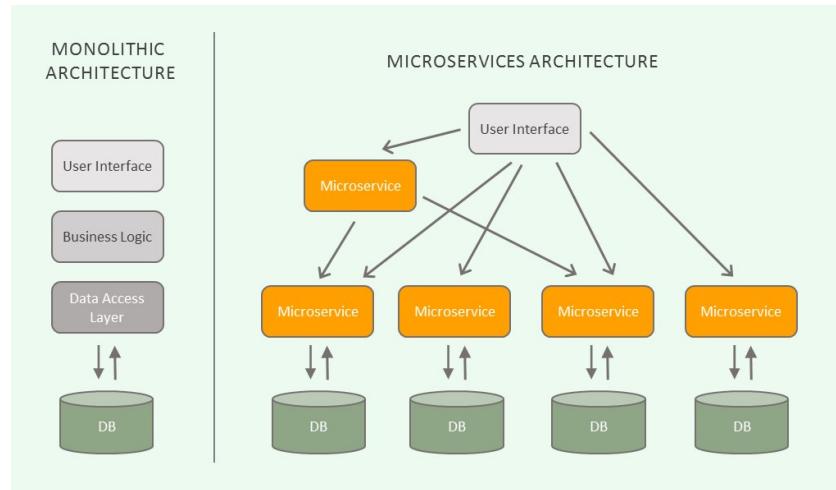
A Sidecar is used **alongside** most of the service implementations and is independent of the business functionality of the services.

4

<https://www.cncf.io/webinars/lets-untangle-the-service-mesh/>

Why is the service mesh suddenly so interesting? For many companies, tools like Docker and Kubernetes have “solved deploys”. But they haven’t solved runtime. This is where the service mesh comes in.

From Monolith to Microservices: lots of east-west traffic



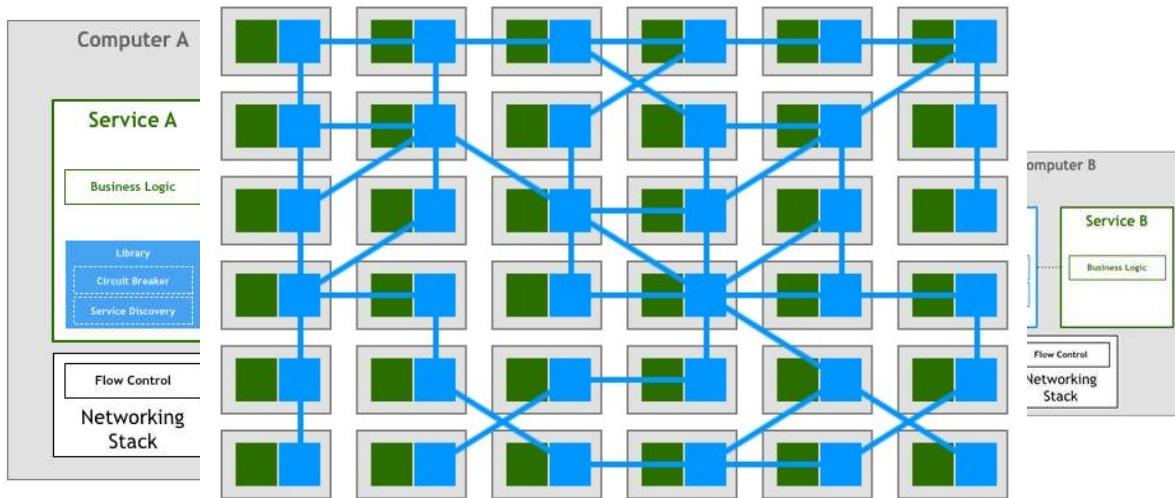
5

With the three-tiered model of application architecture (circa 2010) application traffic is first handled by a “web tier,” which in turn talks to an “app tier,” which in turn talks to a “database tier.”

So far so good, but this model starts to break down under heavy load — especially at the app layer, which can become quite large over time. Early web-scale companies — Google, Facebook, Netflix, Twitter — learned to break the monolith apart into lots of independently-running pieces, spawning the rise of microservices.

The moment microservices were introduced, east-west traffic was also introduced.

Library versus proxy



6

https://philcalcado.com/2017/08/03/pattern_service_mesh.html

Buoyant's CEO William Morgan made the observation that the interconnection between proxies form a **mesh network**. In early 2017, William wrote a definition for this platform, and called it a Service Mesh.

To control all this east-west traffic most companies responded in a similar way — they wrote **“fat client” libraries to handle request traffic**. These libraries — Stubby at Google, Hystrix at Netflix, **Finagle** at Twitter — provided a uniform way of runtime operations across all services.

But libraries are significantly less appealing in light of the operational convenience provided by out-of-process proxies

Proxies, by contrast, can be upgraded without recompiling and redeploying every application. Likewise, proxies allow for polyglot systems, in which applications are comprised of services written in different languages.

Service owners and platform engineers

In this model, developers (“**service owners**”) are blissfully unaware of the existence of the service mesh, while operators (“**platform engineers**”) are granted a new set of tools for ensuring reliability, security and visibility.

By providing APIs to analyze and operate on this traffic, the service mesh provides a **standardized** mechanism for runtime operations across the organization — including ways to ensure reliability, security and visibility.

And like any good infrastructure layer, the service mesh (ideally!) works independent of how the service was built.

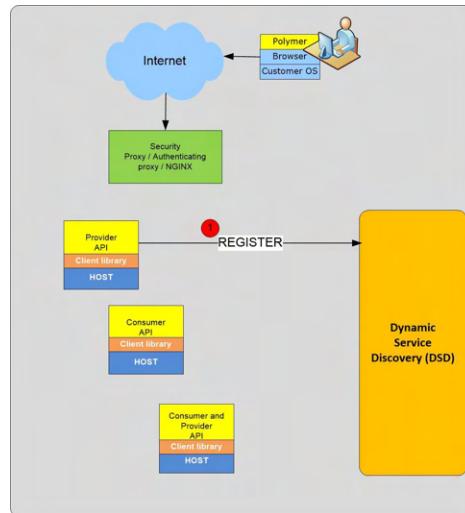
7

Implementing the service mesh in proxies rather than libraries moves responsibility for providing the functionality necessary for runtime operations out of the hands of the service owners, and into the hands of those who are the end consumers of this functionality — the **platform engineering team**.

Advantages Service mesh

- Commodity features are implemented outside microservice code and they are reusable.
- Independent deployment
- Solves most of the problems with distributed tracing, logging, security, access control etc.
- More freedom when it comes to selecting a microservices implementation language: You don't need to worry about whether a given language supports or has libraries to build network application functions.

Dynamic Service Discovery

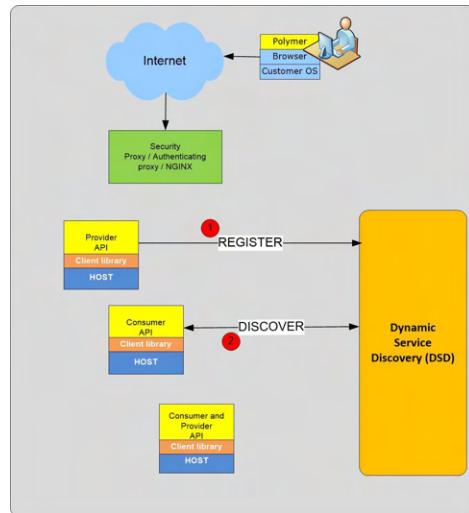


9

API's are registered and discovered by clients, i.e. gateways and consumer API's using the client library software.

Endpoints are dynamically retrieved: this is a huge advantage for changeability as you can add extra instances (scale up horizontally) without changing any configuration. Your consumers will retrieve the new endpoints from the DSD.

Dynamic Service Discovery

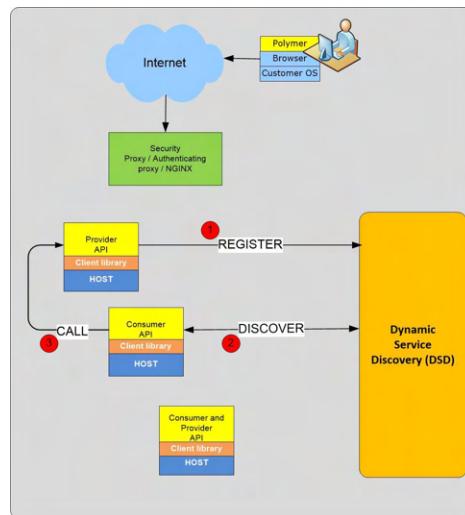


10

API's are registered and discovered by clients, i.e. gateways and consumer API's using the client library software.

Endpoints are dynamically retrieved: this is a huge advantage for changeability as you can add extra instances (scale up horizontally) without changing any configuration. Your consumers will retrieve the new endpoints from the DSD.

Dynamic Service Discovery

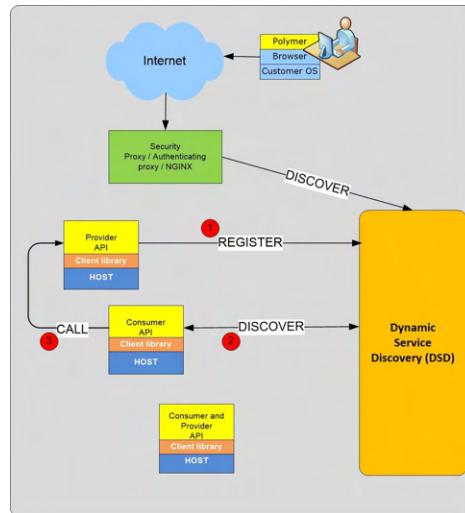


11

API's are registered and discovered by clients, i.e. gateways and consumer API's using the client library software.

Endpoints are dynamically retrieved: this is a huge advantage for changeability as you can add extra instances (scale up horizontally) without changing any configuration. Your consumers will retrieve the new endpoints from the DSD.

Dynamic Service Discovery

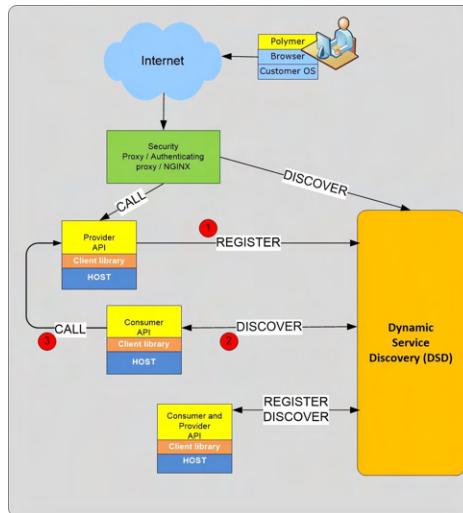


12

API's are registered and discovered by clients, i.e. gateways and consumer API's using the client library software.

Endpoints are dynamically retrieved: this is a huge advantage for changeability as you can add extra instances (scale up horizontally) without changing any configuration. Your consumers will retrieve the new endpoints from the DSD.

Dynamic Service Discovery



13

API's are registered and discovered by clients, i.e. gateways and consumer API's using the client library software.

Endpoints are dynamically retrieved: this is a huge advantage for changeability as you can add extra instances (scale up horizontally) without changing any configuration. Your consumers will retrieve the new endpoints from the DSD.

Benefits of Clientside load balancing

Squad in control

By using a library in your code disabling a pool member, configuring healthchecks or changing any configuration is in control of the squad. You don't have to request your infra provider to change the load balancer.

Dynamic routing

By using the **Dynamic Service Discovery** (next slides) locations are retrieved dynamically: no configuration change for consumers/requesters (or in load balancer middleware) when locations of your instances change.

No dependency on infra

A migration to a different infra provider is easier. A change/LCM of middleware (NGINX/F5) does NOT affect your application.

14

Load balancing for COTS applications

The Touchpoint service mesh/Sidecar is specifically built for COTS applications.

With Touchpoint service mesh/sidecar Clientside load balancing can be used for your COTS applications. **This is preferred.**

An access token (TPA security framework) is not mandatory for your consumers/requesters, however **mTLS** is.

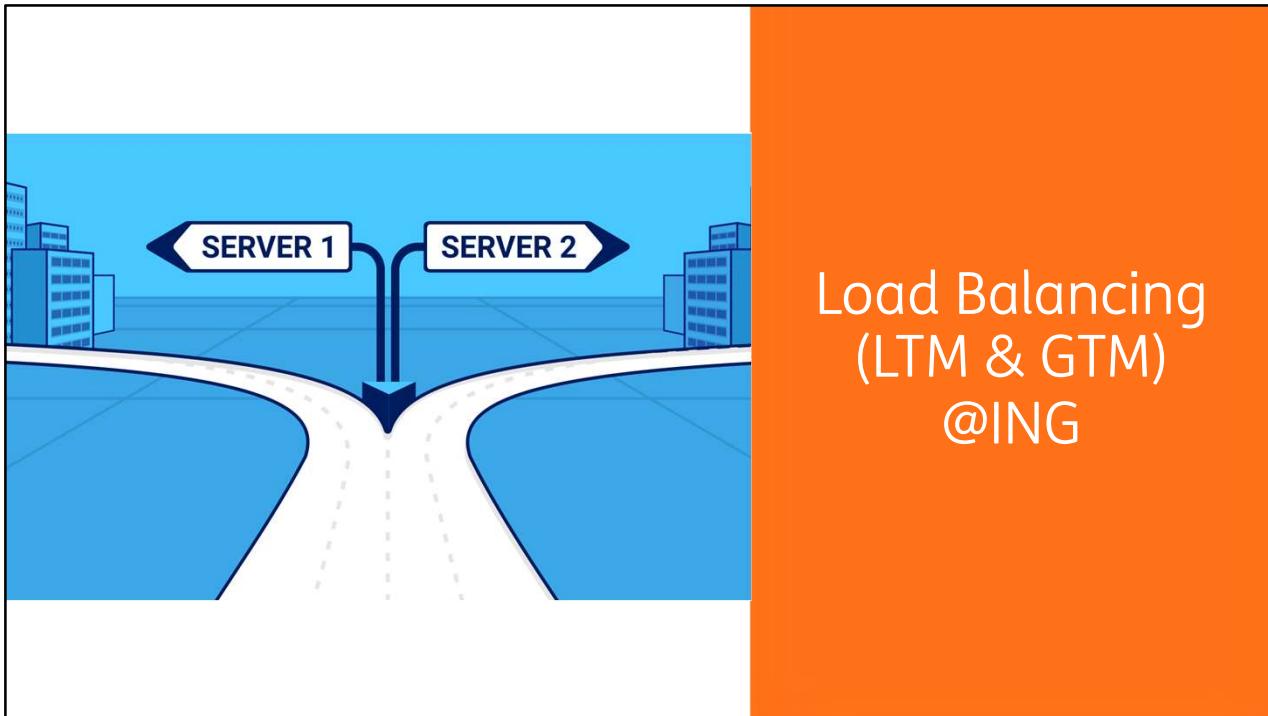
Touchpoint service mesh/sidecar offers a lot more than Clientside load balancing: mTLS, TracING (for observability!) and in the near future firewall automation for your consumers/requesters.

Please consult your Solution Architect or BTA if the Sidecar is **not** an option.

From COTS to TPA enabled!

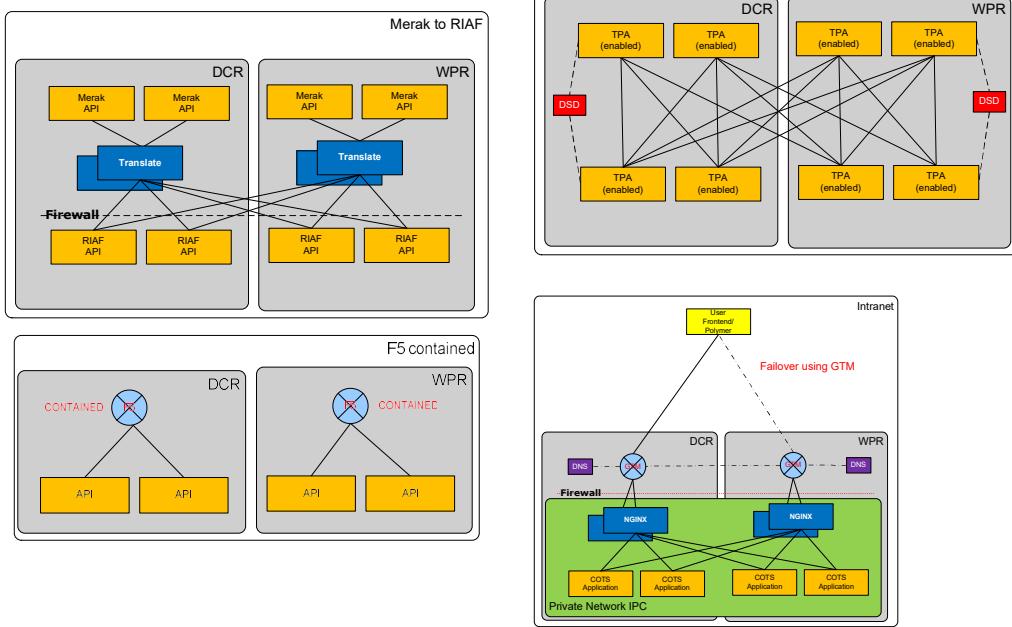


16



Load Balancing
(LTM & GTM)
@ING

Complex ?



VIP = Virtual IP address

A F5 Load Balancer presents a "virtual IP address" to the outside world, and when users attempted to connect, they would forward the connection to the most appropriate real server doing bi-directional network address translation (NAT). NAT is explained in the last session.

Without a DSD a static list of endpoints is used in the Load Balancer or Application (or NGINX as we will see later), meaning the Load Balancer or Application must be modified to change the routing (when you add extra instances/machines for example).

Key takeaways

Use Clientside Load Balancing

Use DSD for dynamic routing

Be smart when mentioning a NGINX...

Use the Touchpoint mesh / sidecar for COTS and non TPA APIs

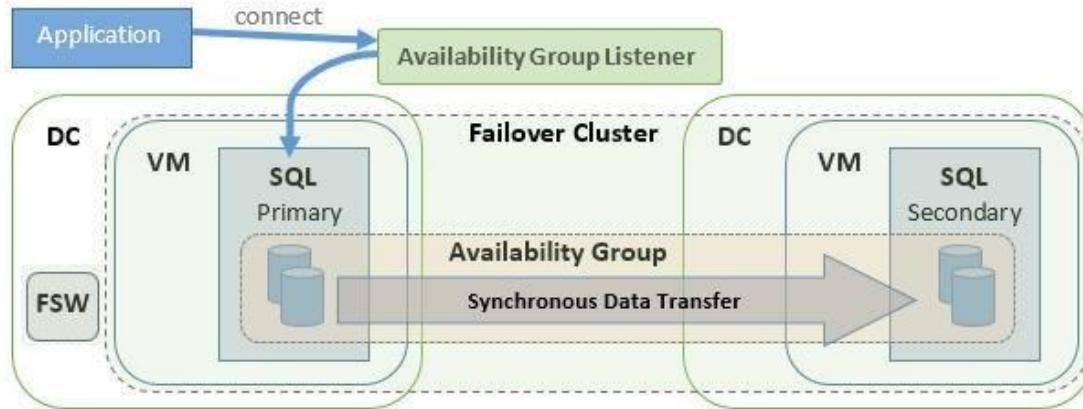
A GTM is active – passive, failover depends on TTL: not preferred

F5 is not allowed, use your OWN NGINX when necessary

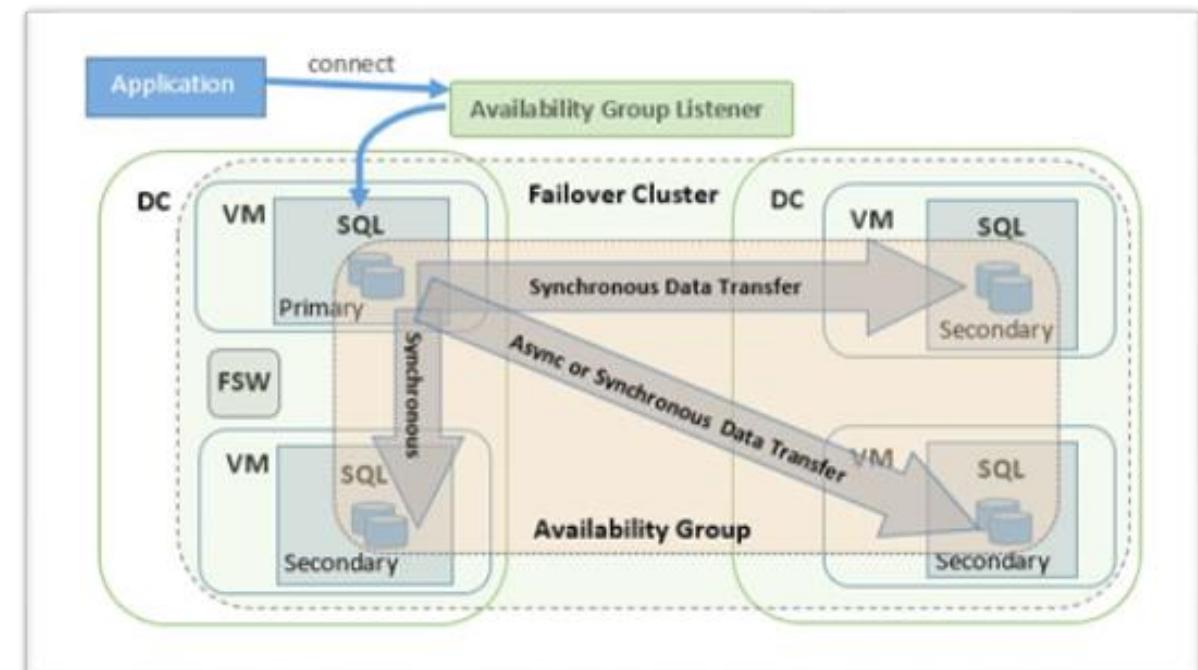
Setting up your SQL environment

2 node versus 4 node

2 node cluster



4 node cluster



Capacity management

Alerting

- Database filling: SCOM monitors database filling → alert → ServiceNow
 - SSMS: Standard Report: Disk usage report, 20% free is threshold for SCOM
 - SSMS: `sp_ing_show_database_usage`: Overview
 - SSMS: `sp_ing_resize_database`: Enlarge or Shrink database(files)
- Drive full: SCOM monitors drive/mountpoint filling → alert → ServiceNow
 - SSMS: `sp_ing_DatabaseVolumeStats` 20% free is threshold for SCOM

Check current disk usage on VM

```
exec sp_ung_show_disk_usage
```

100 % <

	VolumeMountPoint	DiskIdOnOS	VolumeLabel	Description	TotalMBs	AvailableMBs	PctFree	AllocUnitSizeKB
1	J:\	2	DATA		11230.00	5570.56	49.60	4
2	J:\SQL2012\MSSQL11.MSSQLSERVER\MSSQL\TEMPDB_DATA\	3	TEMPDB_DATA		6109.94	42.25	0.69	64
3	J:\SQL2012\MSSQL11.MSSQLSERVER\MSSQL\USERDB_DATA\	4	USERDB_DATA		26493.94	25401.44	95.88	64
4	K:\	5	TRANLOG		2015.87	1968.96	97.67	4
5	K:\SQL2012\MSSQL11.MSSQLSERVER\MSSQL\TEMPDB_TRANL...	6	TEMPDB_TRANLOG		3039.81	87.44	2.88	64
6	K:\SQL2012\MSSQL11.MSSQLSERVER\MSSQL\USERDB_TRANL...	7	USERDB_TRANLOG		14301.94	12244.75	85.62	64
7	P:\	9	APPLICATIONDATA	Including \\SJ4000001107\AppSupp\$	6110.00	5939.59	97.21	4
8	U:\	10	BACKUP	Including \\SJ4000001107\SQLADHOCBACKUP\$	3039.87	479.46	15.77	4

NOTE

1 The PctFree value of volumes TEMPDB_DATA and TEMPDB_TRANLOG are nearly 0 (zero). This is by design. Disks that are not accessible are not reported.

Order disk extension

IPC storage and volume overview

ID	Capacity (GB)	Drive Letter / Mount Path	Label	Device ID	Storage Path
0	120	C	System	SCSI (0:0)	DC-DCR-R-WIN-12-005
1	5	D	Logs	SCSI (0:1)	DC-DCR-R-WIN-12-005
2	10	J	Data	SCSI (0:2)	DC-DCR-R-WIN-12-005
3	5	J01	Tempdb_Data	SCSI (0:3)	DC-DCR-R-WIN-12-005
4	26	J02	Userdb_Data	SCSI (0:4)	DC-DCR-R-WIN-12-005
5	5	K	TranLog	SCSI (0:5)	DC-DCR-R-WIN-12-005
6	3	K01	TempDB_Tranlog	SCSI (0:6)	DC-DCR-R-WIN-12-005
7	14	K02	Userdb_Tranlog	SCSI (0:8)	DC-DCR-R-WIN-12-005
8	5	P	ApplicationData	SCSI (0:9)	DC-DCR-R-WIN-12-005
9	3	U	Backup	SCSI (0:10)	DC-DCR-R-WIN-12-005

196 GB in use

Disk information

Disks overview: [New](#) [Edit](#) | [Delete](#)

diskIndex	capacityInGB	label
0	120	System
1	5	Logs
2	10	Data
3	5	Tempdb_Data
4	120	Userdb_Data
5	5	TranLog
6	3	TempDB_Tranlog
7	14	Userdb_Tranlog
8	5	ApplicationData

Select disk to update:

Disk label: Userdb_Data

* Disk size (GB): (Select 26-6144)

Validation result:

Performance monitoring

ServiceNow

* Name	[REDACTED]		* Config Admin Group	Tech/Infra/Cloud/IaaS/Microsoft Services
* Status	Installed		* Company	ING
: Operational status	Operational		Cost center	[REDACTED]
Environment	Production		Business Unit	
CI Alias	[REDACTED]		* IT Custodian	[REDACTED]
* Asset Owner	[REDACTED]	<input type="button" value="Search"/>	Tech PL Environment List	
ING Category	Windows			
ING Identifier	[REDACTED]			
Entity	ING Bank Netherlands - Bank	<input type="button" value="Search"/>		
Very	Monitoring	Relation	History	
Monitored	<input checked="" type="checkbox"/>	checked!	Choose GMCR!	Monitored By
Accepted	<input checked="" type="checkbox"/>	checked!		Accepted By
Standby	<input type="checkbox"/>		Vulnerability Scanning	Thorough
Is Event 2 Console	<input type="checkbox"/>		Technical State Compliance Monitoring	Yes
Auto-Ticketing Events	<input type="checkbox"/>	unchecked!		

Check Disable Auto-Ticketing Events

Detail*H.Node		Discovery	Monitoring	Relation	History
Monitored	<input checked="" type="checkbox"/>			Monitored By	<input type="text" value="GMCR"/>
Accepted	<input checked="" type="checkbox"/>			Accepted By	<input type="text"/>
Standby	<input type="checkbox"/>			Vulnerability Scanning	<input type="text" value="Regular"/>
Is Event 2 Console	<input type="checkbox"/>			Technical State Compliance Monitoring	<input type="text" value="Yes"/>
Disable Auto-Ticketing Events		<input checked="" type="checkbox"/>			

Onboard on IAT

1 Choose Your Application 2 Choose Your Team 3 Team Details Done

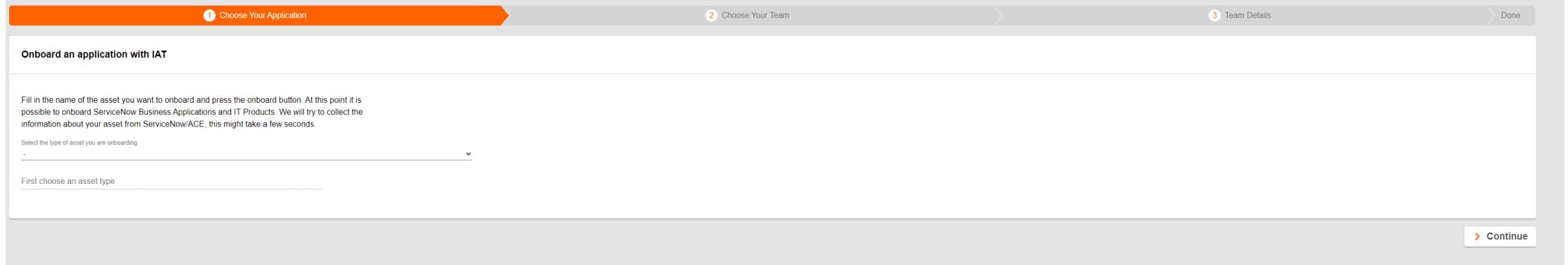
Onboard an application with IAT

Fill in the name of the asset you want to onboard and press the onboard button. At this point it is possible to onboard ServiceNow Business Applications and IT Products. We will try to collect the information about your asset from ServiceNow/ACE, this might take a few seconds.

Select the type of asset you are onboarding

First choose an asset type

> Continue



Find the latest information on the forge

ING Home Marketplace News Journeys Search Login

Version created: 03.08.2023 Last verification: 03.08.2023

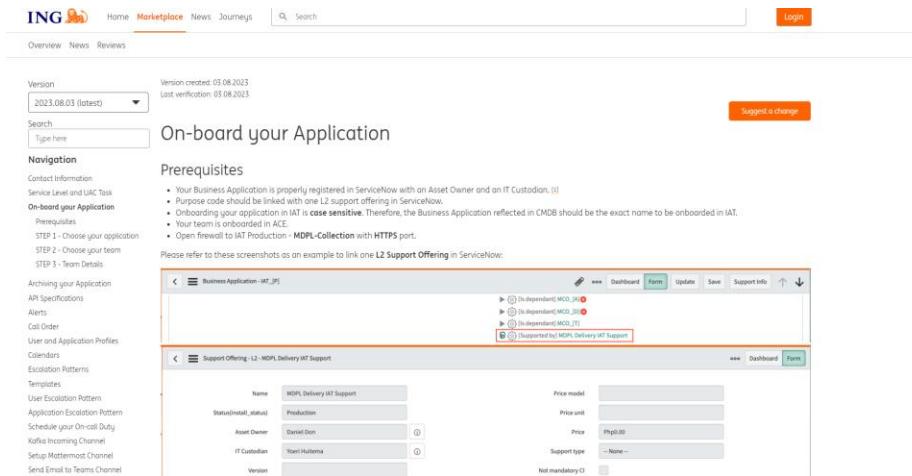
Suggest a change

On-board your Application

Prerequisites

- Your application is properly registered in ServiceNow with an Asset Owner and an IT Custodian.
- Support code should be linked with one L2 support offering in ServiceNow.
- Onboarding your application in IAT is case sensitive. Therefore, the Business Application reflected in CMDB should be the exact name to be onboarded in IAT.
- Open firewall to IAT Production - MDPL-Collection with HTTPS port.

Please refer to these screenshots as an example to link one L2 Support Offering in ServiceNow:



SquaredUp sample

All SCOM Alerts DRAFT

hour | 12 hours | 24 hours | week | all

SQL 2008 DB Average Wait Time is too high	MSSQLSERVER SPNLAPP74061.europe.intranet	a minute ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SDNLAPP72249.europe.intranet	2 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SANLAPP79018.europe.intranet	2 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER spnlapp73039.europe.intranet	2 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SANLAPP79046.europe.intranet	2 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	3 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	3 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SDNLAPP72005.europe.intranet	4 minutes ago
SQL 2008 DB Average Wait Time is too high	MSSQLSERVER SPNLMTG74026.europe.intranet	6 minutes ago
SQL 2012 DB Average Wait Time is too high	MSSQLSERVER SJ4000000277.ad.ing.net	6 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SDNLAPP73034.europe.intranet	7 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	10 minutes ago
SQL 2008 DB Average Wait Time is too high	MSSQLSERVER STNLAPP73005.europe.intranet	11 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	11 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	11 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	13 minutes ago
Operations Manager failed to start a process	Health Service SPNLAPP78003.europe.intranet	13 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SPNLAPP72203.europe.intranet	13 minutes ago
IIS 8 Application Pool is unavailable	TeamCentral SJ4000000643.ad.ing.net	13 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	18 minutes ago
SQL 2008 DB Average Wait Time is too high	MSSQLSERVER SPNLMTG75072.europe.intranet	19 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	19 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	19 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	20 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	21 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	21 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	21 minutes ago
SQL 2008 DB Average Wait Time is too high	MSSQLSERVER SPNLAPP78049.europe.intranet	24 minutes ago
SQL 2012 DB Average Wait Time is too high	MSSQLSERVER SPNLMTG74006.europe.intranet	30 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SPNLAPP72226.europe.intranet	31 minutes ago

quick find

edit

Health System Center suite

ING

Operations Manager

Configuration Manager

Orchestrator

Back-ups and restore

Restore single database to the same database to point in time (with overwrite).

Backup - Backup/Restore On-Demand Operations - sx9320014320

Service Description: <https://confluence.ing.net/x-wiuNQ>

Select On-Demand Restore to perform a restore.

* Available Operation: On-Demand Restore On-Demand Backup

Select Database Restore to perform a restore of database(s).

* On-Demand Restore Options: Database Restore

Select No as we are restoring a single database

* Perform Multiple DB Restore?: No

Select Yes as we are restoring to same database.

* Restore to same DB?: Yes

Select Point In Time as we want to restore to a specific timestamp.

* Restore Options: Point In Time Available Backups

Enter the date and time which is intended for PIT restore
Format: YYYY-MM-DDTHH:mm:ss

Note: It is recommended to check last successful backup before entering date/time. This can be done by selecting Available Backups at Restore Options on this form, or by running backup reports.

* Enter Restore Point in Time: 2023-07-03T12:00:00

PitLink: https://ing.sharepoint.com/sites/msportal_cs/SQL/SitePages/Commvault_Info.aspx

** Date Format: YYYY-MM-DDTHH:MM:SS Eg: 2021-02-27T19:04:23 Restore PIT is only possible to time <= latest transaction-log-backup

Check the checkbox to verify if any backup can be found.

* Select to verify Restore Point in Time:

The results of checking if any backup can be found will be shown in this textarea.

Restore Validation Results: OK

Select the database for which restore needs to be performed.

* Select Database To Restore: msdb

** Database will be overwritten to selected Date and Time!!

Give confirmation for database overwrite warning.

Do you want to Continue?

Validation: OK

Enter a valid reason for the restore, e.g. an incident number.

* Reason: Incident: INC0000001

Click the Submit button to initiate the restore.

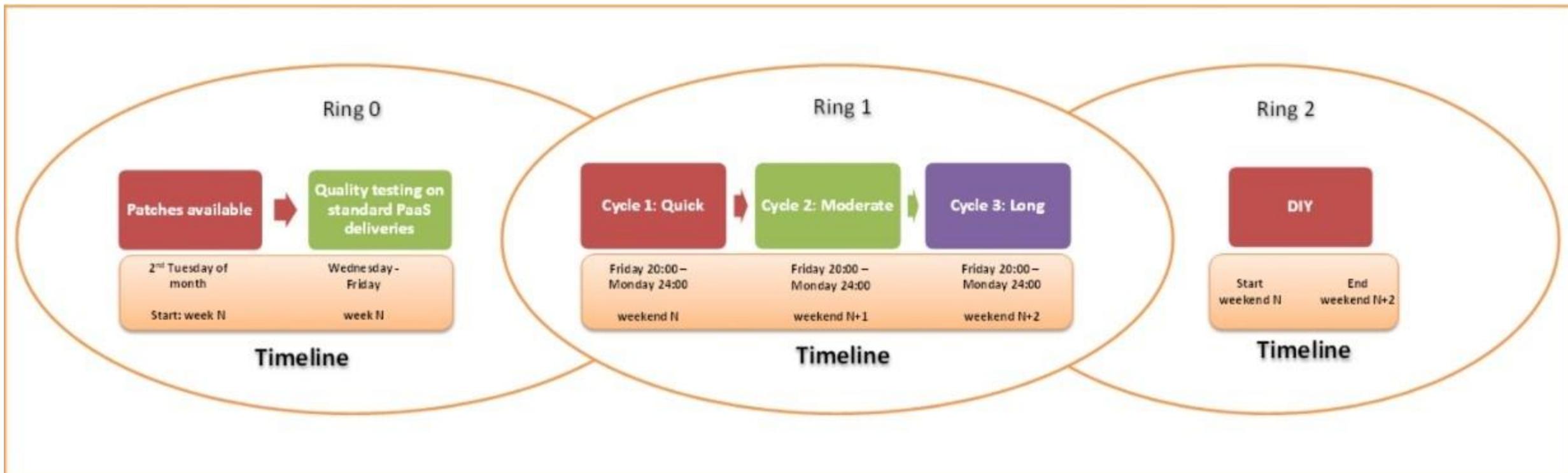
SUBMIT **CANCEL**

Database restore options

- 1 Restoring a single database to the same database by choosing 'Point in time' (overwrites data)
- 2 Restoring a single database to the same database by choosing 'available backups' (overwrites data)
- 3 Restoring a single database to an alternate database by choosing 'Point in time'
- 4 Restoring a single database to an alternate database by choosing 'available backups'.
- 5 Restoring multiple databases to the same databases by choosing 'Point in time' (overwrites data).
- 6 Restoring multiple databases to the same databases by choosing 'available backups' (overwrites data)
- 7 Restoring multiple databases to alternate databases by choosing 'point in time'.
- 8 Restoring multiple databases to alternate databases by choosing 'available backups'.

Patches and change strategy using IPC tools

Ring model



View patch window in SCOM

The screenshot shows the System Center Operations Manager (SCOM) web interface. At the top, there is a navigation bar with links like Alerts, BMG, CIO, DAS, GS, INFRA NL, TECH, WBS, BE CoreBank, Default, Teams, Getting Started, Overview, Overview, NOC, Applications, Microsoft Services, Applications, a search bar, and a menu icon.

The main content area displays the status of an SCCM agent named SX9000074681.ad.ing.net. The page includes tabs for IIS WEB SERVER, WINDOWS SERVER (2016 OR 2019), WINDOWS SERVER (2016), MEMORY, NETWORK, DISKS, MONITORED ENTITY, PERFORMANCE, ALERTS, VADA, CUSTOM, SERVICENOW, and SCCM AGENT. The SCCM AGENT tab is highlighted with a yellow box.

Status of SCCM agent

Server Name	SCCM Agent Activity Status	Last Active Time	Last Health Evaluation	Last Evaluation Healthy	Last Health Evaluation Result
SX9000074681	Active	August 2nd 2022, 1:01:05 pm	February 18th 2022, 9:00:28 pm	Evaluation Succeeded	Pass

Health Status

SX9000074681
Healthy
Last Health Evaluation: February 18th 2022, 9:00:28 pm

Active Status

SX9000074681
Healthy
Last Active Status: August 2nd 2022, 1:01:05 pm

Additional informations

If you have experienced any issues with your SCCM agent installed on IPC host, please follow up this Confluence article: click [HERE](#)
Patching process and maintenance windows are described here: click [HERE](#)

Patching Maintenance Window

Server Name	Maintenance Window
SX9000074681	RING R1C2W10 Saturday N+1 00:00 to 06:00

version 5.4.1.8070 | SCOM Edition | © copyright 2022 Squared Up Ltd.

Disaster recovery scenarios

Scenarios

2 node cluster

Situation	Result
FSW down	AlwaysOn cluster stays up, Listener stays available, failover possible
Primary server down	AlwaysOn cluster stays up, Listener stays available, failover is automatically executed.
Primary and FSW down	AlwaysOn cluster goes down as majority of servers are lost. Listener is unavailable. Secondary SQL Server is running and is connectable, directly, not through Listener
Secondary and FSW down	AlwaysOn cluster goes down as majority of servers are lost. Listener is unavailable. Primary SQL Server is running and is connectable, directly, not through Listener

4 node cluster

Situation	Result
FSW down	AlwaysOn cluster stays up, Listener stays available, failover possible
Primary server down	AlwaysOn cluster stays up, Listener stays available, failover is automatically executed.
Secondary server down	AlwaysOn cluster stays up, Listener stays available, no failover possible (but also not necessary)
Primary, secondary and FSW down (same datacenter)	AlwaysOn cluster goes down as majority of servers are lost. Listener is unavailable. Secondary SQL Servers are running and connectable, directly, not through Listener
2 Secondaries and FSW down (same datacenter)	AlwaysOn cluster goes down as majority of servers are lost. Listener is unavailable. Primary SQL Server and one secondary are running and connectable, directly, not through Listener



OVERLOAD

Overload
prevention

Shared platform

Resources and services are shared.

Shared Hardware /
network

Virtualization /
multitenant

Shared
platforms

DBaaS, KaaS, ICHP, MDPL,
etc.

Shared services

Having multiple
consumers

The point is when your service encounters an overloading issue (DDOS, retry storm, sudden data increase, excessive lookup queries) the impact, or blast radius, can be enormous.

Overload prevention: Throttling

With throttling you control the consumption of resources used by your service.

This can allow your service to continue to function and meet service level agreements, even when an increase in demand places an **extreme load** on resources.

Dynamic configuration change of throttling behavior at runtime is desirable. If a system faces an abnormal load throttling limits might need to increase or decrease to stabilize the system and keep up with the current traffic.

A service may throttle based on **different metrics over time**, such as:

- The number of concurrent requests per second.
- The amount of data (for example, 2 GiB per minute).

Examples **throttling strategies**:

- Rejecting requests from a consumer who's already accessed your API more than n times per second over a given period of time.
- Disabling or degrading the functionality of selected nonessential services



3

<https://learn.microsoft.com/en-us/azure/architecture/patterns/throttling>

Overload prevention: Rate limiting

Rate limiting involves setting a maximum limit on the number of API requests that can be made within a specific time frame.

Limits can be set in a contract (like a *Rate Limiting SLA*) or defined in a rate limiting policy.

Rate Limiting is a security measure meant to protect a service from malicious or excessive usage by limiting the number of requests over a given period of time.

The most common rate-limiting technique is the Token Bucket Algorithm.

You can use a rate limiting pattern to help you avoid or minimize throttling errors related to throttling limits.



4

Overload prevention: data sources

Be aware of:

- a (sudden) increase of the size of a data source
- Excessive lookup queries

On a shared platform like DBaaS or KaaS this may impact the performance and QoS of multiple services.

Set a rate limit on the number of requests or the amount of data that a user or consumer can request within an interval of time.

For example, set a maximum for the no. of rows a user or consumer can fetch.



5

Overload prevention: risks

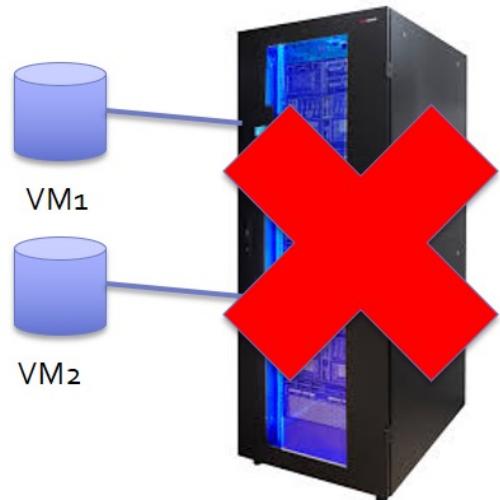
Implementing overload mechanisms are **not without risk**: it can have a direct impact on our customer or on your consumer.

For example, a customer gets an error message: "Please try again later". Or a consuming application is not able to retrieve data.

Implementing overload mechanisms requires a thorough investigation, ceiling tests and coordination in a service chain.



Rack A

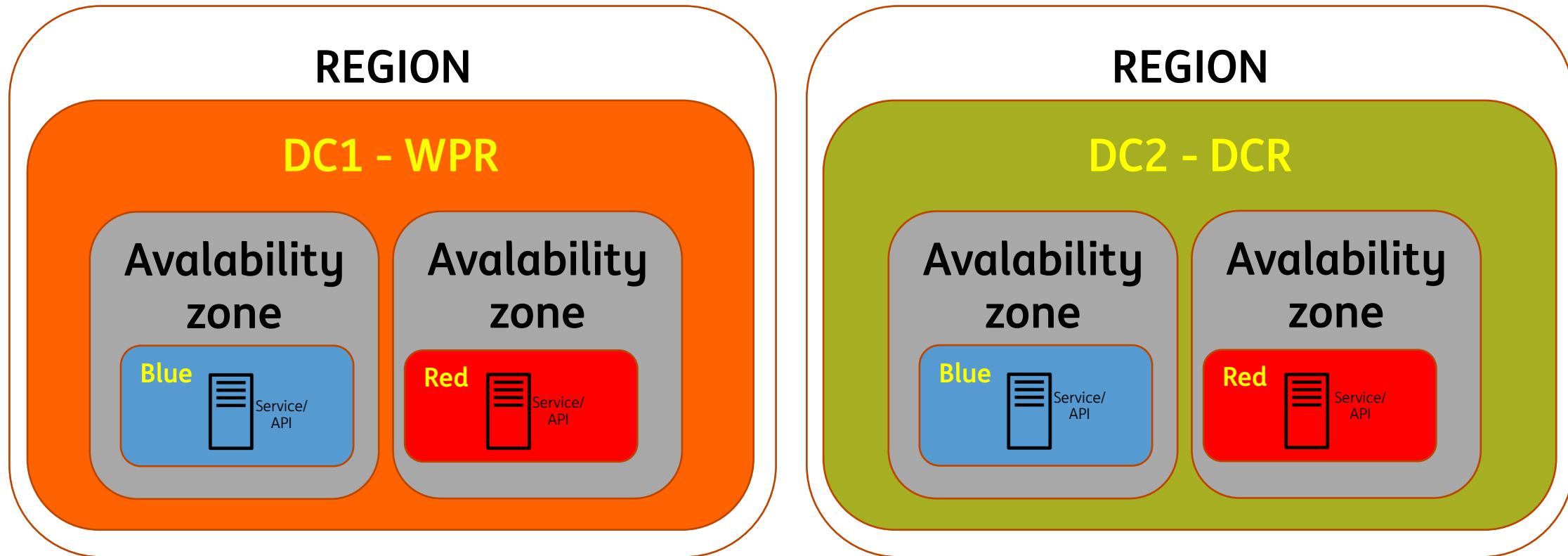


Rack B



Red and Blue,
Availability zones
@ING

High Availability Basics



For resilience and high availability reasons business critical applications need in total
FOUR times the capacity/resources

Why do you need Red and Blue availability zones?

Hardware
Failures

Maintenance,
patching by infra
provider

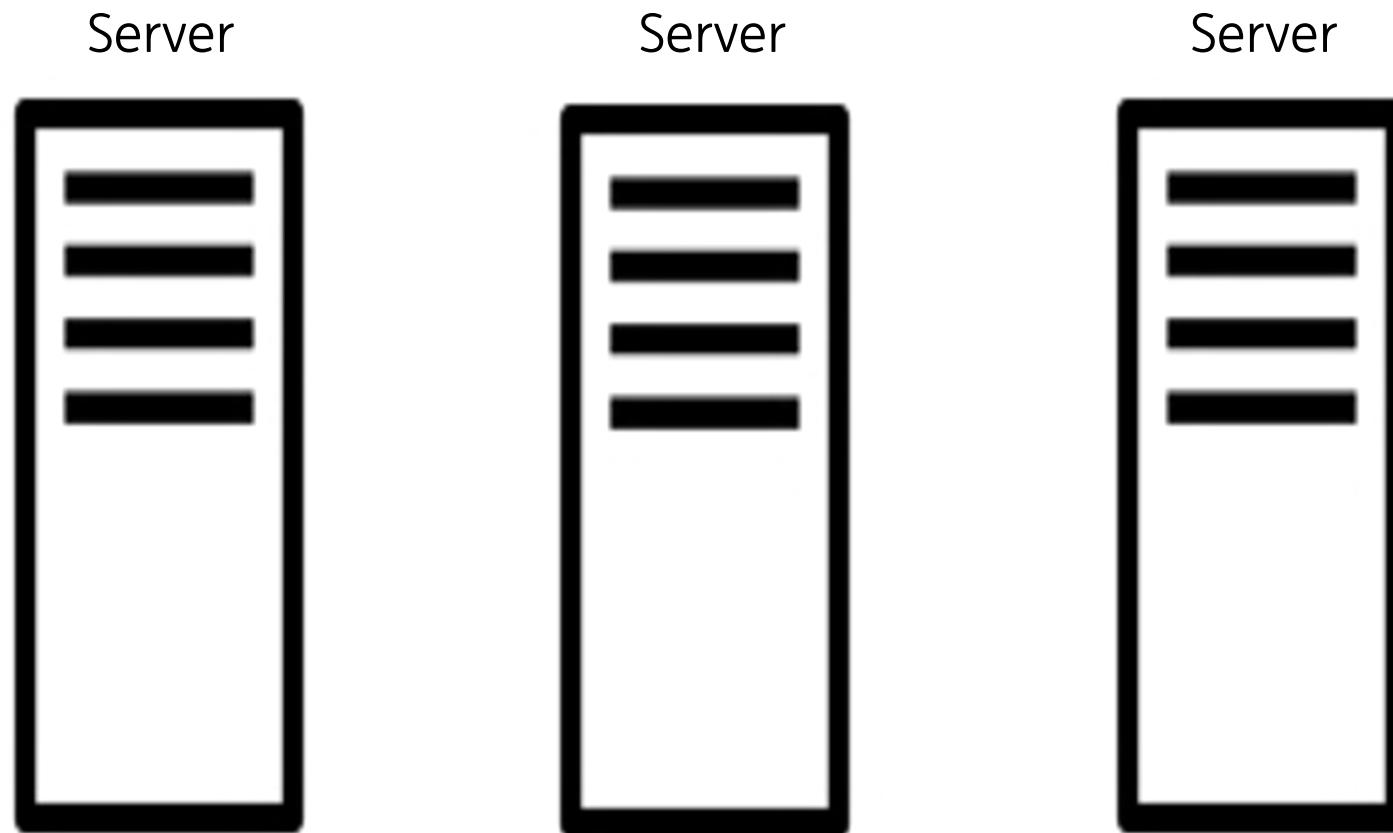
A different availability zone also means a different **update domain** and **fault domain**

High Availability Basics

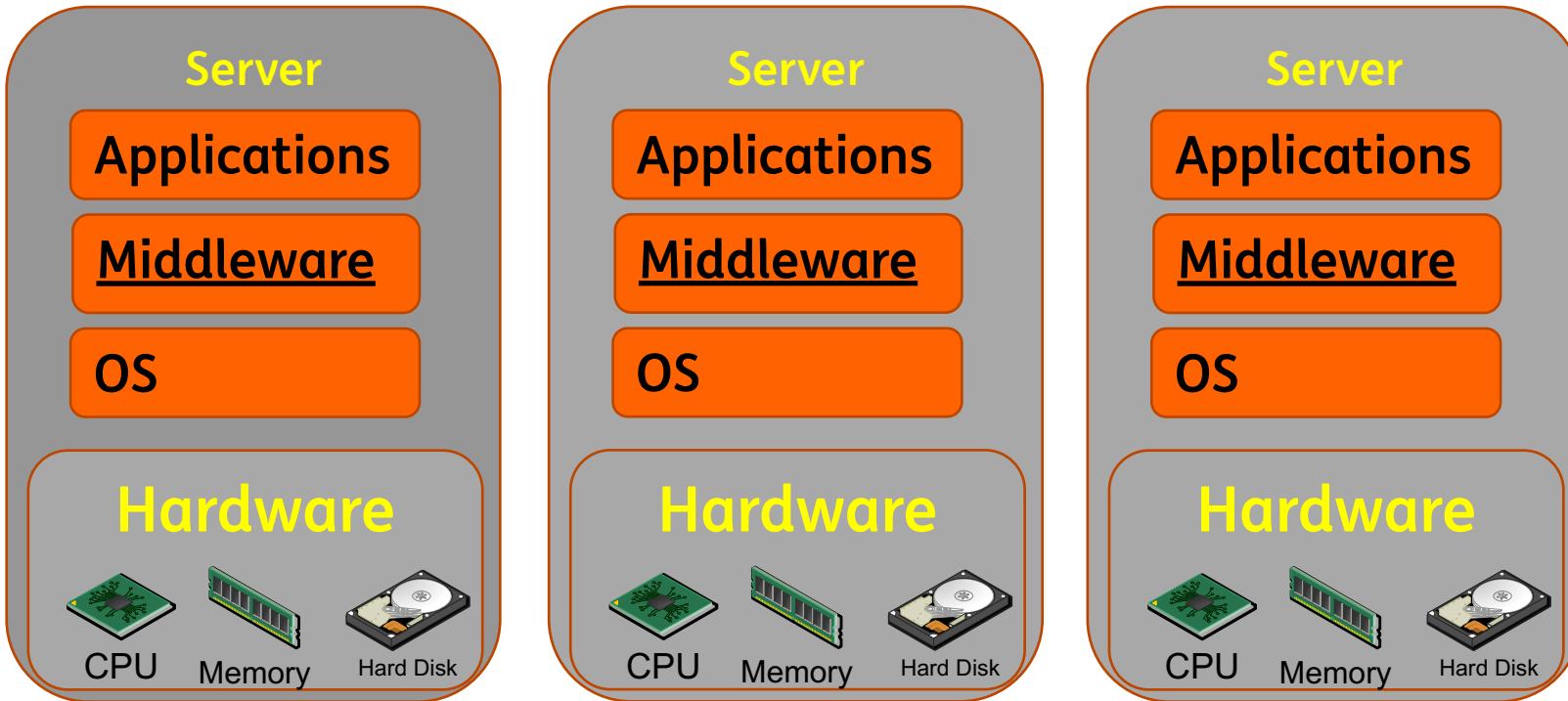
Having a second server in the same DC offers component failover.

However, does it offer **hardware failover**?

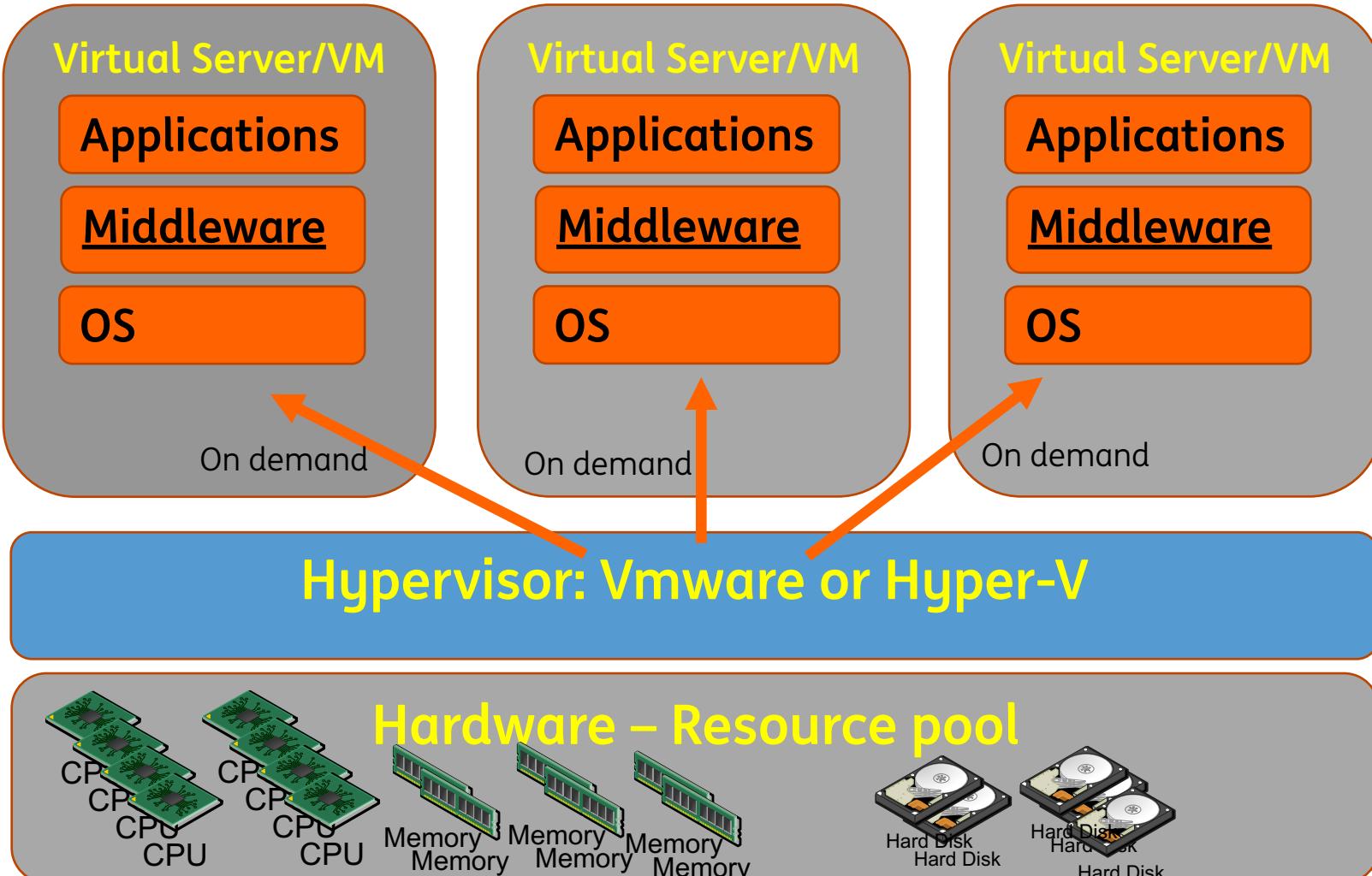
Traditional architecture



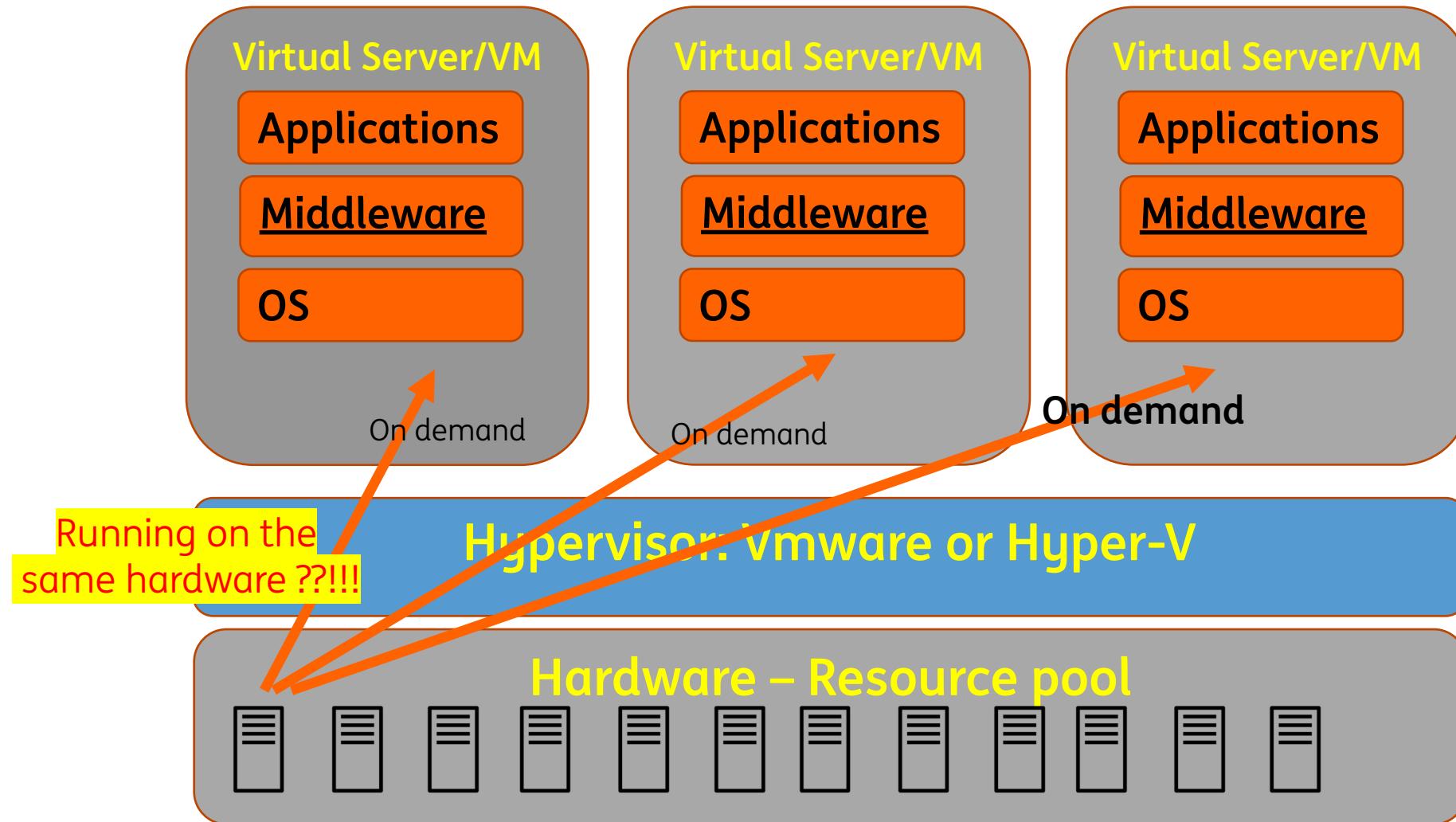
Traditional architecture



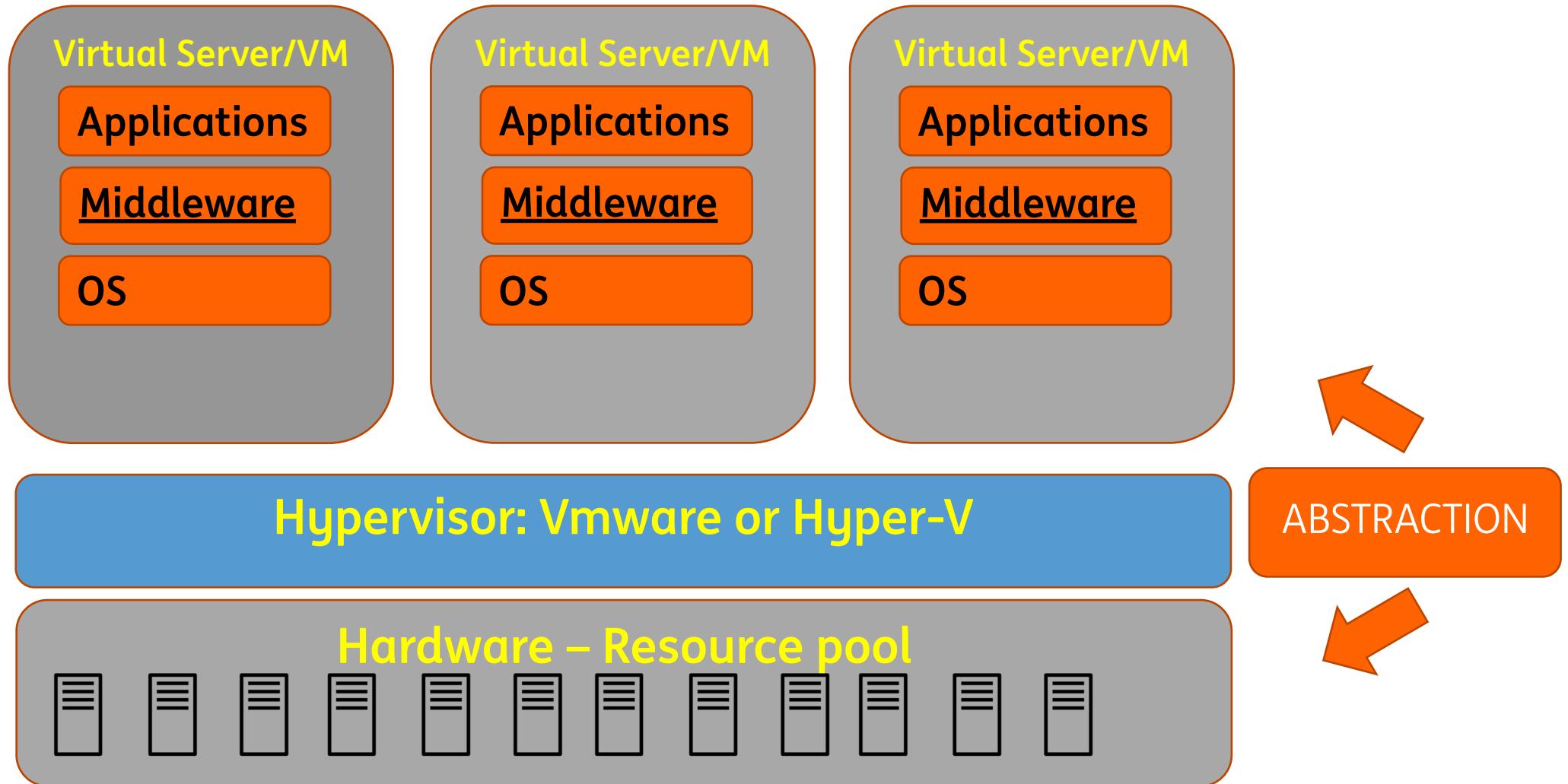
Virtual architecture



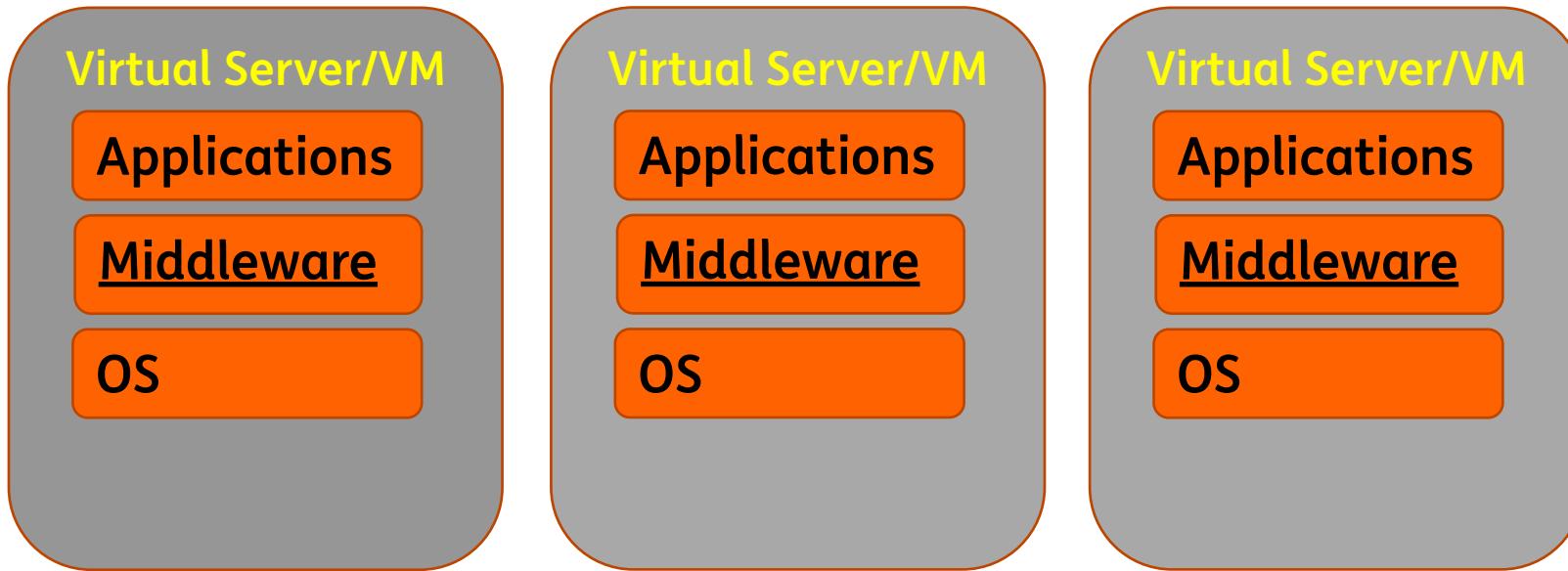
Virtual architecture



Virtual architecture



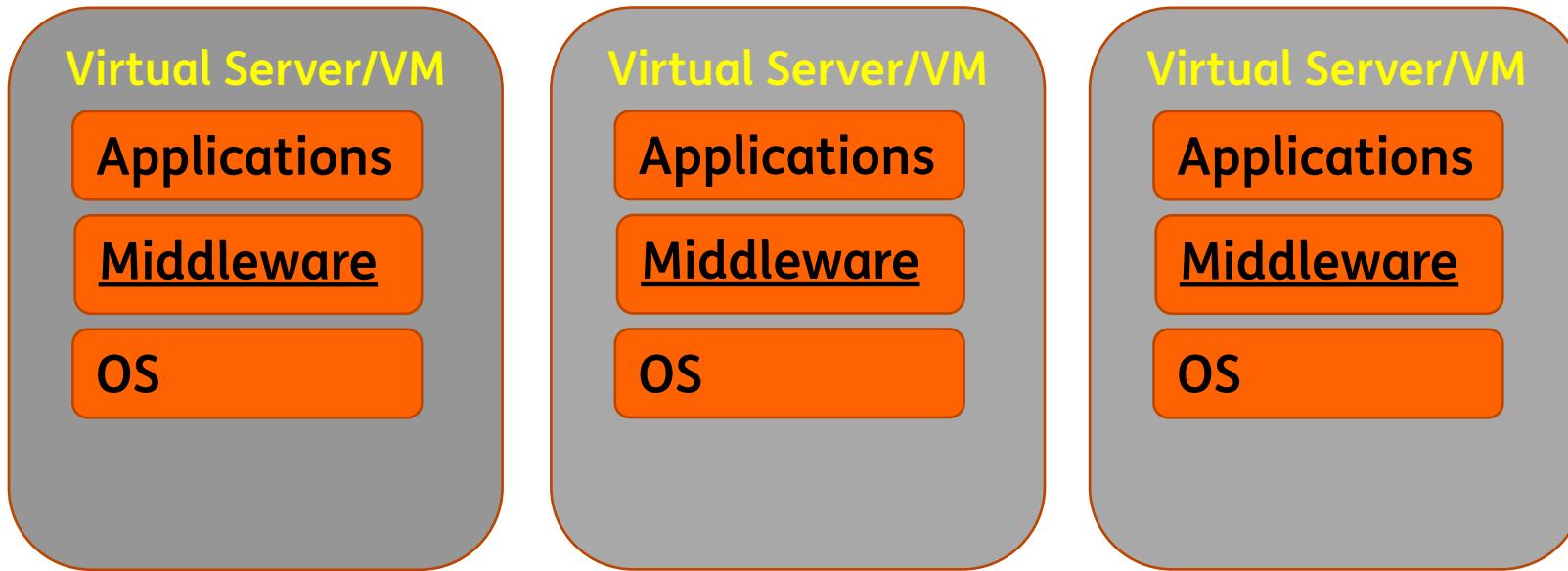
Virtual architecture



The application is not aware which hardware
is used, it is managed by the hypervisor

ABSTRACTION

Virtual architecture



Hypervisor: Vmware or Hyper-V



Alleys in a Datacenter

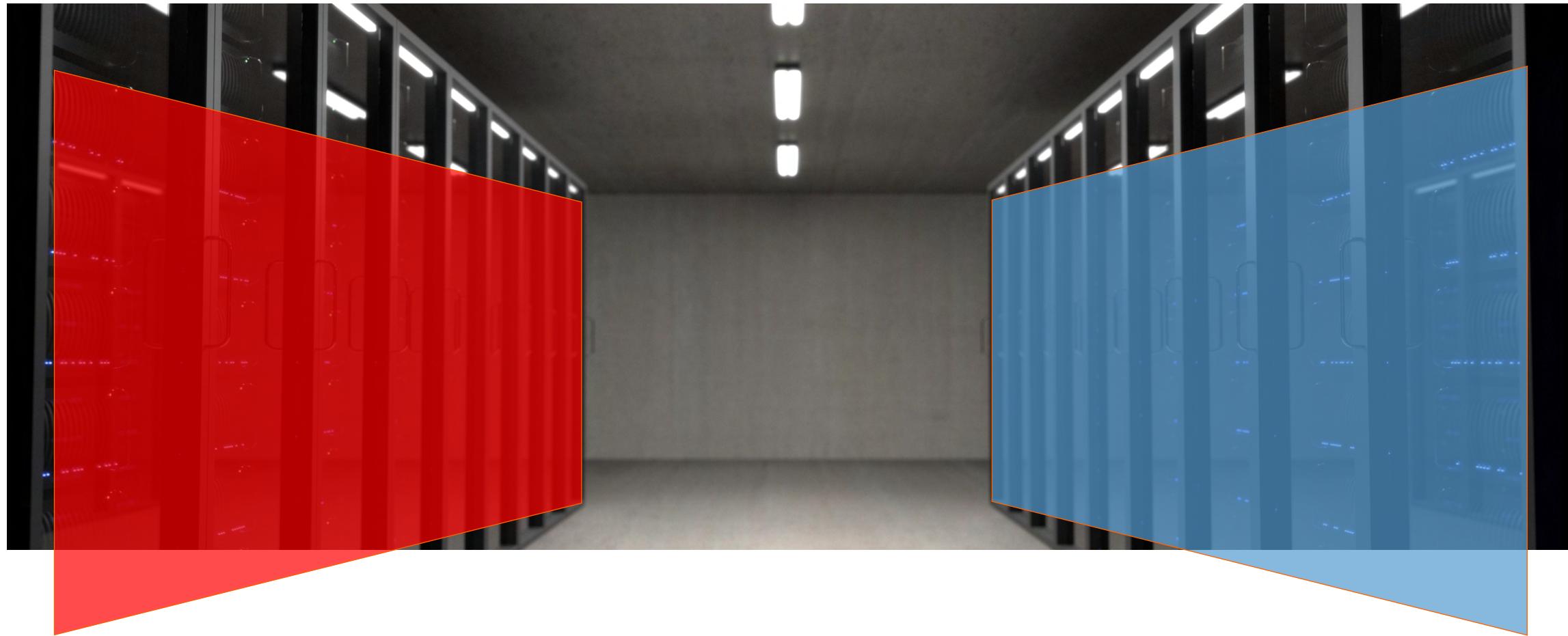
Hardware – Resource pool



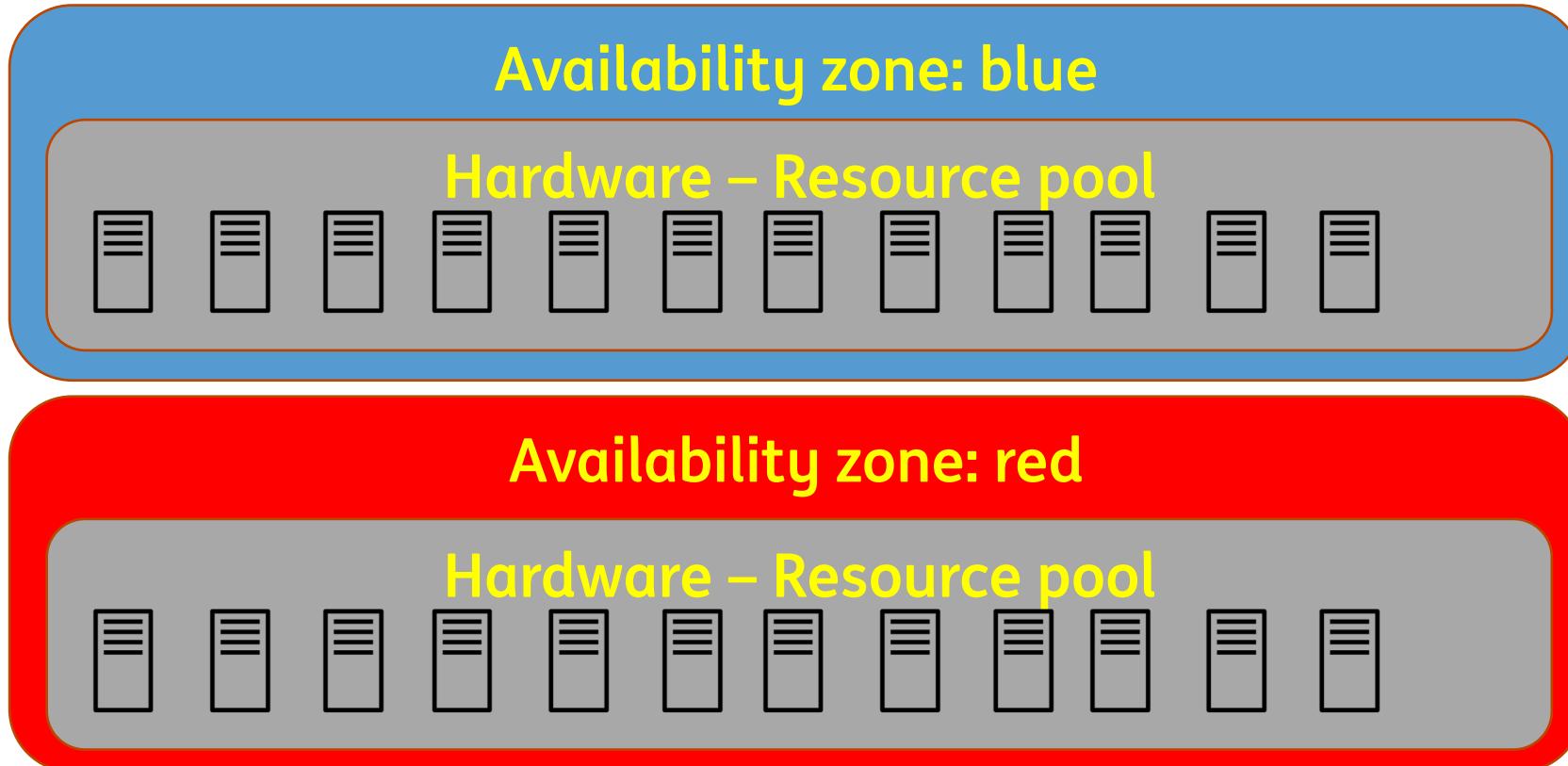
Hardware – Resource pool



Alleys in a Datacenter: Red and Blue



Availability zones: Red and Blue



BMaas – Bare Metal as a Service



Standalone Platforms (BMaaS) offering provide physical servers automated provisioning functionality as extension to current IPC service.

It fulfils the gap for applications demanding highly compute power .

This service provides various types of physical servers, including GPU's ([hardware details](#)), with minimal pre-installations which enables ING Business Partners to use IPC environment in case due to specific application requirements.

As the service is at a lower level as the pattern-based services, there are less components included into the delivery and the service has less integration out-of-the-box.

This means more work and responsibilities for the consumers.

Performance test types

- Peakloadtest
- Endurance (SOAK) test
- Breakpoint test
- Combitest
- Spike test
- NO / Slow backend test



Performance risks

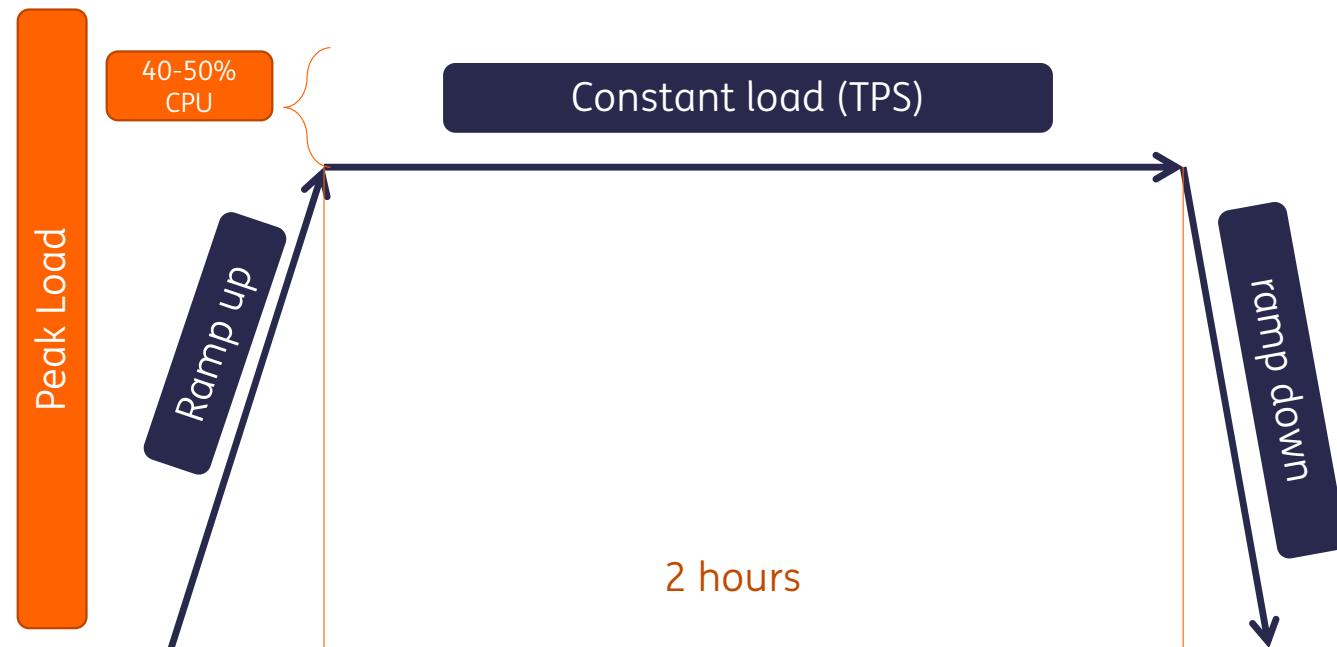
Risk type	Risk Description	Test
Reliability/ Scalability	Application response times met the performance requirements during peakloadtest	Peakload test
Reliability	Error percentages are too high under peak load	Peakload test
Scalability	The CPU usage of the API size not linearly with load	Breakpoint test
Stability	After an overload situation of the API, it does not recover bij itself	Breakpoint test
Stability	The API becomes unstable over time (mem, leaks, log are full, connection pools etc).	Endurance (SOAK) test
Stability	After a temporary unavailability of a backend, the performance is not how it should work.	Resilience test, no service
Stability	The performance test can handle PROD load on all endpoints	Resilience test
Stability	Application can handle unexpected peaks. Higher than estimated peakloadnumbers	Spike test

Loadtest

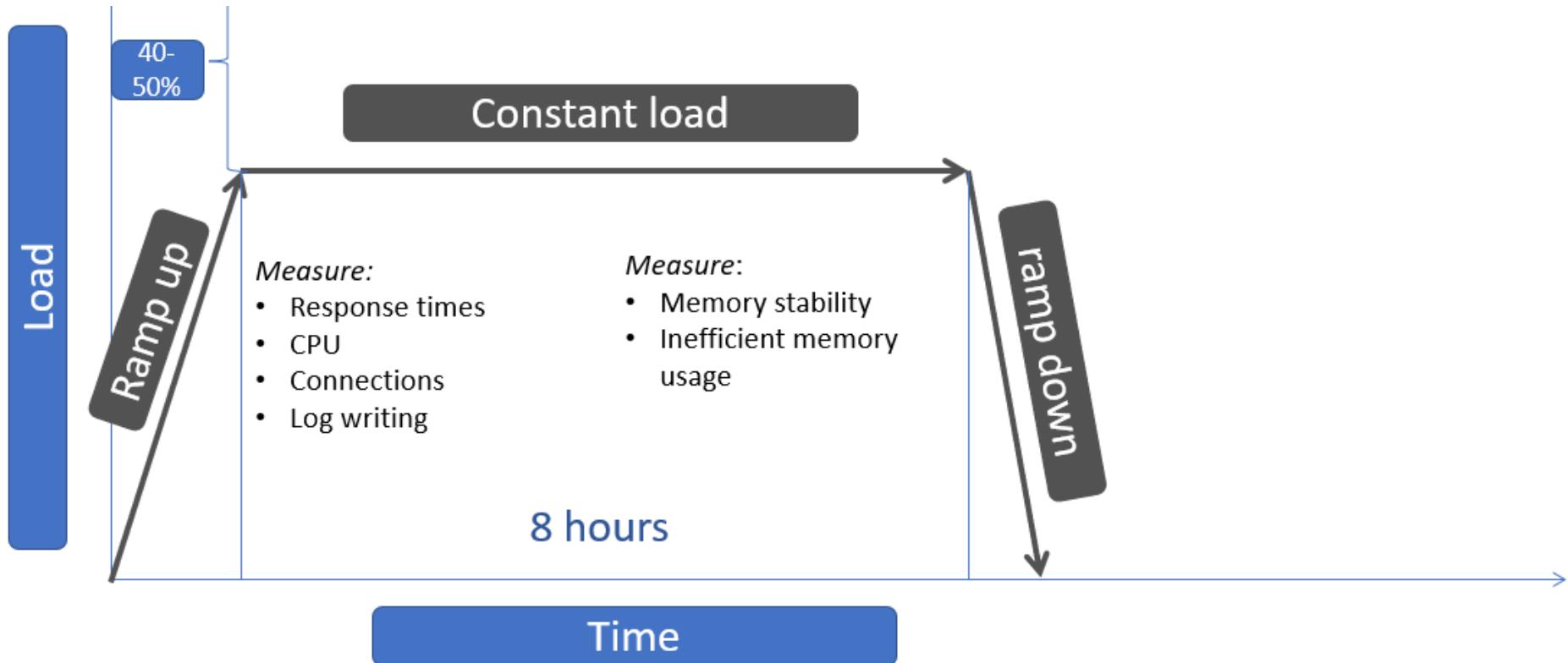
Simulate measured peakload or estimated loadnumbers for your API and endpoints.

What performance counters do you want to validate during this test?

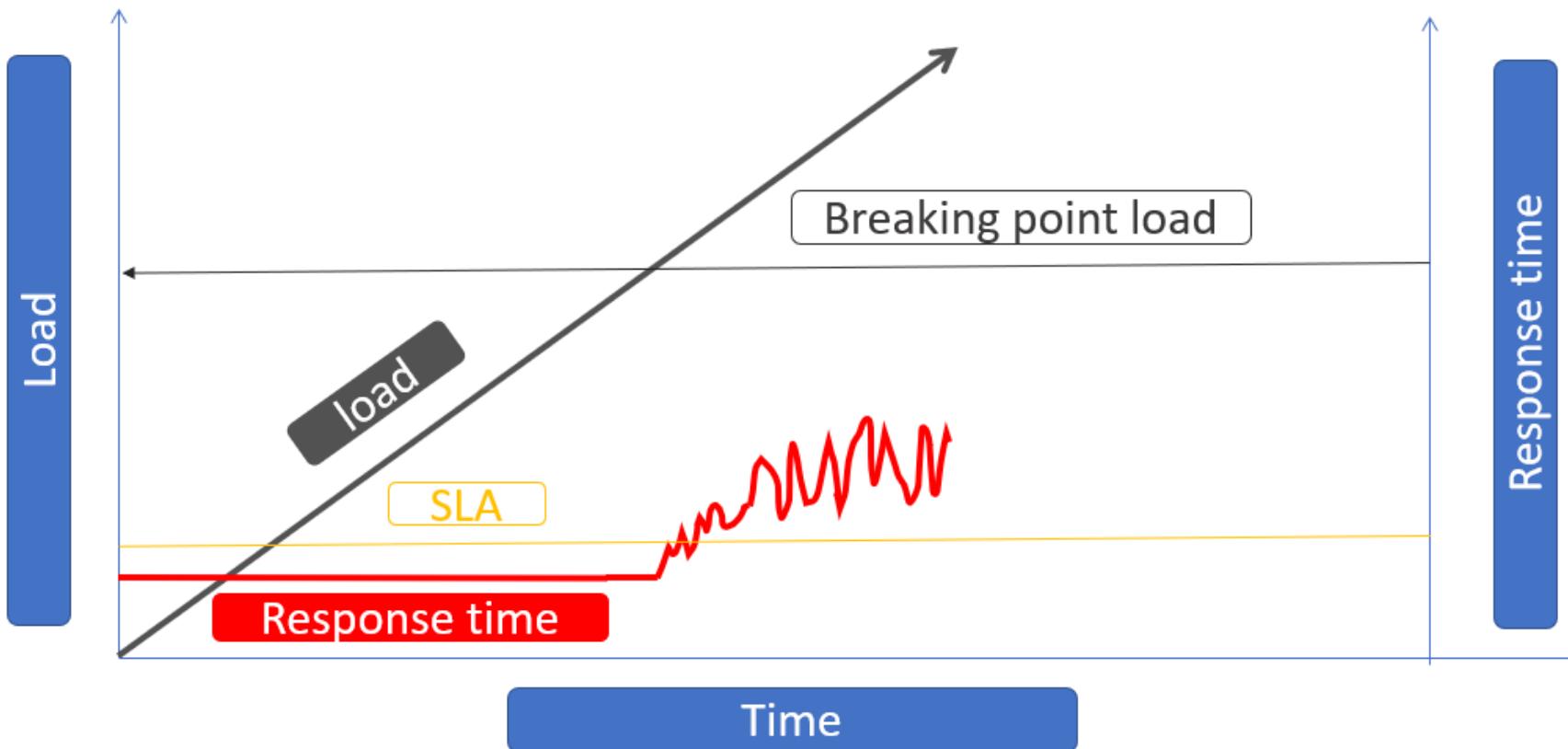
- Response times
- Error codes
- CPU usage
- Memory stability
- HTTP codes
- Log writing
- Connections



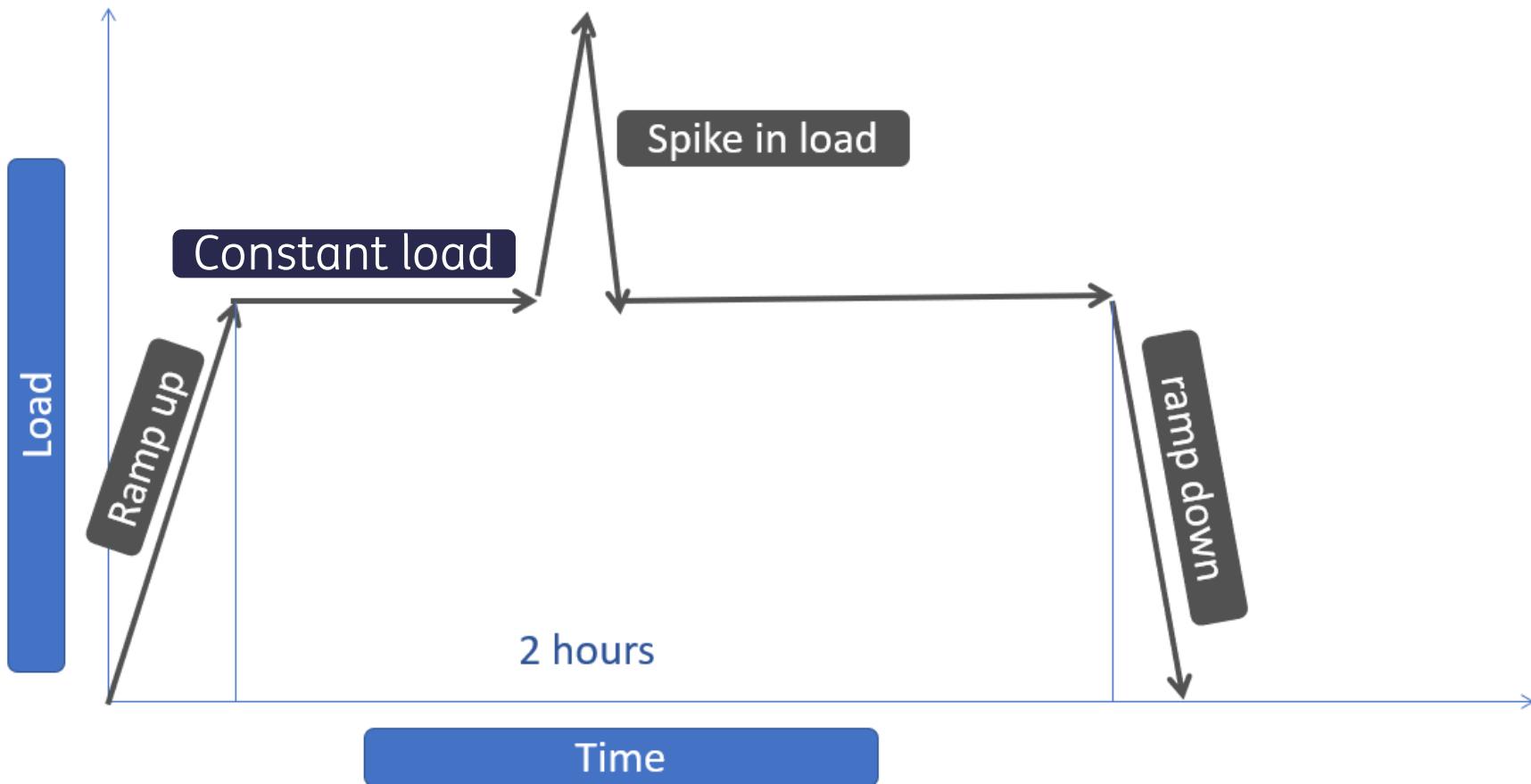
Endurance (SOAK) test



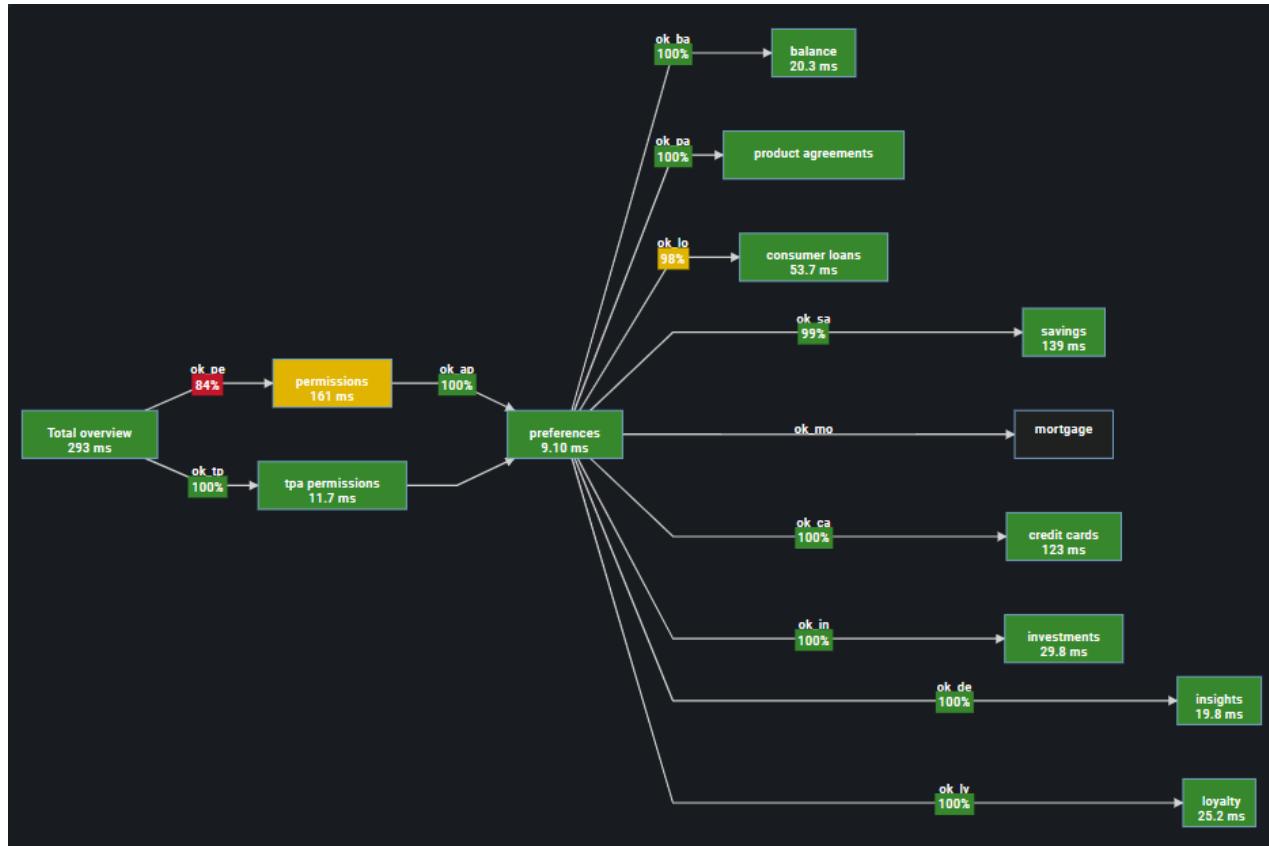
Breakpoint test



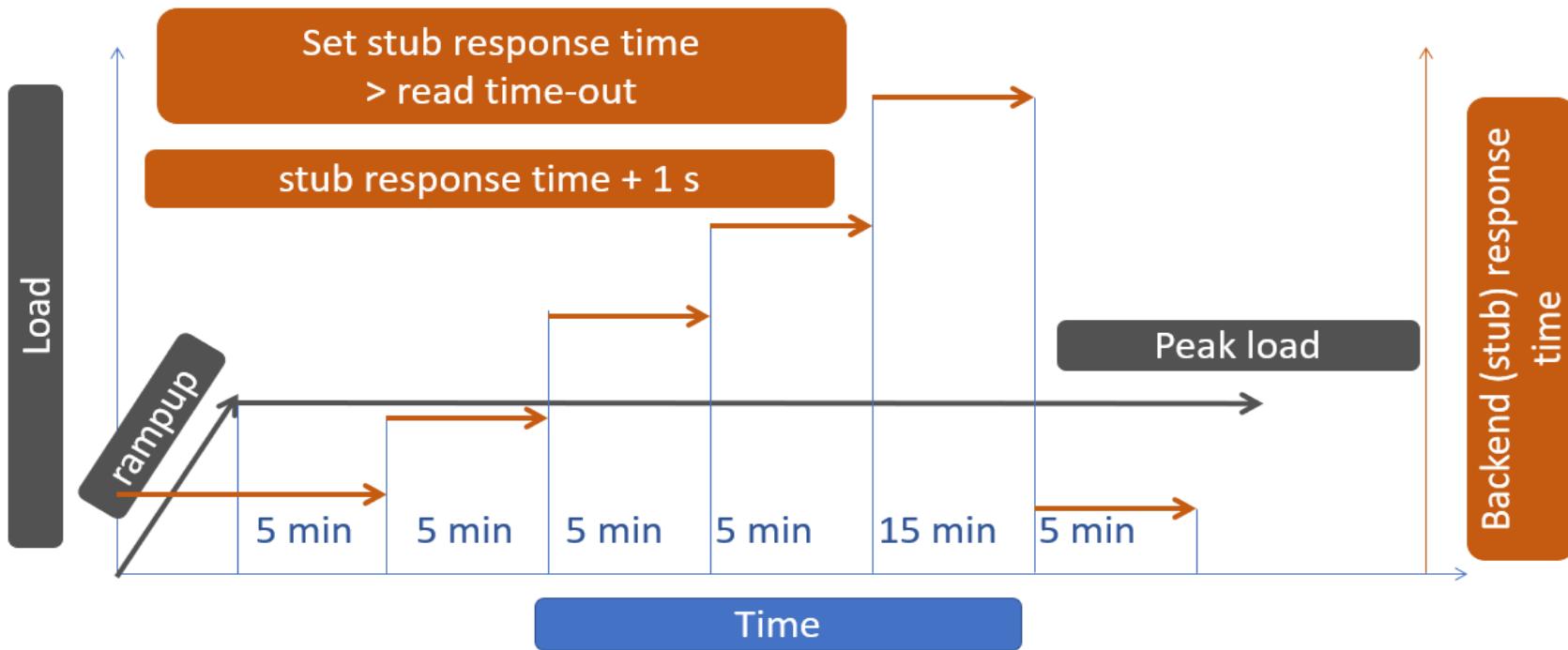
Spike test



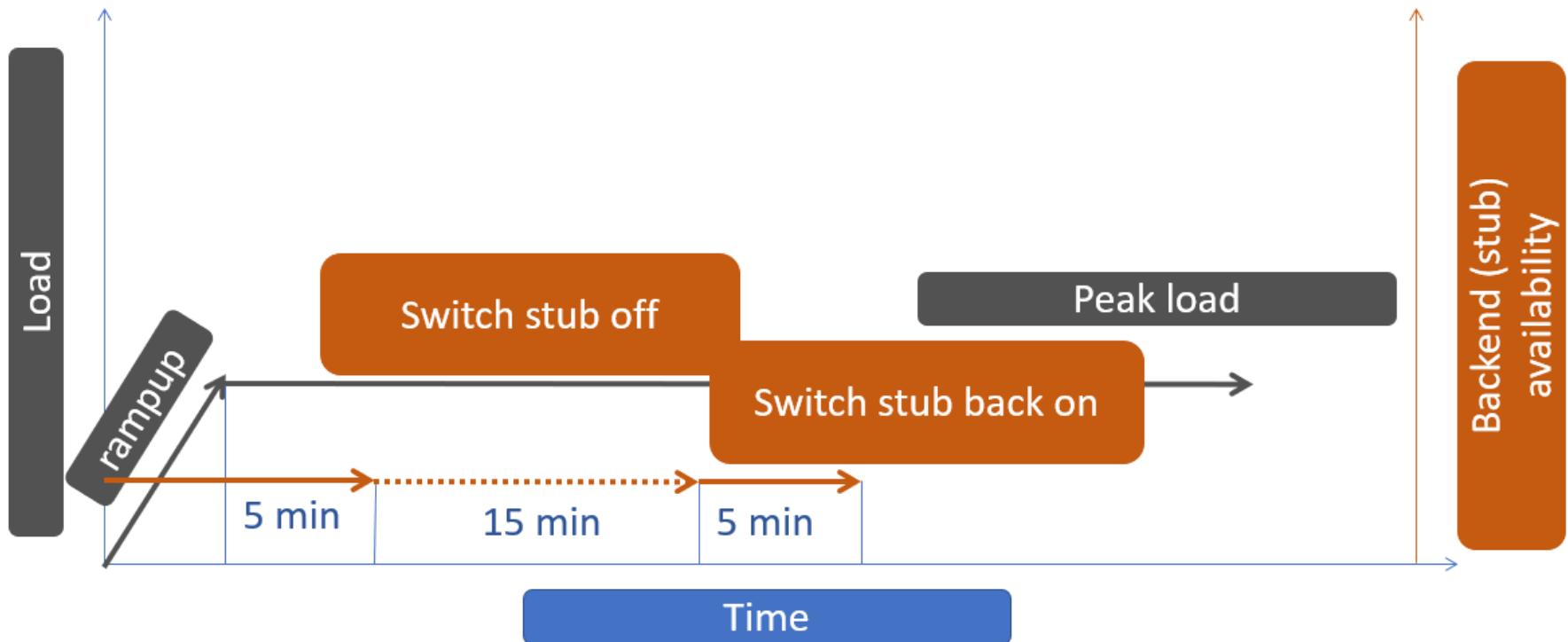
Resilience testing



Resilience test, slow responses from service



Resilience test, no response of service



Decision diagram, risk VS performance test types

Condition of the API		Combination				
		1	2	3	4	5
Do you have a high load API?		Y	Y	N	N	Y
Do you change more than 100 rules or 3 classes in the latest version of the API code?		Y	Y	N	Y	N
Do you connect to a new endpoint?		Y	N	Y	N	N
Testtype	Purpose		↓	↓	↓	↓
Loadtest	Run the loadtest for the API on expected peakload numbers. Validate resources like cpu, mem and poolsizes. But validate also response times, errors and http status codes. Compare with latest version of the API in acceptance but also towards PROD.	R	R	O	O	R
Endurance (SOAK) test	Run the loadtest for the API on expected peakloadnumbers for a longer period (8 hrs) while remaning a stable state of the API. Validate resources like cpu, mem and poolsizes. But validate also response times, errors and http status codes.	R	R	O	R	R
Breakpointtest / Spike test	Run the ramp up in levels towards the API. Testresults give an impression of the weakest spot in your chain when response times are increasing. A spiketest could also be part of this proces. validate what the behaviour will be of your API with unexpected increases of load in a short period of time.	R	R	O	O	O
Resilience test, Slow service	Simulate a slow endpoint during peakload. What is the impact on the API and other endpoints? Validate in mocked setup	R	O	R	O	O
Resilience test, No service	Simulate a broken endpoint during peakload. What is the impact on the API and other endpoints? Validate in mocked setup.	R	O	R	O	O
Combitest	Simulate your API seen from an END2END perspective and will pass all ING shared components in between your API landscape.	R	O	O	O	R
Explanation of Model						
Recommend	Recommendation					
Optional	Optional					



Analysis & Reporting + Evidence

Antipatterns

- Slopes
- Peaks
- Dips

Benchmark

- Compared to previous versions
- SLO's

Evidencing

- Azure testplan

Available tooling

- Conformance Testkit
 - The Conformance Testkit (**CTK**) is a CLI tool, that allows developers to write their application and focus on the happy path.
The CTK will take care of all the plumbing required for the testing of functional requirements and non-functional requirements.
 - See The Forge : <https://theforge.ing.net/product/238353/details/>
- MjolnirAPI
 - MjolnirAPI is providing a REST API to run Gatling tests in ICHP
 - See The Forge : <https://theforge.ing.net/product/239193/details/>
- k6
 - k6 is an open-source load testing tool that makes performance testing easy and productive for engineering teams. It is free, developer-centric, and extensible. We provide you with the necessary docker image and the steps required to make k6 work in ING-It environment. You can then adapt our example scripts to your product and contact us for any difficulties and questions that may arise.
 - See The Forge : <https://theforge.ing.net/product/239853/details/>

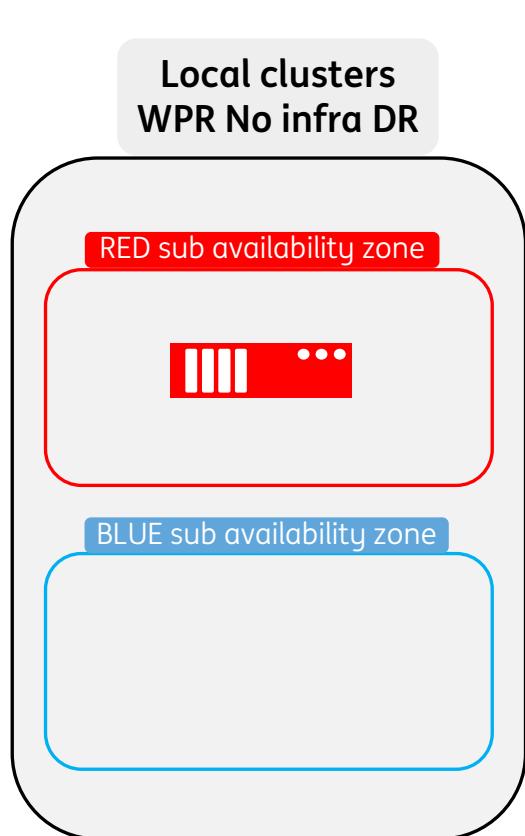


do your thing

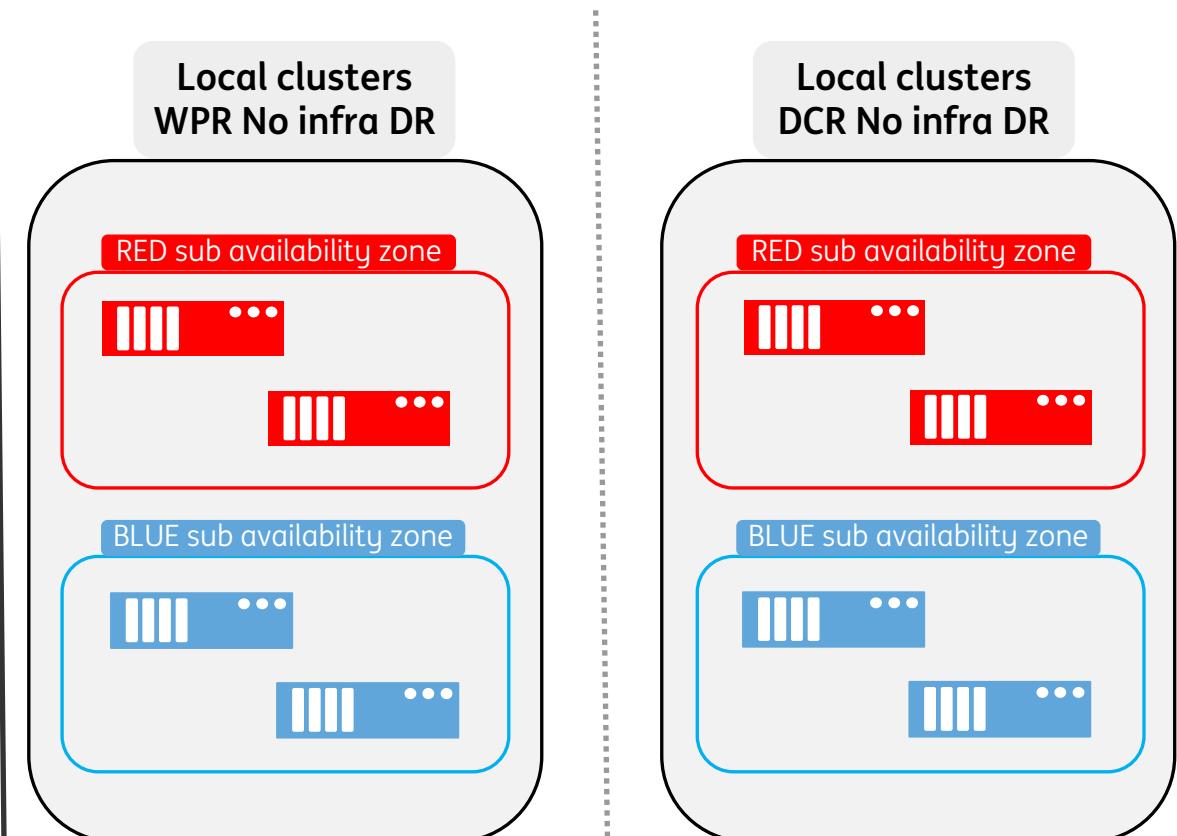
Correct use of sub-availability zones

Distribute your infra across availability zones and failure domains

Distribute your infra across availability zones and failure domains cross datacenter



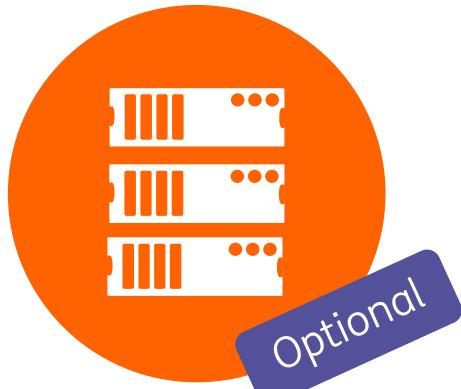
Distribute your infra using all failure domains across both datacenters



Performance monitoring

Back-ups and restores

IPC Back-up & Restore for virtual servers



Virtual Servers

- ❖ Retention time options of 14 days or 31 days
- ❖ Option of Image level or File level back-up
- ❖ Set your start time
- ❖ Suspend or resume back-up schedule

Service Level Gold

- Backup data stored in single data center

Service Level Platinum

- Only available for DR enabled virtual servers
- Backup data stored in both data centers

Patches and change strategy using IPC tools

Home > OS / Software | Patch > Patch Java

Patch Java

Patch the Oracle JSE 8 component For patching Java OpenJDK, use Patch RHEL New Java patch available since 29 June 2023, read more

① Selection ② Submit ③ Done

	VM name ↑	Patch status	Cloud	Applications	Environment	DC	AZ
VM name	CLRV0000177560 CLRV0000177564 CLRV0000223001	Patched Patched Patched	IPC		Development	WPR	Blue
Patch status	CLRV0000225103 CLRV0000251742 CLRV0000264579	Patched Patched Patched	IPC		Test	DCR	Blue
Cloud	CLRV0000271370 CLRV0000271371	Patched Patched	IPC		Development	DCR	Red
Applications	CLRV0000271376 CLRV0000271379	Patched Patched	IPC		Development	DCR	Blue
Environment	CLRV0000271406 CLRV0000271407 CLRV0000271408	Unknown Patched Patched	IPC		Development	WPR	Blue
Datacenter	CLRV0000271410 CLRV0000271413	Patched Unknown	IPC		Development	WPR	Red
Availability zone	Blue (11) Red (12) Unknown (16)						

Items per page: 15 1 - 15 of 39

① Selection ② Submit ③ Done

Clear filter

VM name

Patch status

Cloud

Applications

Environment

Datacenter

Availability zone

Next

① Selection ② Submit ③ Done

Clear filter

VM name

Patch status

Cloud

Applications

Environment

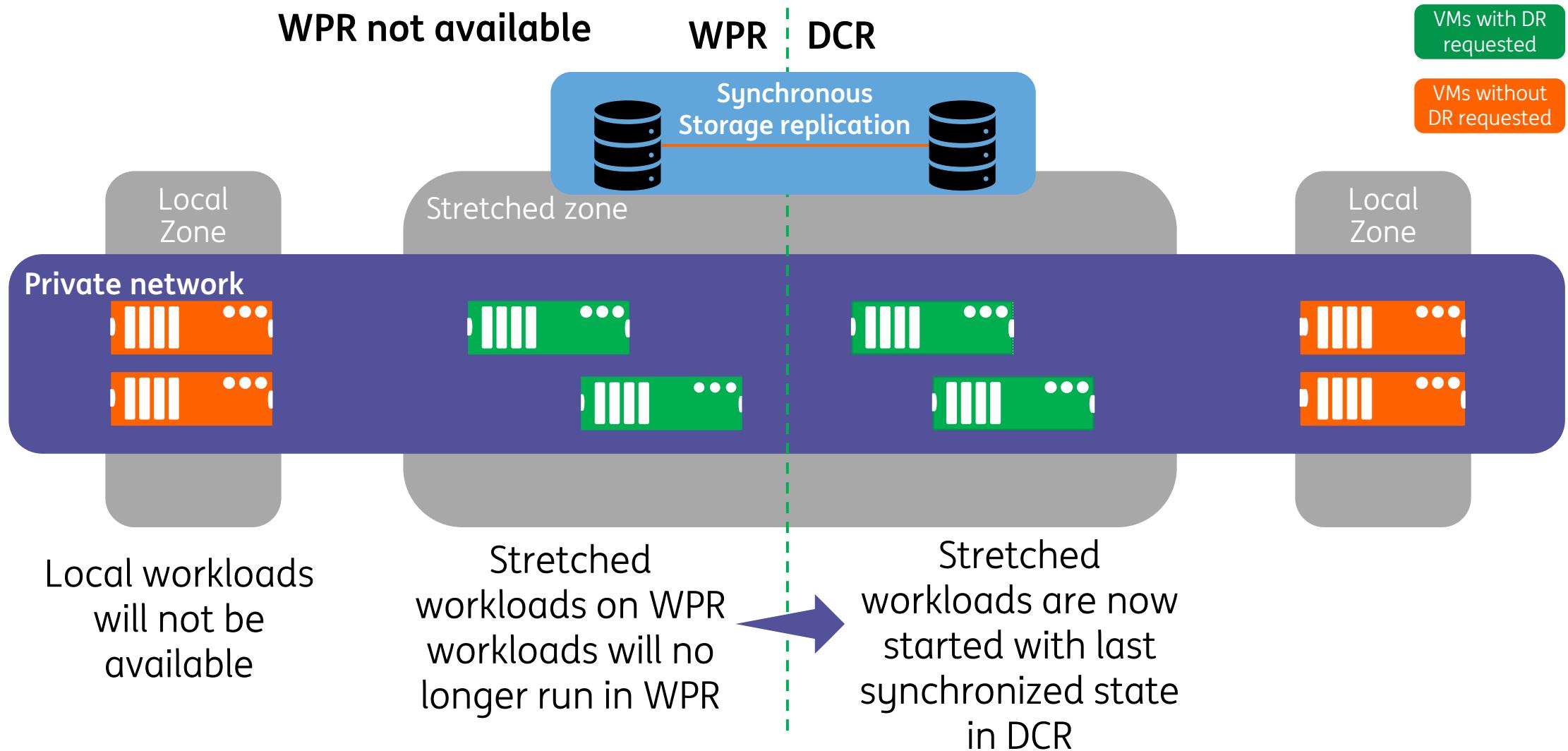
Datacenter

Availability zone

Next

Disaster recovery scenarios

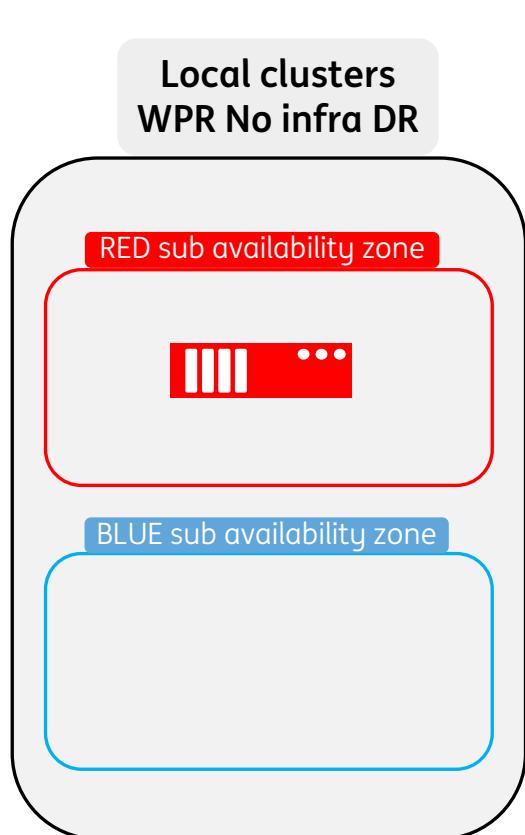
In case of data center outage



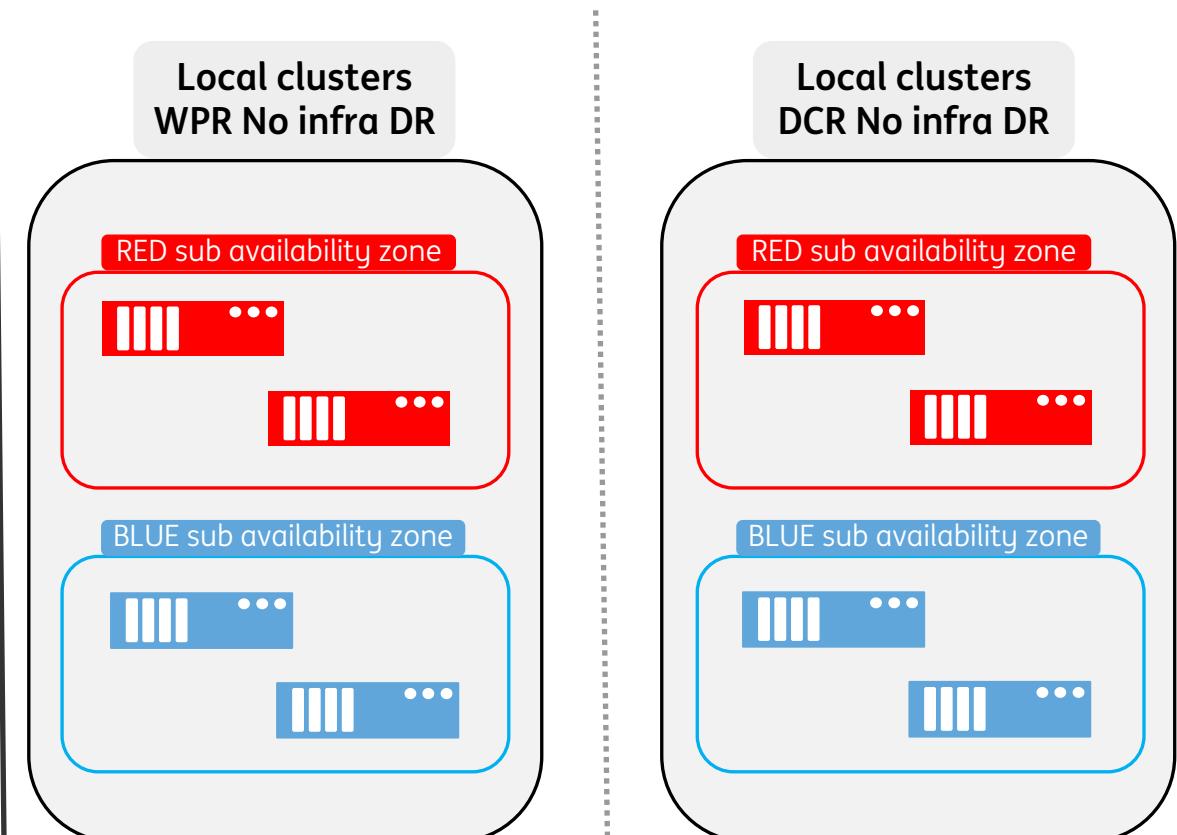
Correct use of sub-availability zones

Distribute your infra across availability zones and failure domains

Distribute your infra across availability zones and failure domains cross datacenter



Distribute your infra using all failure domains across both datacenters



Performance monitoring

ServiceNow

* Name	[REDACTED]		* Config Admin Group	Tech/Infra/Cloud/IaaS/Microsoft Services
* Status	Installed		* Company	ING
: Operational status	Operational		Cost center	[REDACTED]
Environment	Production		Business Unit	
CI Alias	[REDACTED]		* IT Custodian	[REDACTED]
* Asset Owner	[REDACTED]	<input type="button" value="Search"/>	Tech PL Environment List	
ING Category	Windows			
ING Identifier	[REDACTED]			
Entity	ING Bank Netherlands - Bank	<input type="button" value="Search"/>		
Very	Monitoring	Relation	History	
Monitored	<input checked="" type="checkbox"/>	checked!	Choose GMCR!	Monitored By
Accepted	<input checked="" type="checkbox"/>	checked!		Accepted By
Standby	<input type="checkbox"/>		Vulnerability Scanning	Thorough
Is Event 2 Console	<input type="checkbox"/>		Technical State Compliance Monitoring	Yes
Auto-Ticketing Events	<input type="checkbox"/>	unchecked!		

Check Disable Auto-Ticketing Events

Detail*H.Node		Discovery	Monitoring	Relation	History
Monitored	<input checked="" type="checkbox"/>			Monitored By	GMCR
Accepted	<input checked="" type="checkbox"/>			Accepted By	
Standby	<input type="checkbox"/>			Vulnerability Scanning	Regular
Is Event 2 Console	<input type="checkbox"/>			Technical State Compliance Monitoring	Yes
Disable Auto-Ticketing Events	<input checked="" type="checkbox"/>				

Onboard on IAT

1 Choose Your Application 2 Choose Your Team 3 Team Details Done

Onboard an application with IAT

Fill in the name of the asset you want to onboard and press the onboard button. At this point it is possible to onboard ServiceNow Business Applications and IT Products. We will try to collect the information about your asset from ServiceNow/ACE, this might take a few seconds.

Select the type of asset you are onboarding

First choose an asset type

Continue

Find the latest information on the forge

ING Home Marketplace News Journeys Search Login

Overview News Reviews Version created: 03.08.2023 Last verification: 03.08.2023 Suggest a change

On-board your Application

Prerequisites

- Your Business Application is properly registered in ServiceNow with an Asset Owner and an IT Custodian.
- Put a code should be linked with one L2 support offering in ServiceNow.
- Onboarding your application in IAT is **case sensitive**. Therefore, the Business Application reflected in CMDB should be the exact name to be onboarded in IAT.
- Open firewall to IAT Production - **MDPL-Collection with HTTPS port**.

Please refer to these screenshots as an example to link one **L2 Support Offering** in ServiceNow:

SquaredUp sample

All SCOM Alerts DRAFT

hour | 12 hours | 24 hours | week | all

SQL 2008 DB Average Wait Time is too high	MSSQLSERVER SPNLAPP74061.europe.intranet	a minute ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SDNLAPP72249.europe.intranet	2 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SANLAPP79018.europe.intranet	2 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER spnlapp73039.europe.intranet	2 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SANLAPP79046.europe.intranet	2 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	3 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	3 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SDNLAPP72005.europe.intranet	4 minutes ago
SQL 2008 DB Average Wait Time is too high	MSSQLSERVER SPNLMTG74026.europe.intranet	6 minutes ago
SQL 2012 DB Average Wait Time is too high	MSSQLSERVER SJ4000000277.ad.ing.net	6 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SDNLAPP73034.europe.intranet	7 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	10 minutes ago
SQL 2008 DB Average Wait Time is too high	MSSQLSERVER STNLAPP73005.europe.intranet	11 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78047.europe.intranet	11 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	13 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	13 minutes ago
Operations Manager failed to start a process	Health Service SPNLAPP78003.europe.intranet	13 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SPNLAPP72203.europe.intranet	13 minutes ago
IIS 8 Application Pool is unavailable	TeamCentral SJ4000000643.ad.ing.net	13 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	18 minutes ago
SQL 2008 DB Average Wait Time is too high	MSSQLSERVER SPNLMTG75072.europe.intranet	19 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	19 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	19 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	20 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	21 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	21 minutes ago
ING GALM Alert	Microsoft Windows Server 2008 R2 Enterprise SPNLAPP78046.europe.intranet	21 minutes ago
SQL 2008 DB Average Wait Time is too high	MSSQLSERVER SPNLAPP78049.europe.intranet	24 minutes ago
SQL 2012 DB Average Wait Time is too high	MSSQLSERVER SPNLMTG74006.europe.intranet	30 minutes ago
SQL DB 2008 Engine Page Life Expectancy is too low	MSSQLSERVER SPNLAPP72226.europe.intranet	31 minutes ago

quick find

edit

Health System Center suite

ING

Operations Manager

Configuration Manager

Orchestrator

Back-ups and restores

IPC Back-up & Restore for virtual servers



Virtual Servers

- ❖ Retention time options of 14 days or 31 days
- ❖ Option of Image level or File level back-up
- ❖ Set your start time
- ❖ Suspend or resume back-up schedule

Service Level Gold

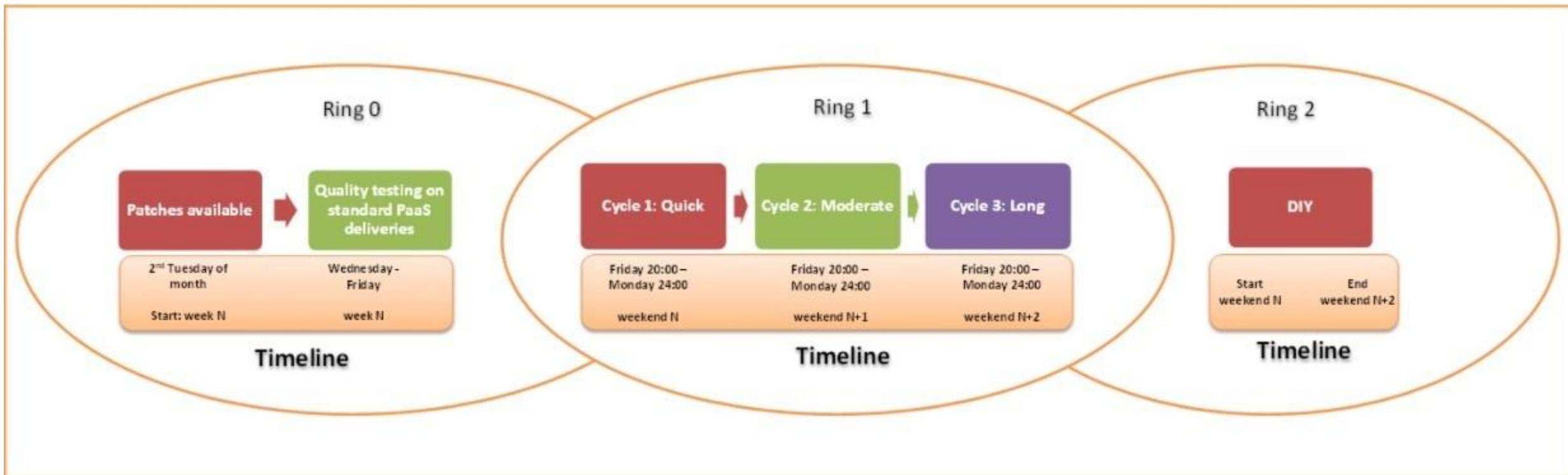
- Backup data stored in single data center

Service Level Platinum

- Only available for DR enabled virtual servers
- Backup data stored in both data centers

Patches and change strategy using IPC tools

Ring model



View patch window in SCOM

The screenshot shows the System Center Operations Manager (SCOM) web interface. At the top, there is a navigation bar with links like Alerts, BMG, CIO, DAS, GS, INFRA NL, TECH, WBS, BE CoreBank, Default, Teams, Getting Started, Overview, Overview, NOC, Applications, Microsoft Services, Applications, a search bar, and a menu icon.

The main content area displays the status of an SCCM agent named SX9000074681.ad.ing.net. The page includes tabs for IIS WEB SERVER, WINDOWS SERVER (2016 OR 2019), WINDOWS SERVER (2016), MEMORY, NETWORK, DISKS, MONITORED ENTITY, PERFORMANCE, ALERTS, VADA, CUSTOM, and SERVICENOW. The SCCM AGENT tab is highlighted with a yellow box.

Status of SCCM agent

Server Name	SCCM Agent Activity Status	Last Active Time	Last Health Evaluation	Last Evaluation Healthy	Last Health Evaluation Result
SX9000074681	Active	August 2nd 2022, 1:01:05 pm	February 18th 2022, 9:00:28 pm	Evaluation Succeeded	Pass

Health Status

SX9000074681
Healthy
Last Health Evaluation: February 18th 2022, 9:00:28 pm

Active Status

SX9000074681
Healthy
Last Active Status: August 2nd 2022, 1:01:05 pm

Additional informations

If you have experienced any issues with your SCCM agent installed on IPC host, please follow up this Confluence article: click [HERE](#)
Patching process and maintenance windows are described here: click [HERE](#)

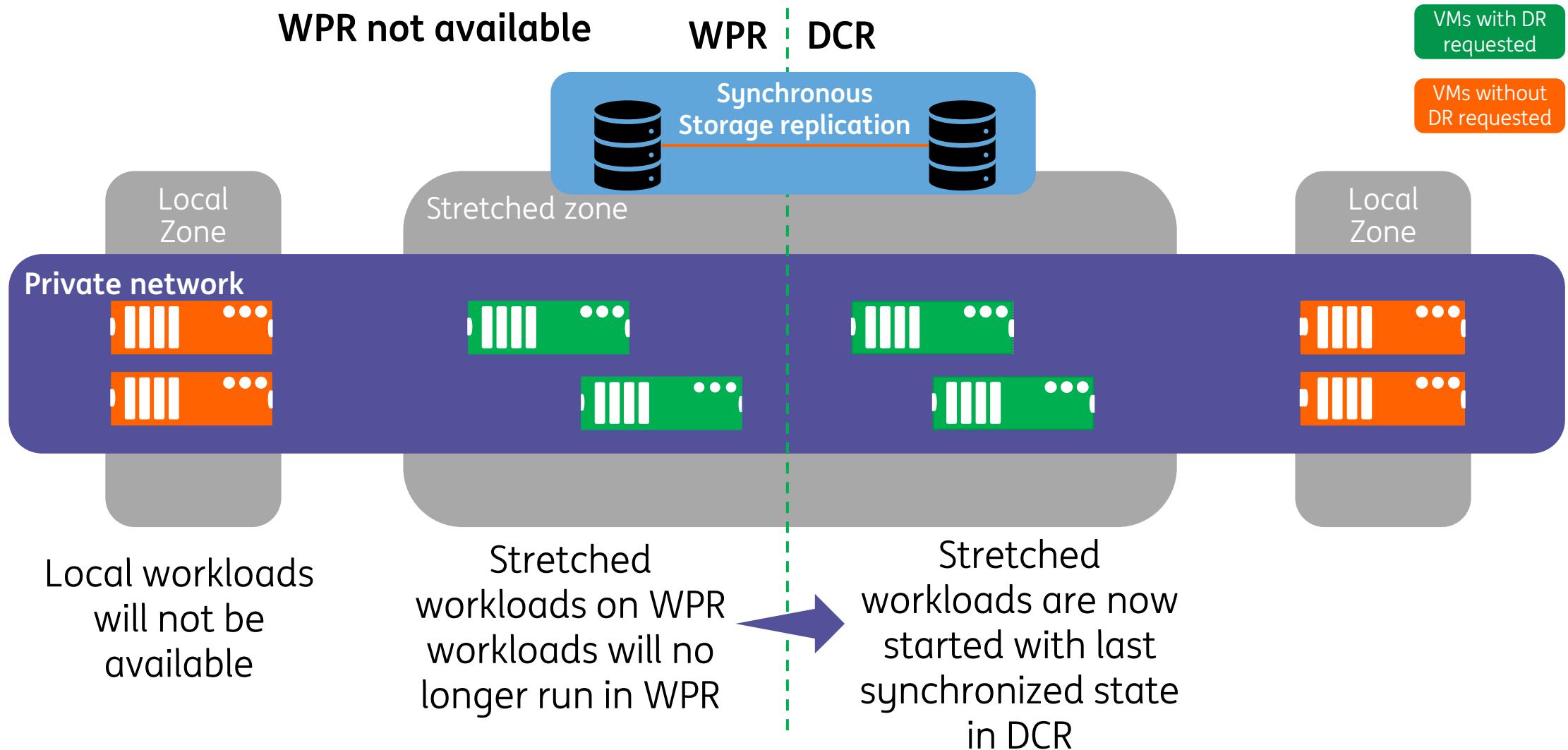
Patching Maintenance Window

Server Name	Maintenance Window
SX9000074681	RING R1C2W10 Saturday N+1 00:00 to 06:00

version 5.4.1.8070 | SCOM Edition | © copyright 2022 Squared Up Ltd.

Disaster recovery scenarios

In case of data center outage





Reliability

Reliability training



do your thing

Introduction Reliability training

This training offers an introduction and overview of IT services and essential resilience design principles as well as a deep dive into services as for example DBaaS, TPA / OTP services (Touchpoint servicemesh, API gateway, eventbus) and Observability tooling (TracING, RTK2, IAT etc.)

The goal of this training is to increase the reliability of ING IT.

We want to help the squads and ING engineers by providing the essential knowledge of ING IT services, best practices and design principles with a focus on resilience and high availability.

The target is to offer one platform and one format for all trainings.

Because of the urgency of reliability, we offer this MVP using the existing training materials and elearnings. Some subjects might be offered as documentation, links to the Forge for example.



Introduction Reliability training

Reliability

Reliability ensures that your application can meet the commitments you make to your customers. Architecting reliability into your application framework ensures that your workloads are available and can recover from failures at any scale.

Resilience

Resiliency is the ability of a system to gracefully handle and recover from failures, both inadvertent and malicious.

High Availability

High availability (HA) is the ability of a system (applications and data) to operate continuously without failing for a designated period of time. High availability (HA) is a process that eliminates the single points of failure (SPOF's).

On our way to a seamless digital experience

Our purpose is to

Empower people to stay a step ahead in life and in business

We'll be



Going for

Superior customer experience

Sustainability at the heart

We'll do this by

Providing seamless, digital services

Using our scalable Tech & Ops

Staying safe & secure

Unlocking our people's full potential

And to improve our Reliability, we focus on

- Applications
- Platforms / Services
- Infrastructure
- Processes / Operational Due Diligence
- Architecture

Why is this urgent?

ING customers furious about yet another malfunction: 'What a worthless company'

Problems at ING over, customers angry over of yet another malfunction in a short time

ING malfunctions frustrate entrepreneurs



More malfunctions at ING than at other banks, customers experience a lot of inconvenience

پاسکال @DarkStar755 • 4 d ...
@ingnl payed for a bed
yesterday and now I find out I
can't pay for petrol. 2x 2300
reserved. I'm in Germany 🇩🇪.
Now what????
1 1 1 48

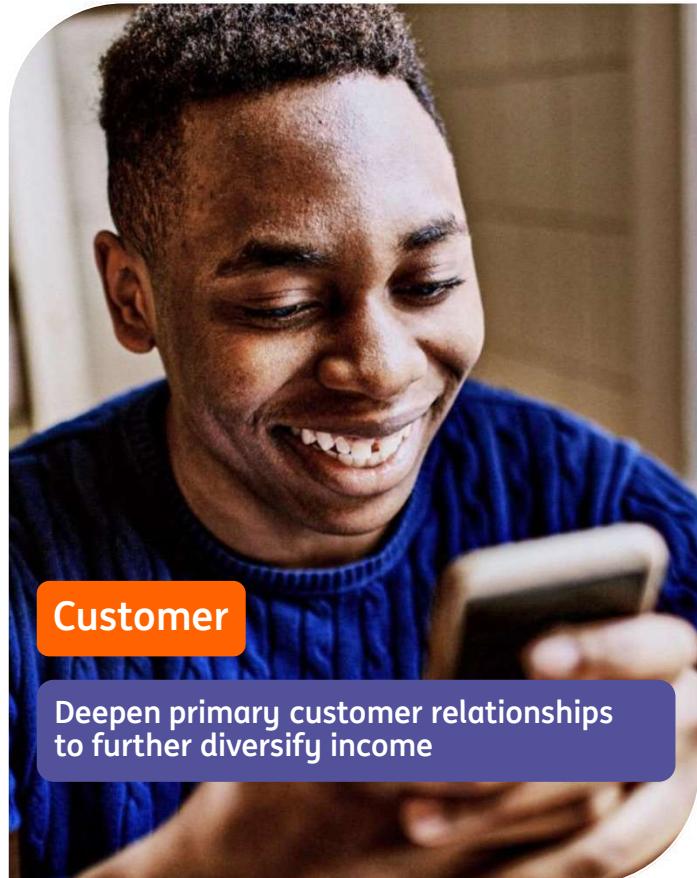
پاسکال @DarkStar755 • 3 d ...
You call that unfortunate???
That's quite an understatement.
I'm waiting for hours because
my bank account is hijacked!!!
1 1 1 37

ING Nederland 02-06-2023
Hi! Our colleagues do their
utmost to solve the problems
as quickly as possible. Sorry for
the inconvenience! We do not
offer any compensation for this.
This is included in our terms
and conditions: ing.nl/media/ING_voorwaarden..
12 1 1 2.720

Ray2401 @ray24... 02-06-2023
Well, that's easy. Every time
apologies and referring to the
conditions. When will it actually
be resolved? Who's in charge
over there, those 2 old muppets
on that balcony?
1 1 36 4.648

Malfunctions at ING cause business annoyance and damage

We are obliged to deliver an optimal experience for our customers by delivering on channel availability of 99.78%.



- **Channel Availability definition.**
A customer can Log-on
• Sees his/her Current Account overview & balance
• And is able Initiate a payment transaction & return to overview

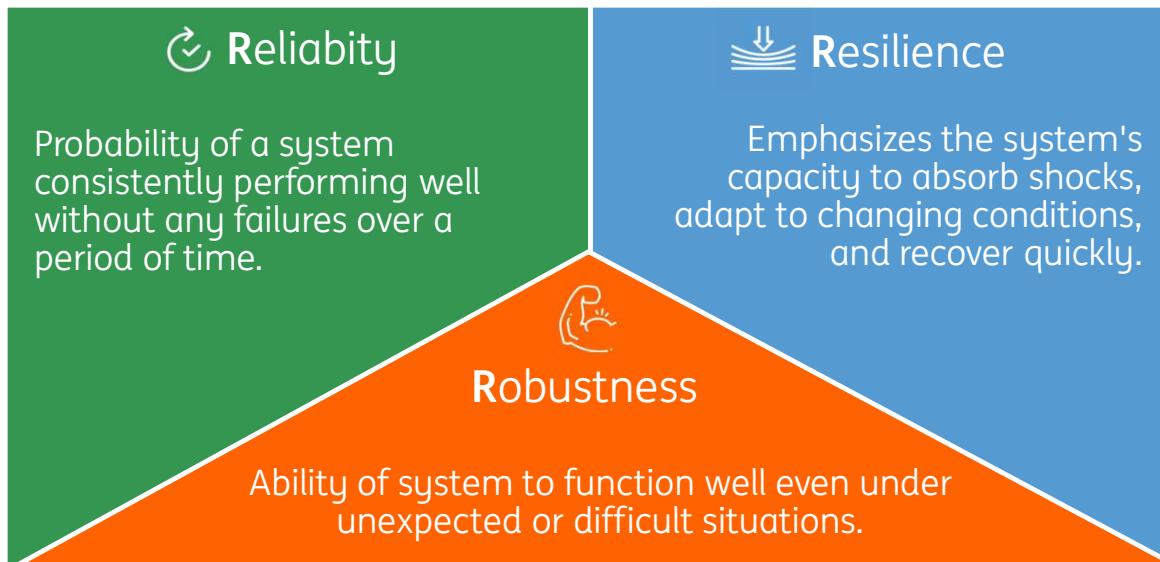
Channel Availability – Mobile & Internet target

- The prime-time availability target set at $\geq 99.78\%$ for BE & NL. → equals approx. **max. 75 minutes downtime per month**
- iDeal DNB target is 3 months weighted average $\geq 99.88\%$

H1 2023 results are clearly not in line with Targets for 2 out 10 Retail countries

- NL clients **received** promised levels **1 out of 6 Months**
- BE clients **received** promised levels **2 out of 6 Months**

Reliability and Resilience are often used interchangeably, however they actually address distinct aspects of a system's total operational effectiveness



Availability

Refers to the percentage of time that a system, or service, is accessible and operational.

Mean Time Before Failure

The total operating time of a system without failure.

Mean Time To Repair

Time to restore service to normal operations.

Achieving higher availability requires a systematic approach to the prevention, restoration, and resolution of failures and incidents

1

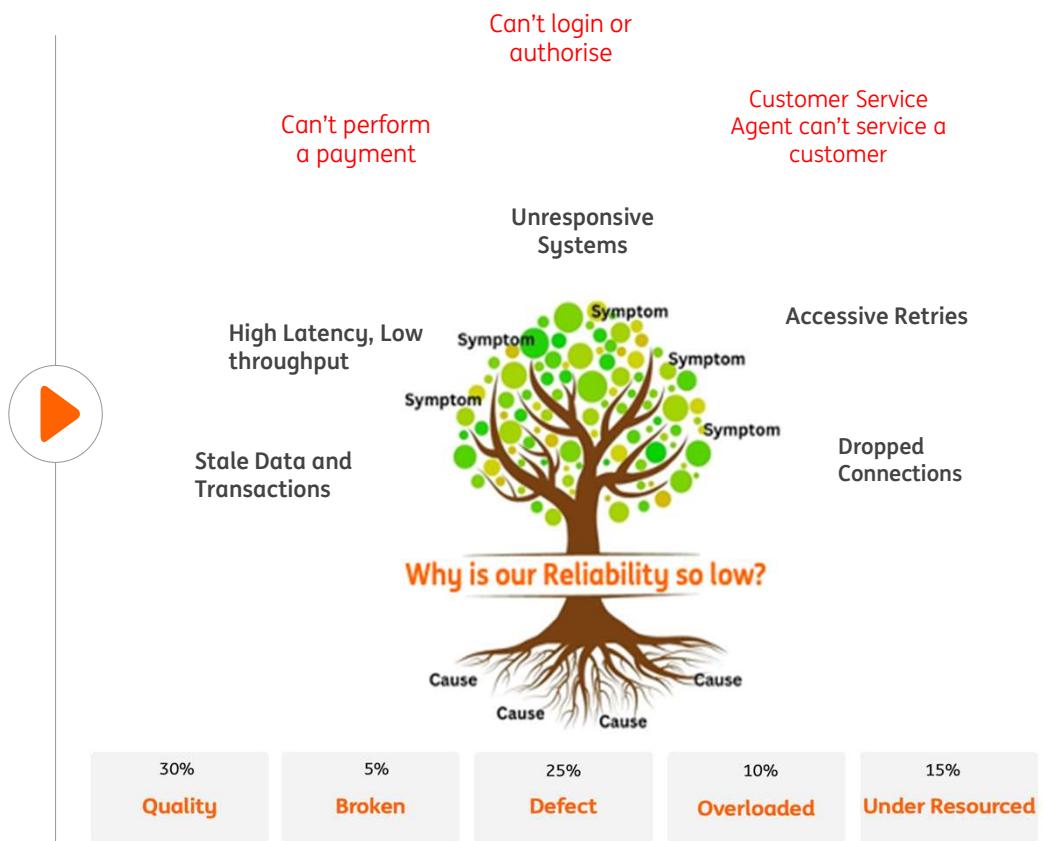
Prevent any **impact** from occurring, by following sound engineering practices. But remember: systems fail at some point in time.

2

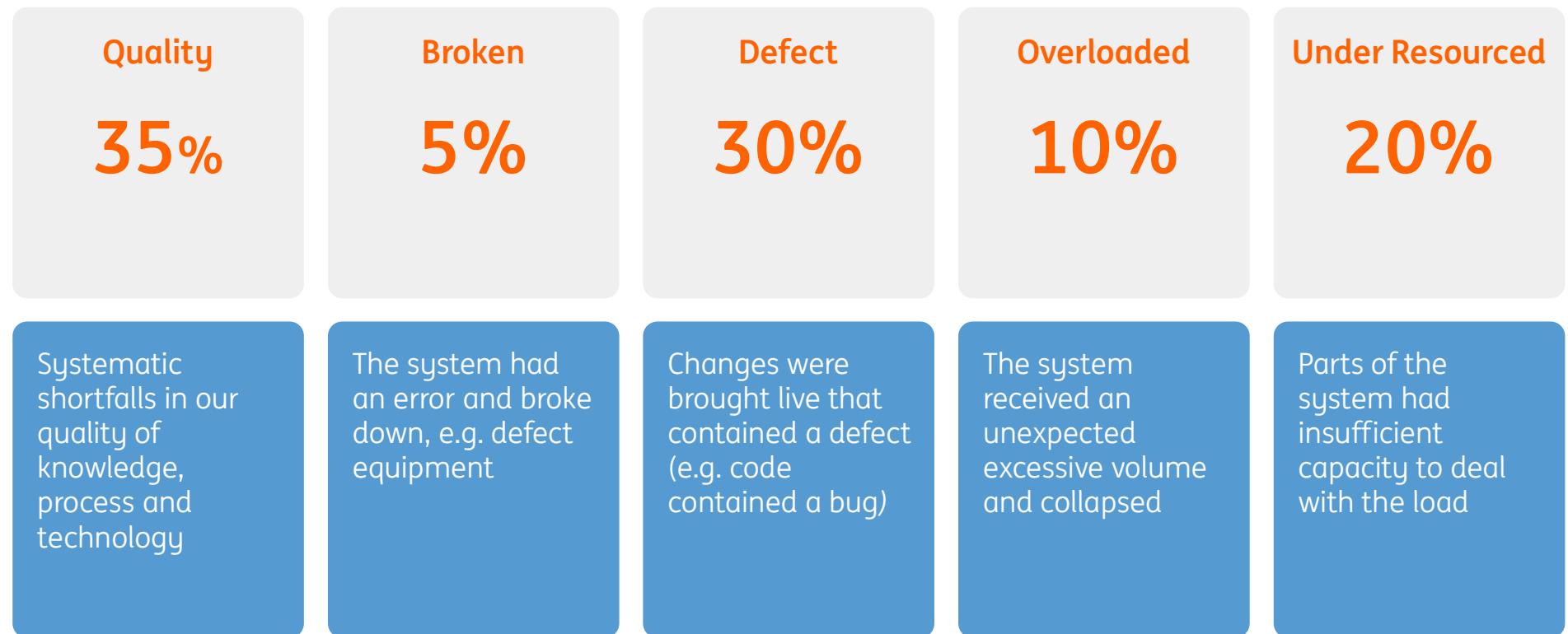
Restore service as fast as you can, address the **symptom** first and subsequently the cause.

3

Resolve take a systematic approach to identify the underlying **cause** and address it effectively, thereby preventing any future recurrence.



We conducted a comprehensive root cause analysis and identified five underlying causes that contributed to the outages



Reliability – Resilience : work in progress

IaC

Infrastructure as Code

PBA

Policy Based
Automation

Conformity bots

Resilience
embedded in
platforms/
processes

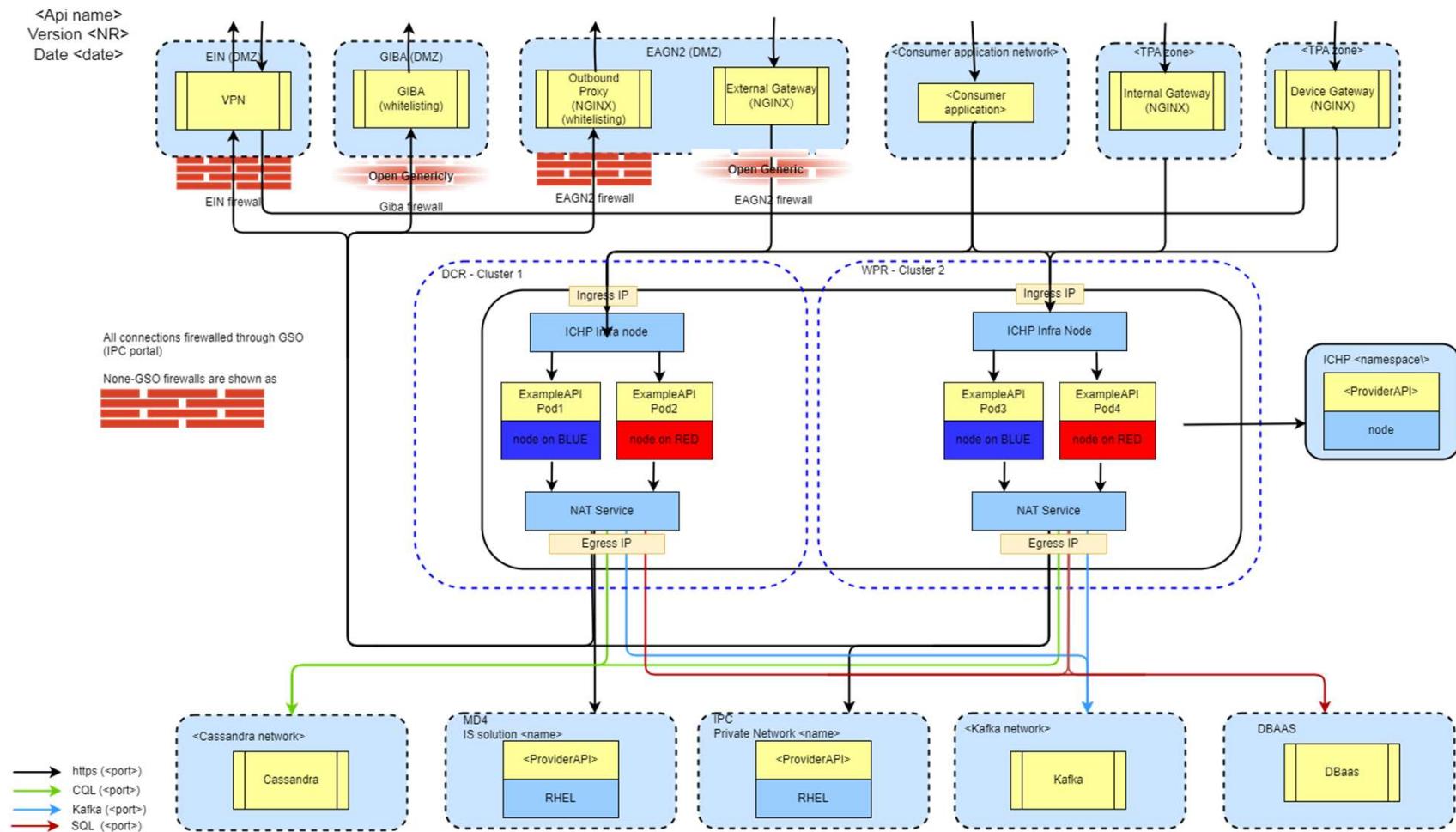
Resilience testing
/
Chaos testing

Self-healing

AIOps

Reduce Cognitive
load

Physical design picture





High Availability Basics

There are many aspects to High Availability.

- How do we reduce the number of visible **stoppages**?
- How do we reduce downtime during a **failure**?
- How can we eliminate **Single Points of Failures (SPOF's)**
- How can we turn an application into a true 24 by 365 application? The biggest single cause of downtime is **scheduled/planned downtime**.

But maybe the most important is how to ensure that there is **no data loss**

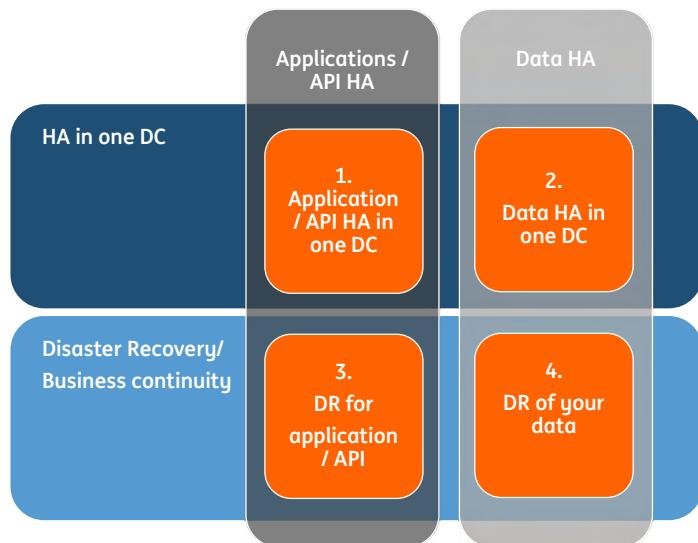


High Availability Basics

If you look at IT High Availability (HA) it is best to divide it in 4 parts based on the divisions:

Ha **within one DC** versus HA over two DC's (DR) (which is actually for business continuity)

Application HA versus **Data HA**

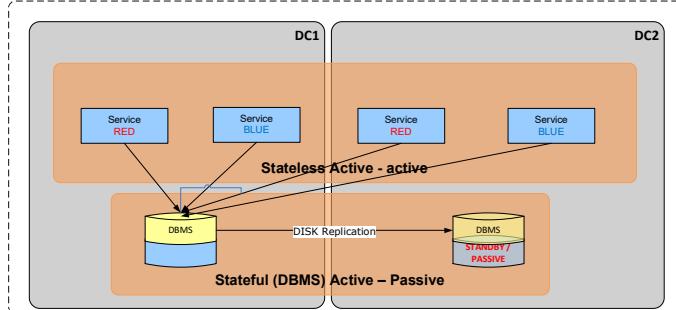


3

High Availability Basics

Using a backup system is the obvious way to build resiliency.

We recognize 3 different configurations of using backup systems:



Active – active

The primary and backup system handle the load parallel

Active – passive / cold-standby

Only the primary system handles the load. The backup system is not running and not processing any load

Active – passive / hot-standby

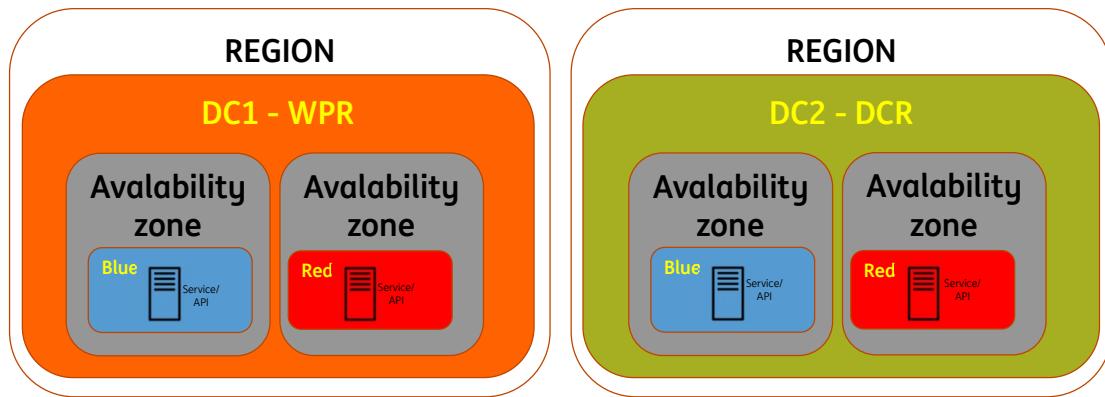
Only the primary system handles the load. The backup system is running but not processing any load

Advantages active-active architecture

The secondary system is proven to be operational and identical

Fast Failover

High Availability Basics

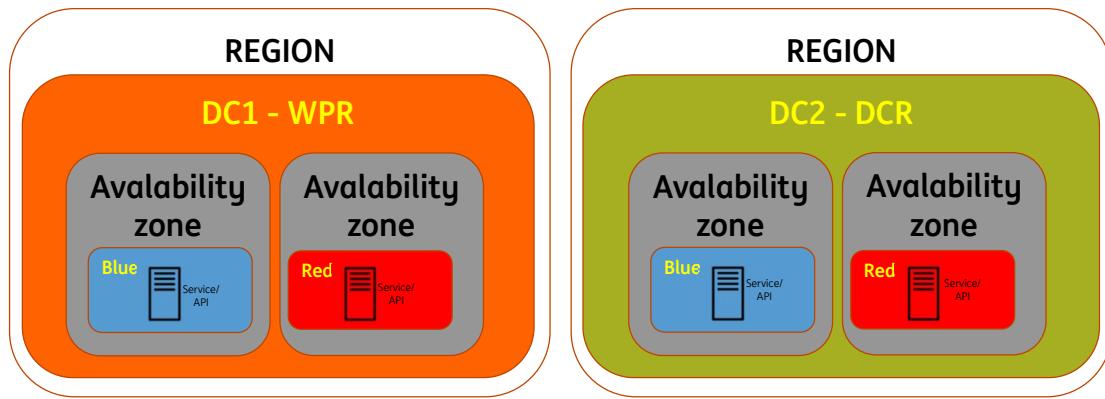


At ING a region is a datacenter (DCR and WPR).

An availability zone is a fault and update domain: different hardware is used within the same datacenter. It offers:

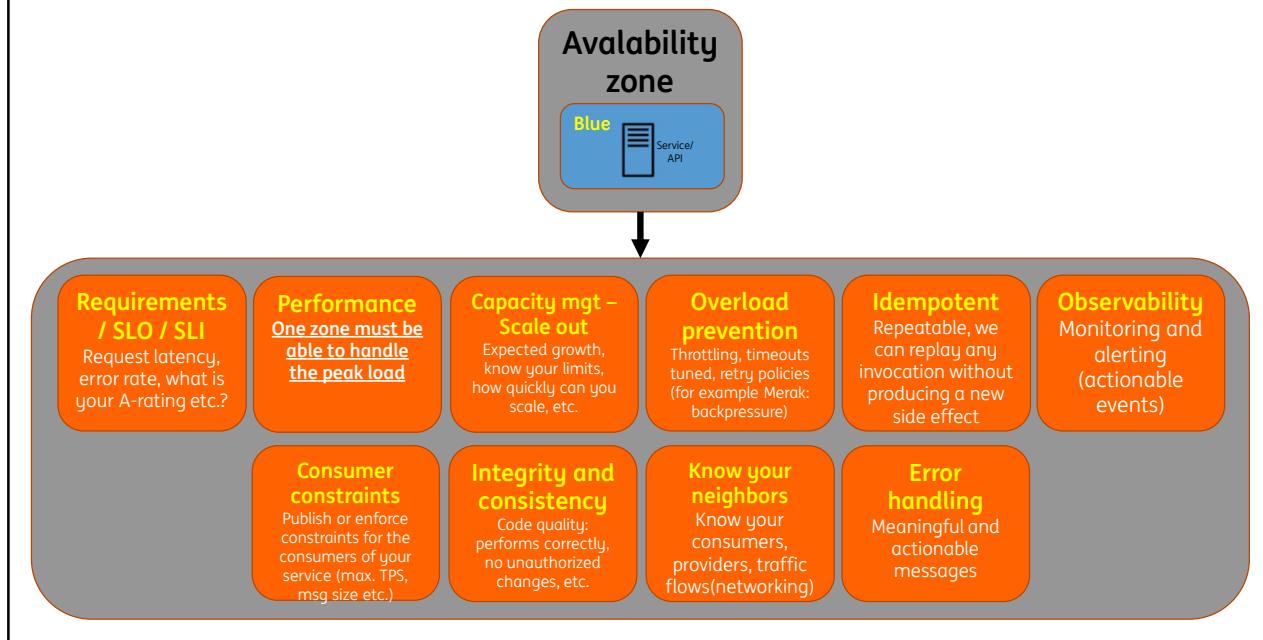
- Hardware failover
- The infra provider the ability to update the hardware (and middleware, for example VMWare) without business impact

High Availability Basics



This setup with the application deployed on two availability zones in each datacenter is mandatory for business critical applications (A3 and A4).

High Availability & Resilience per zone



An **SLI is a service level indicator**—a carefully defined quantitative measure of some aspect of the level of service that is provided. Most services consider *request latency*—how long it takes to return a response to a request—as a key SLI. Other common SLIs include the *error rate*, often expressed as a fraction of all requests received, and *system throughput*, typically measured in requests per second. Another kind of SLI important to SREs is *availability*, or the fraction of the time that a service is usable.

An **SLO is a service level objective**: a target value or range of values for a service level that is measured by an SLI. A natural structure for SLOs is thus $SLI \leq \text{target}$, or $\text{lower bound} \leq SLI \leq \text{upper bound}$.

Choosing an appropriate SLO is complex. To begin with, you don't always get to choose its value! For incoming HTTP requests from the outside world to your service, the queries per second (QPS) metric is essentially determined by the desires of your users, and you can't really set an SLO for that.

On the other hand, you *can* say that you want the average latency per request to be under 100 milliseconds, and setting such a goal could in turn motivate you to write your frontend with low-latency behaviors of various kinds or to buy certain kinds of low-latency equipment.

Choosing and publishing SLOs to users sets expectations about how a

service will perform. This strategy can reduce unfounded complaints to service owners about, for example, the service being slow. Without an explicit SLO, users often develop their own beliefs about desired performance, which may be unrelated to the beliefs held by the people designing and operating the service.

High availability & resilience

If you did not test it, assume it is not working

Processes

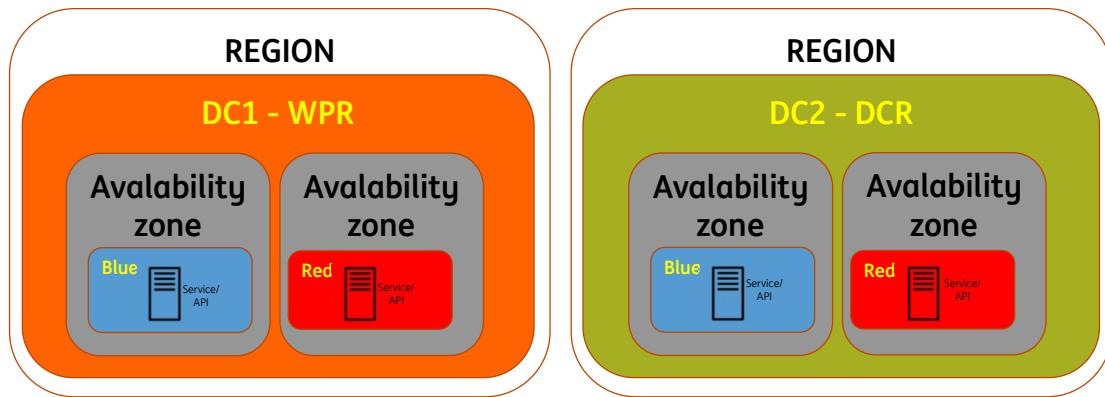
What about processes and the CMDB?

Incident
management

Problem
management

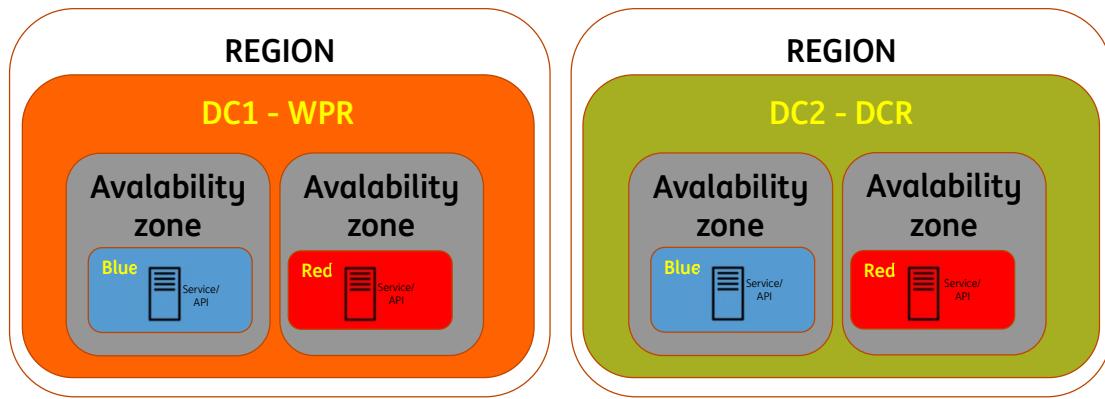
CMDB -
ServiceNow

High Availability Basics



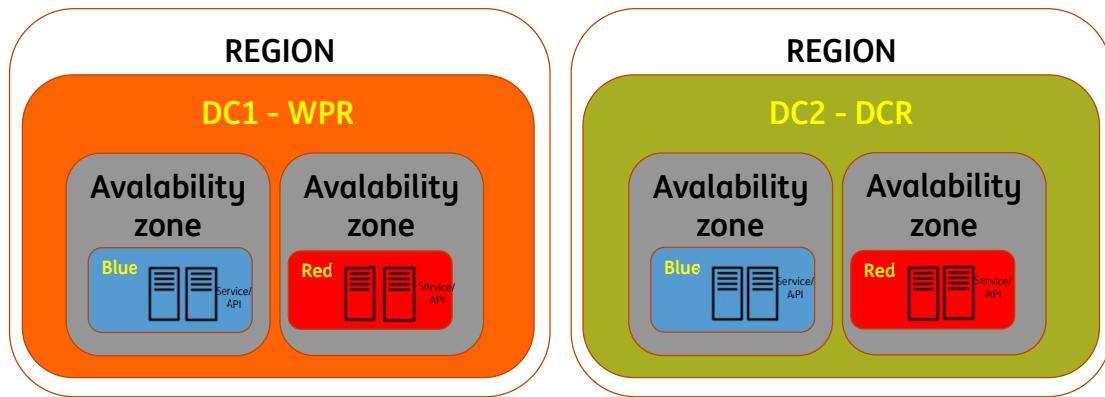
For resilience and high availability reasons business critical applications need in total
FOUR times the capacity/resources

High Availability Basics



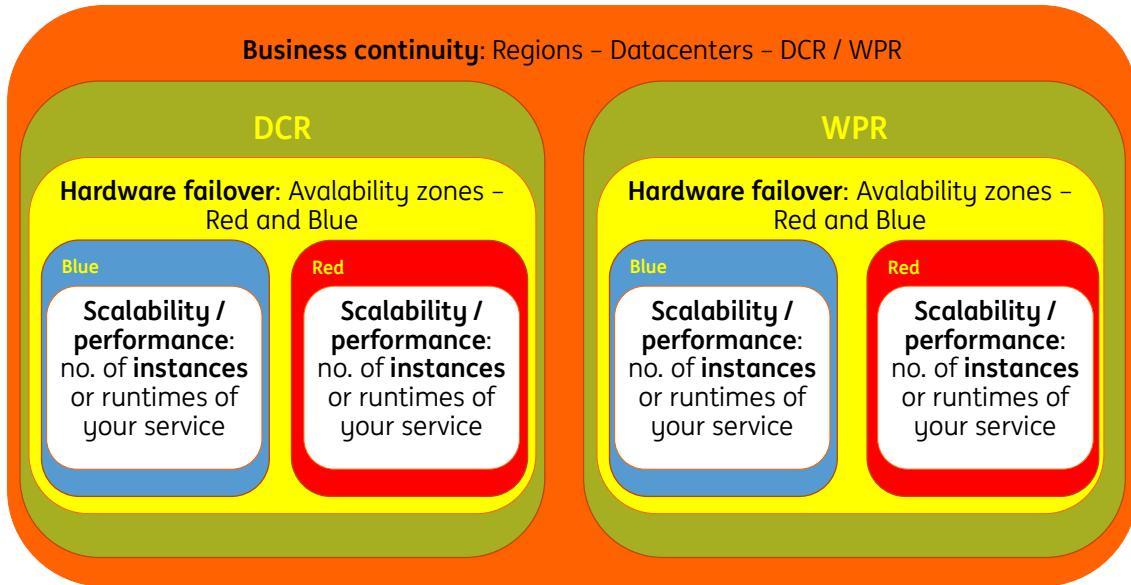
This is the setup for business critical applications (A3 and A4), if one runtime can handle the peak load.

High Availability Basics



If one runtime is not able to handle the load, you need to add a runtime to each availability zone (eight runtimes in total)

High Availability Basics: terminology

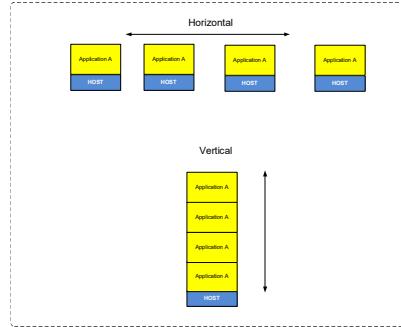


14



Clustering

Horizontal clustering	Vertical clustering
<p>Multiple machines/hosts with the same application, also for hardware failover</p> <p>In the context of scaling: adding more machines/hosts</p>	<p>Multiple application instances on the same machine/host, for process failover</p> <p>In the context of scaling: adding CPU/memory to the same machine/host</p>



2

Horizontal clustering

Two different (virtual) machines are used, offering Hardware/host failover.
BE AWARE: if hardware failover is the goal, the Virtual machine must be running on different hardware (anti-affinity) !

Vertical clustering

Multiple processes on one machine offering process failover or adding more CPU/Memory to one machine

If a failure occurs

A failure must be detected

Work in progress
must be managed
OR is lost

A failover must
take place

Detect the failure

Monitoring must be in place to detect the failure, for example with blackbox monitoring, performing healthchecks etc.

There are many monitoring tools available depending on the type of application and the protocols used.

Worst case scenario is when Ops becomes aware of a problem after the customer or end-user calls the help-desk...

Active monitoring opposed to passive monitoring is vital for detecting a failure early.

Work in progress

Data in memory is lost.

In a request-reply pattern:

- The requester must be able to resend the request
- The provider must be able to process the same request twice

A **microservice** must be idempotent: it **must** produce the same output if given the same inputs



5

Benefits of Idempotent

Resilient

We can replay any invocation without producing a new side effect: if error occurs we can simply retry the invocation

Parallel Processing

we can process multiple invocations in parallel. If one of them fails, we can retry it

Data Consistency

By guaranteeing **exactly one side effect** per unique invocation we can guarantee that our data is consistent: we don't have duplicate records or malformed data from incomplete invocations

Work in progress: stateful

When persistent data is in scope, when your application is **stateful** and you cannot accept the risk of losing data, extra measures must be in place.

Switching to the backup system means that all **data** on the primary system must be available on the backup system:

- Some kind of **data replication** (synchronous or asynchronous?) must be in place
- If your application is using transactional processing, the data AFTER LAST **COMMITTED TRANSACTION** must be available on the backup system

We will discuss HA of persistent data (for example a file, messages or DBMS) in a separate section.

7

We will look closer at data resiliency in the next chapter.

HA and Stateless

If your application or server is **stateless** high availability is **much, much easier**



Separate your data from your application (server)

Failover

If a failure is detected and a failover must take place, ask yourself:

- How quickly can the backup system takeover?
- How does the consumer handle the failover (and the first failure)?
- Are timeout settings configured correctly, also in your application chain?
- Is my backup system up-to-date (LCM etc.), identical to the primary?
- Can you handle loss of data in memory, are consumers and providers idempotent?

Error handling

Consider these scenarios:

You have a high available architecture.

So your primary system crashes, produces an error, and your secondary system takes over within milliseconds.

However your consumer stopped processing after the first error?

Or, let's say you are stateful and are using a database.

Your database is high available, it crashes, produces an error, but the secondary system takes over and no data (after last committed transaction) is lost.

However, the frontend application (the consumer) produces an error message on the screen of the end-user after receiving the DB error.

What will the end-user do?

High availability requirements

Having an HA architecture is not enough!
Always test the non-happy flow:
Resilience testing

Always test the non-happy flow



12



ACTIVE

Disaster Recovery:
active-active
@ING

Disaster recovery

For Disaster recovery at ING we have our two main datacenters DCR and WPR

Two options are available

- An active – active architecture (preferred)
- An active – passive architecture, optionally with ING **DR Capable** enabled (**NOT PREFERRED!**)

2

An active – passive configuration might be less complex and enables easier LCM and changeability on the secondary site.

However a big risk exists that the DR site is not in synch with the primary site. A change on the primary site might not be executed on the DR site and remains unnoticed until a takeover/DR test takes place. DR testing is crucial for the effectiveness of the DR site.

Active - Active

Many departments have a requirement to deploy their stateless API's **Active – Active**.

In this context we mean

- Your application is deployed in **two DC's** and both DC's are processing requests/transactions
- In each DC your application is **deployed twice**, also both processing requests/transactions

This means you have deployed in total 4 instances of your API.

We prefer this deployment for all API's, independent of your A-rating, for simplicity/consistency reasons.

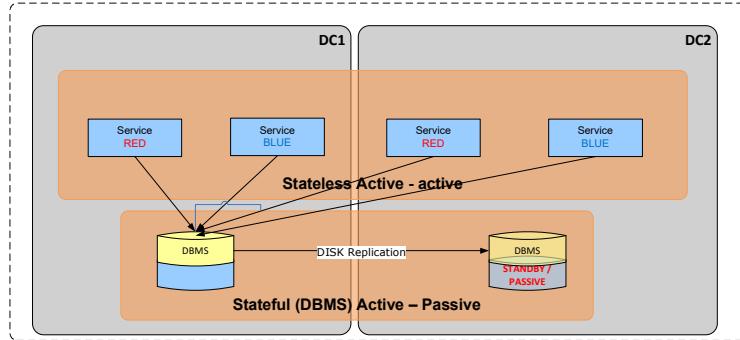
Having an active second DC has a big advantage for your **DR test is simple**: just disable the traffic to one DC.

Active - Active

But why do we need two instances in each DC?

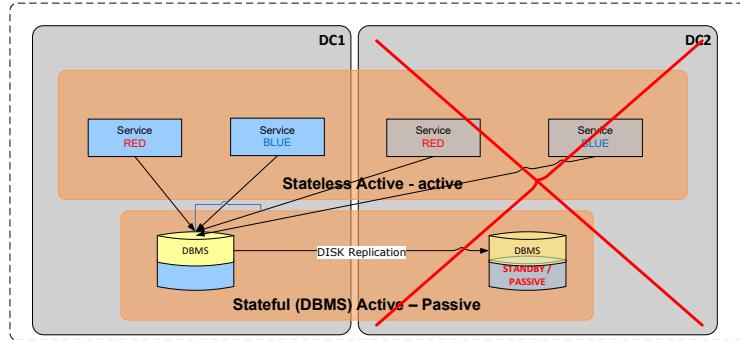
Why not one instance per DC?

Active- active, 2 - 2 pattern



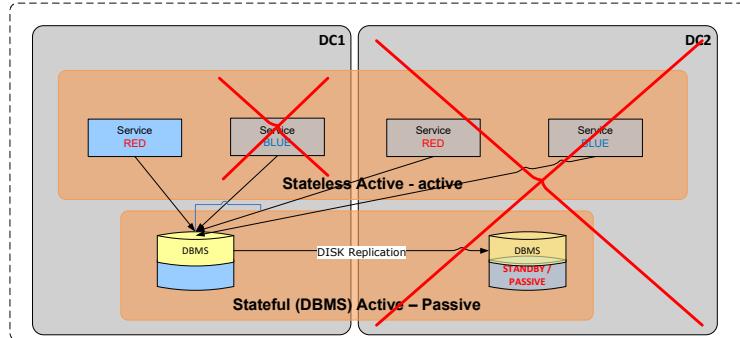
5

Active- active, 2 - 2 pattern



You are still HA and can execute LCM on your application during a DR test (twice a year for one week).

Active- active, 2 - 2 pattern



One availability zone must be able to handle all load!

Active- active, 2 - 2 pattern

You want to be HA
during a DR test

You want to be able to
execute LCM during A
DR test

The problem with Active – Active

If **Data Loss of Stateful** components/servers is unacceptable or must be limited, data must be replicated or a regular backup must be made.

For data **integrity** and **consistency** the data in both datacenters must be in sync.

For DBMS's like Oracle (DBaaS) it is very complex to have two active instances and ensure integrity at the same time. Therefore often the DB is deployed active-passive.

With **DBaaS** an **RPO of 5** minutes is offered.

The different stateful technologies (DBMS, Cassandra, XFB, messaging/Kafka etc.) will be discussed in another section of this training.

9

Depending on the qualification (RPO) some data loss might be accepted.

Cassandra supports an active-active configuration. However Cassandra is not a DBMS (relational DB) and data integrity cannot be guaranteed.

The problem with Active – Active

For stateful components often an active-passive architecture is used.

The question is: how do we keep our data in synch between the two datacenters?

- It is preferred data replication is managed by the application and the squad and NOT be dependent on your infra provider or a technology specific to one infra provider.
- Always separate your data from your application (server) as much as possible so your application can be deployed active-active over two DC's.

For example, use a **connectivity node**, a separate machine for your state (NFS server for XFB or MQ etc.)

Latency

For some applications using a Database (DBaaS) handling a high number of TPS, **Latency** must be considered.

However for most use cases this is not an issue: execute a performance test.

One Cross DC call can have a latency of up to 6 ms (IT DEPENDS...).

DC affinity can be configured but is not preferred: it adds complexity (and failover issues).

11

DC affinity is a mechanism where you control interfaces/data transport to remain within one DC.

The distance between the two DC's of ING, DCR – WPR is over 40 km and can be an issue in some cases for cross DC traffic

The 40 km limit is determined by technology: current fastest ethernet technology (light speed), to be explained in the last session of this course.

One DC can be disabled using configuration of LB's and NGINX or Application logic. **NGINX offers a dynamic function to disable one DC.**



Disaster Recovery: DR Capable @ING

What is DR capable?

When you order a virtual machine at ING you have the option to set:

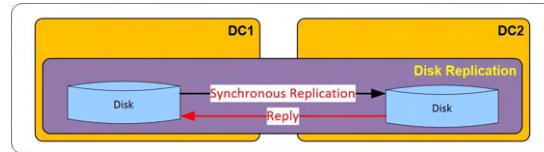
DR Capable = Yes

This means the infra provider will enable **data replication** at the disk level (used by your VM). A copy of your data and a copy of your VM is replicated to the other DC.

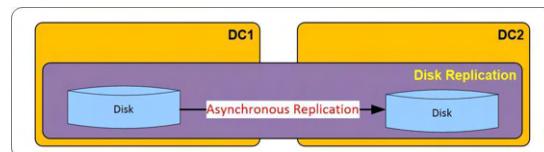
- In IPC this is a **synchronous** replication
- In many older environments it is a **asynchronous** replication

Disk Replication at ING

Synchronous replication decreases data loss but is a performance hit: processing continues **after** the confirmation is received. It might result to hanging processes in case of a disaster (synchronous replication must be disabled)...



Asynchronous: processing continues in DC1 **without waiting for a confirmation** from the disk in DC2: in case of a DR some data might be lost as the disk write action has not been confirmed.



What is DR capable?

When a DR (test) is executed:

Your VM is restored in the other DC with the **same hostname** AND in the **IPC** with the **same IP-address**.

In the old environments a DNS switch must take place as the IP-address does change.

As the IPC uses a different technology (stretched VLAN) you keep the same IP-address.



Never use IP addresses in your application

4

With a DNS switch we mean in the Domain Name Server the A-record relating the Host name to an IP address must be changed.

The DNS cache can be a problem during a DNS switch:

Application Servers keep using the old IP-address from cache.

Browsers (Chrome etc.) keep using the old IP-address from cache.

This will result in errors as the old IP address is not valid anymore.

Flushing the DNS cache is the solution.

Why not to use DR capable

You are not in control during a DR (test) and depend on the provider

It will take more time to restore your machine than with an active-active setup

Your VM is slower due to synchronous replication

Always consult your architect or BTA when you want to enable the *DR Capable* option



Future architecture

Requested Decision

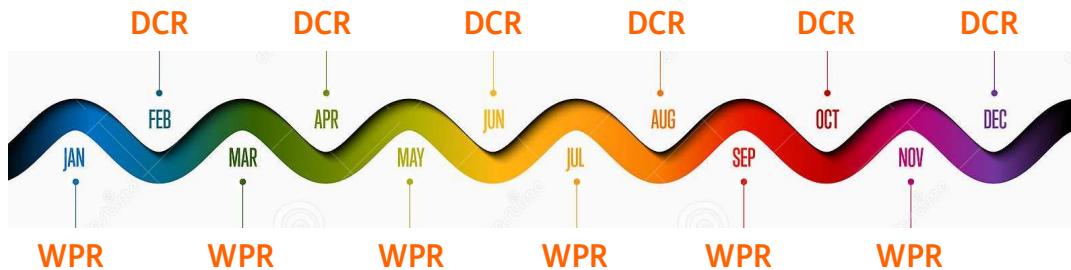
ING must move from "component based" failover of our channels to "**site based**" disaster recovery of our **channels**. (TikTok)

<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No

ING must progress towards an **DR testing schedule that alternates the active site** rather and stays there.

<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No

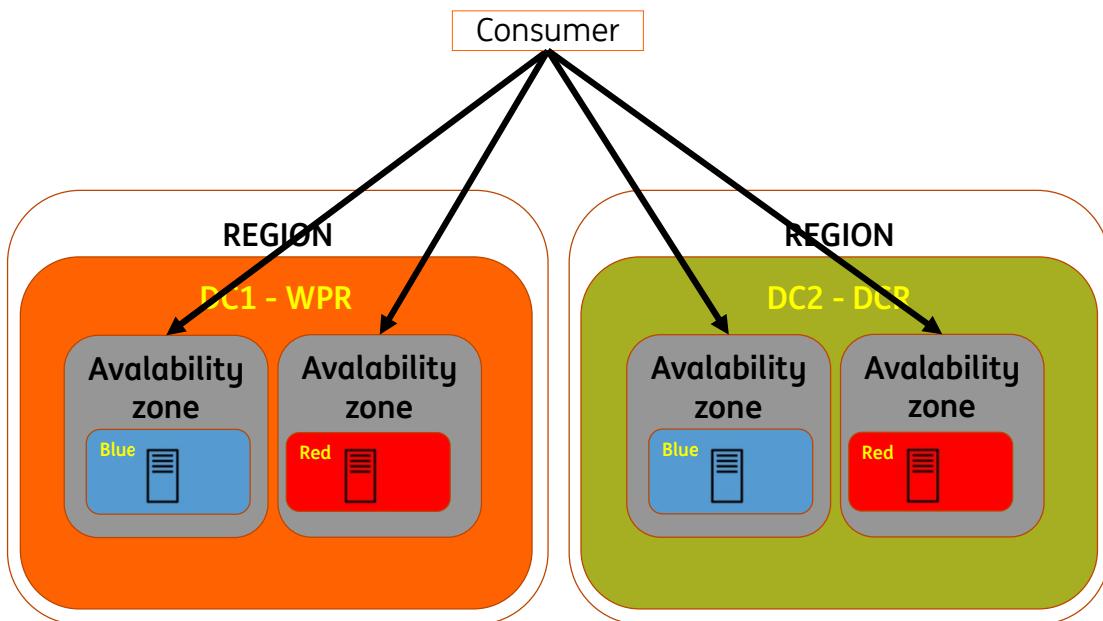
Disaster recovery will move from “DR testing” to “tik-tok”.



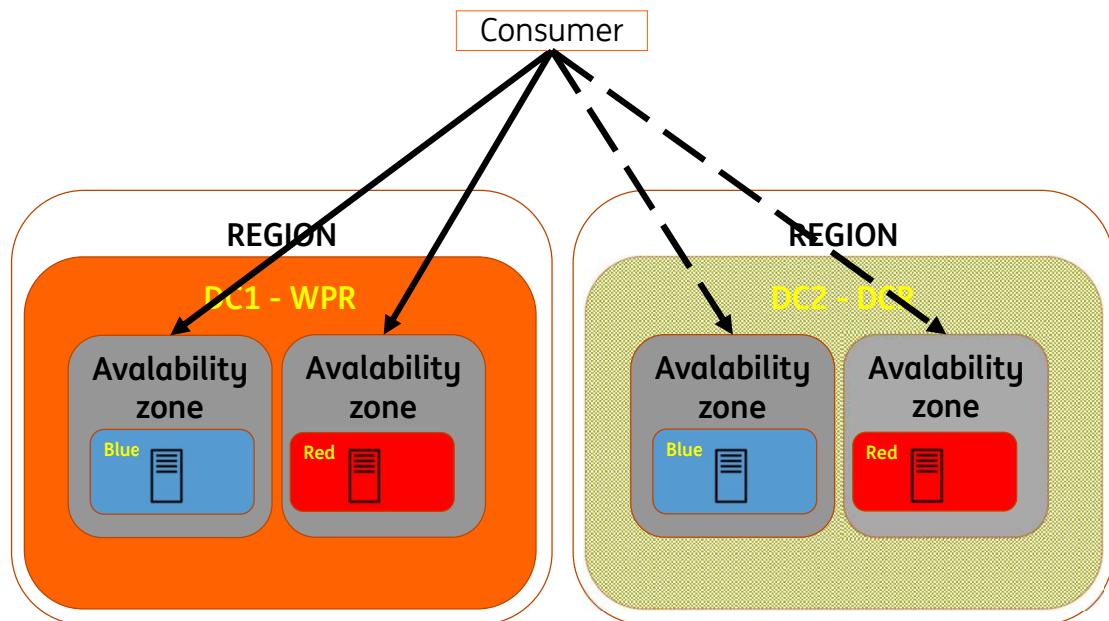
- The “active” site is a regular scheduled event.
- The entire site is failed over in a controlled way (i.e. it is not an application attribute)
- It runs for one month in each site.

3

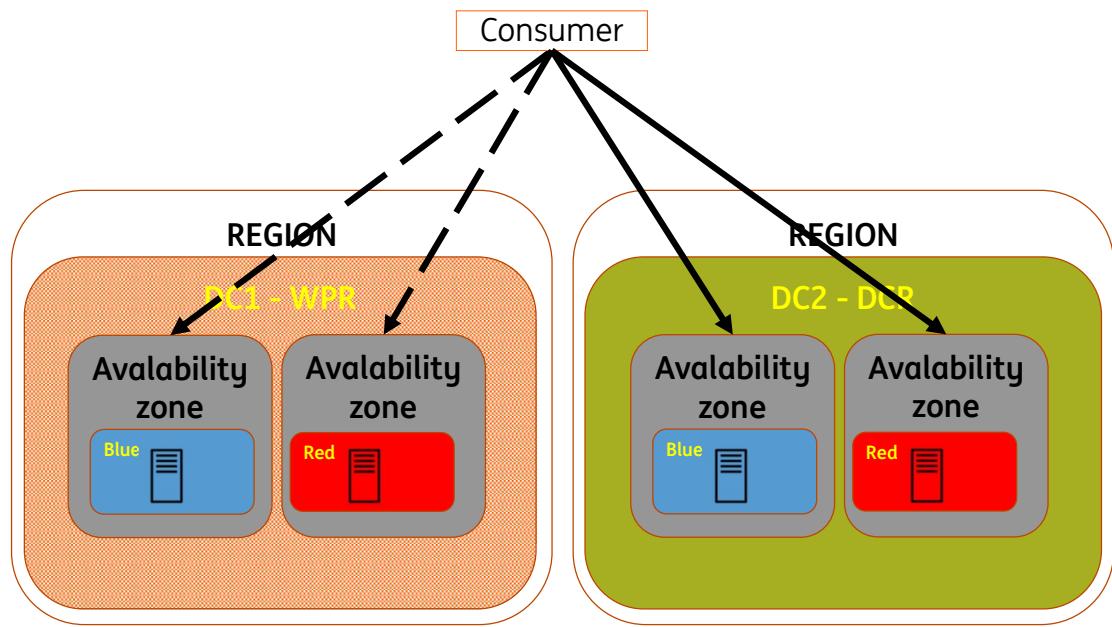
Active - active



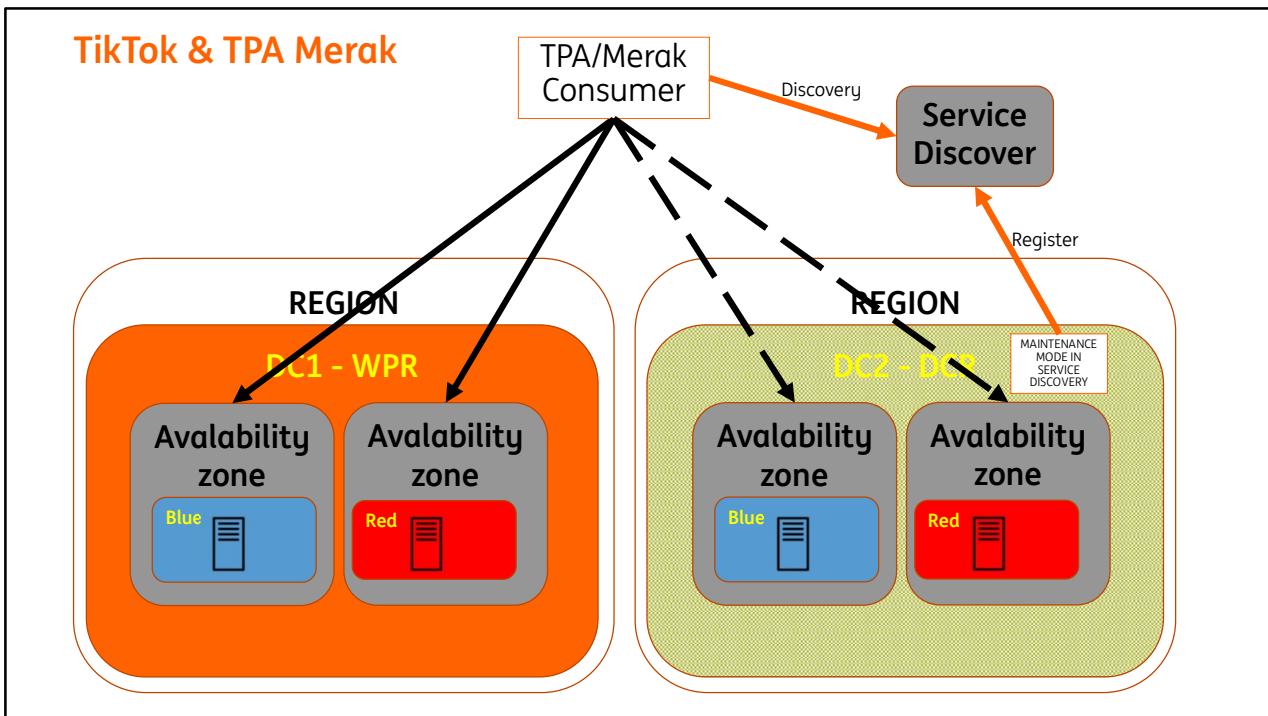
Tik



Tok



TikTok & TPA Merak



Active-active, 2 - 2 pattern

At the moment we are NOT actively moving yet from an active-active DC setup to an active-passive DC setup

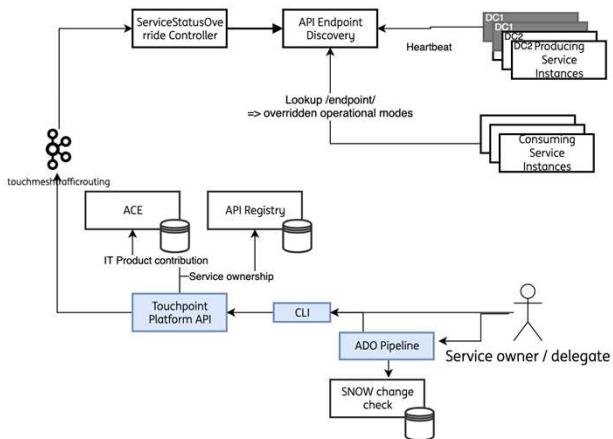
This would have a considerable impact and needs more preparation etc.

When your squad is in control of the incoming traffic flows, in control of which datacenter is active, you can migrate to this TikTok architecture, carefully.

Knowing about this (TikTok) target architecture:

1. Ensure you are **HA** in one DC (and one zone can handle the load)
2. For TPA/Merak: ensure you know how to use the **End Point Controller (EPC)**

Load to one DC: usage of End Point Controller (EPC)



Put your application in maintenance mode per DC:
change operational mode from 'live' to 'maintenance' using ServiceStatus Override.

Only prerequisite is that **API make use of Touch Point Automated Framework and Service Discovery!**

9

https://theforge.ing.net/product/45322/documentation/latest/consumers/automated_features.html#servicestatusoverride

<https://theforge.ing.net/product/45322/documentation/latest/index>



Requirements

High Availability
Requirements
@ING

High availability requirements

Building a resilient infrastructure requires a considerable investment. Each design choice must be confirmed by a **business requirement**.

Especially today it is important that the customer /business(stakeholders) performs a BIA, clearly state the requirements and perform a Risk Assessment.

We have several relevant formal requirements for availability:

- The **A-rating** in the **BIA** is originally meant for HA within one DC
- **RTO (Recovery Time Objective), RPO (Recovery Point Objective)** and **MOT (Maximum Outage Time)** are meant for disaster recovery.

2

The quantitative measures resulting from a **BIA** are the **CIA** ratings: a qualification of Confidentiality, Integrity and Availability.

The Availability rating is the requirement stated by the business which determines how resilient the application/system should be.

High availability requirements

Unfortunately, the implementation of
A-rating is NOT set in stone and depends on
interpretation, risk assessment etc.
Please consult your Solution Architect or BTA
when in doubt!

A-rating

At ING during design the Availability rating is a decisive factor:

- A1 - The service may be unavailable for 48 hours or more
- A2 - The service may be unavailable for 4 - 48 hours at maximum
- A3 - The service may be unavailable for 2- 4 hours at maximum
- A4 - The service may be unavailable for 2 hours at maximum

Another interpretation of the A-ratings:

- A1 – Single instance
- A2 – Two instances, second cold standby
- A3 – Two instances, second hot standby
- A4 – Two instances, active – active

4

A3 should also mean no SPOF's?

Building a resilient infrastructure requires a considerable investment (Cost).

Each design choice must be confirmed by a business requirement or an approval by the business.

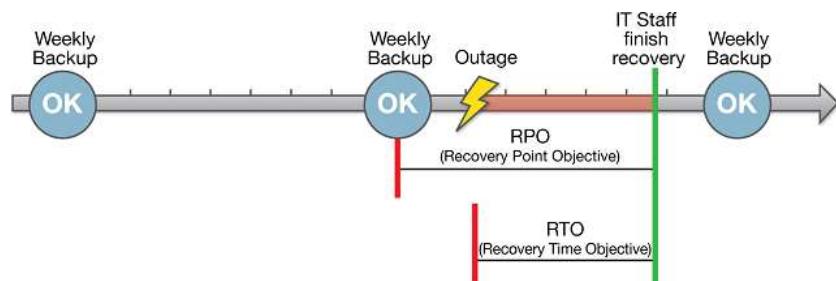
RTO - RPO

The **RTO** and **RPO** must be defined in the **Business Continuity Plan**.

But at ING these are also stated in de the **BIA**. RTO and RPO are the requirements for DR.

RTO: Recovery Time Objective, this is the time needed to restore a business process.

RPO: Recovery Point Objective, this describes the amount of data loss the business is willing to accept.



MOT

The **MOT**, Maximum Outage Time is the realistic value for the time between the disaster and the service being available again.

RTO does not take into account that a management decision has to be taken to failover to the DR datacenter.

The MOT includes all time spent between the exact moment the disaster occurred and the moment the IT service is available and up and running on the DR location.

MOT = RTO + time spent before IT starts the DR procedure



Network basics IPC



Network zone – Private network

TCP/IP networks are the most common type of network today. With such a network, a number of computers or nodes can communicate with each other. Each TCP/IP packet transferred over the network contains a source and destination **IP address**.

LANs (Local Area Networks) are networks usually confined to a **geographic area**, such as a single building or a college campus. A **VLAN** has the same attributes as a physical local area network (LAN), but it allows for end stations to be grouped together even if they are not located on the same network switch.

A logical grouping of nodes (computers) is also called a **network zone** or a **network segment**.

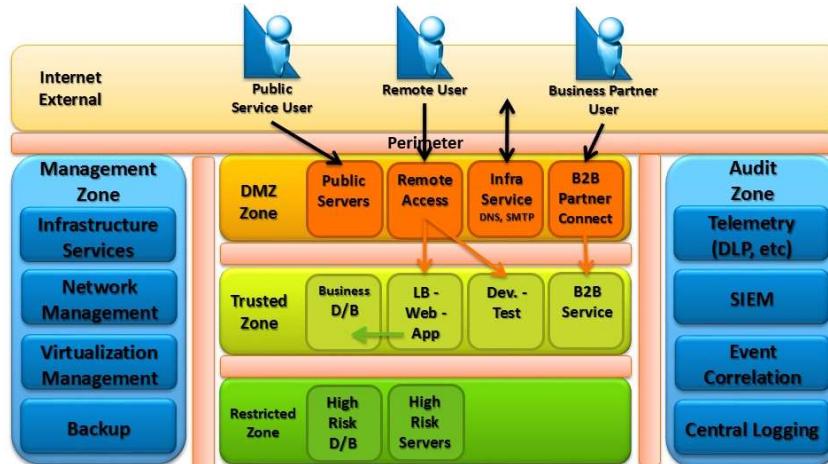
A **Private Network** in the IPC is a **network zone**.

Each network zone has a firewall to filter in- and outgoing traffic. WITHIN a network zone traffic is NOT filtered.

A network is a collection of computers or other devices, commonly called **nodes**, that are able to communicate with each other. This communication takes place on different network levels.

VLANs are created to provide the segmentation services traditionally provided by routers in LAN configurations.

The perimeter / castle-wall model



3

Traditional network security architecture breaks different networks (or pieces of a single network) into zones or segments, contained by one or more firewalls.

Each zone is granted some level of trust, which determines the network resources it is permitted to reach.

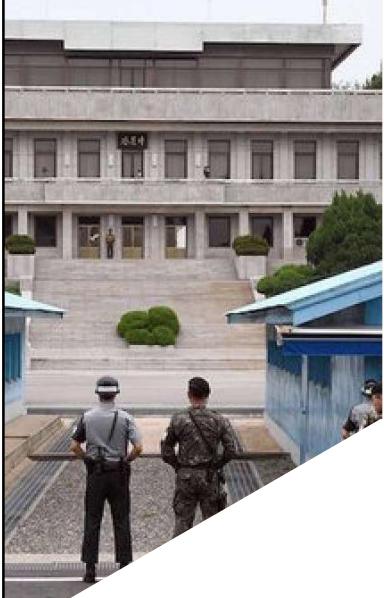
Zones were often divided in **external/internet – DMZ – Trusted – Restricted**.

Trusted and Restricted are **business zones**. If **persistent data** is stored in the business zone it is automatically treated as a **Restricted** zone (for the RCEC for example).

The **perimeter model**, after the castle-wall approach used in physical security.

In the old world, we had a castle with a door that was used to gain access. Securing the door was easy as we just needed one guard to secure the main door, one way in, one way out, nice and easy, right? But now in today's digital world, we have so many small doors and ways to enter...

Demilitarized Zone (DMZ)



A special **secured network zone**, connected with the **outside world**: the internet.

All **external facing** applications must connect through this zone (using a proxy like the API Gateway, XFB Gateway, Giba proxy etc.).

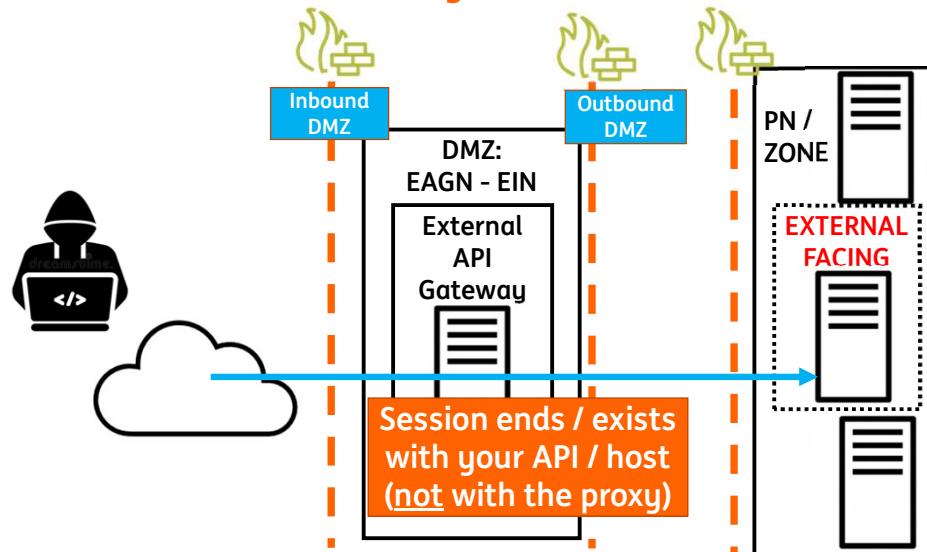
Machines/servers within the DMZ are **hardened**, meaning these machines are protected with special measures like security monitoring tools, intrusion detection tools etc.

What is external facing?

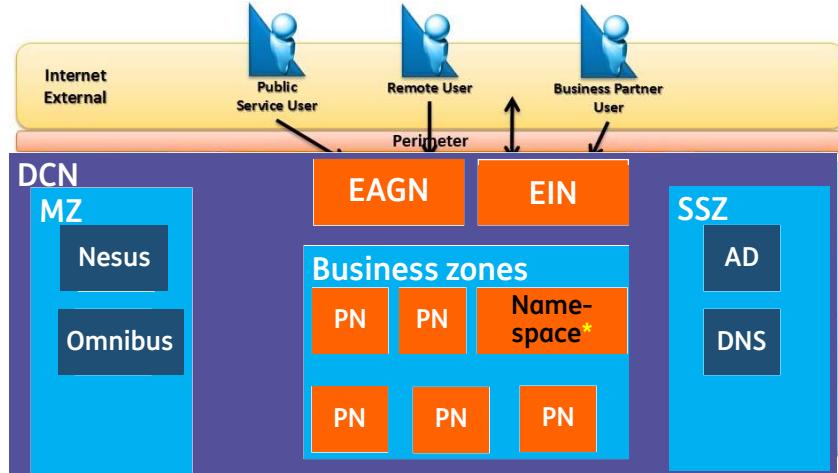
A connection between a non-ING governed IT environment (read: external party) and ING internal network is considered external facing.
This traffic must go through the DMZ: **EAGN** and **EIN**.
A connection from the ING internal network to our DMZ is also considered external facing.

Traffic coming from ING access zones (offices, ING intranet and ING user lan) are NOT considered external facing

Session between external and your API



The perimeter / castle-wall model



7

ING Network zones

Management Zone (MZ)	Supporting Services Zone(SSZ)	External Application Gateway Network (EAGN)	External Interface Network (EIN)
<ul style="list-style-type: none"> MZ hosts solutions aimed at <u>managing the ING IT-estate</u> for example with the purpose of vulnerability scanning. There is a general MZ firewall policy and a specific firewall policy to the solution can be implemented. 	<ul style="list-style-type: none"> SSZ hosts servers and appliances to <u>support ING applications</u> (AD, LDAP, DNS, DHCP, etc.) Every application hosted in the BZ should be able to establish a connection to the SSZ: <u>the firewall is default open for BZ</u>. 	<ul style="list-style-type: none"> Offers intermediary function between the internal and external network and the Internet. Typical DMZ workloads are reverse proxies (e.g. NGINX) with a north bound physical firewall and a south bound physical firewall. You need a RCEC to open the firewalls 	<ul style="list-style-type: none"> Used to connect to known Business Partners. The "known" means ING has some contractual arrangements in place to guarantee "good behavior". The connectivity is built using point to point connection using lines or VPN across the Internet. You need a RCEC to open the firewalls

The public character of the Internet makes the difference to implement a solution in the External Application Gateway Network (EAGN) or External Interfacing Network (EIN) building.

An example for EIN is a business partner is an insurance company doing business with ING exchanging data via a dedicated circuit. Service providers are for example Reuters- and Bloomberg feeds. Service providers could also be cloud providers, like Amazon Web Services (AWS), Azure and Office 365 (O365).

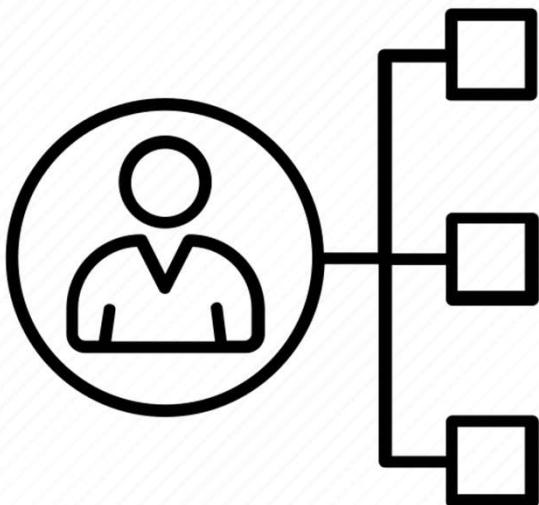
Business Zones (BZ)

Any zone where a DevOps team can deploy an application.

in IPC a business zone is called a **Private Network (PN)**, managed by the self-service portal.



Access networks (AN) - Userlan



Access networks are oftentimes called **Userlans (Local Area Networks)** in network architectures and are the network implementation at an ING owned or rented location(s).

A location could be an ING office, branch office, Point of Sale (POS) or an ATM.

We have different AN's depending on connection type (Wifi, Gras, LAN etc.) and location (country, branche etc.).

Firewall Basics



A Firewall acts as a gatekeeper and inspects traditionally the **5-tuple**:

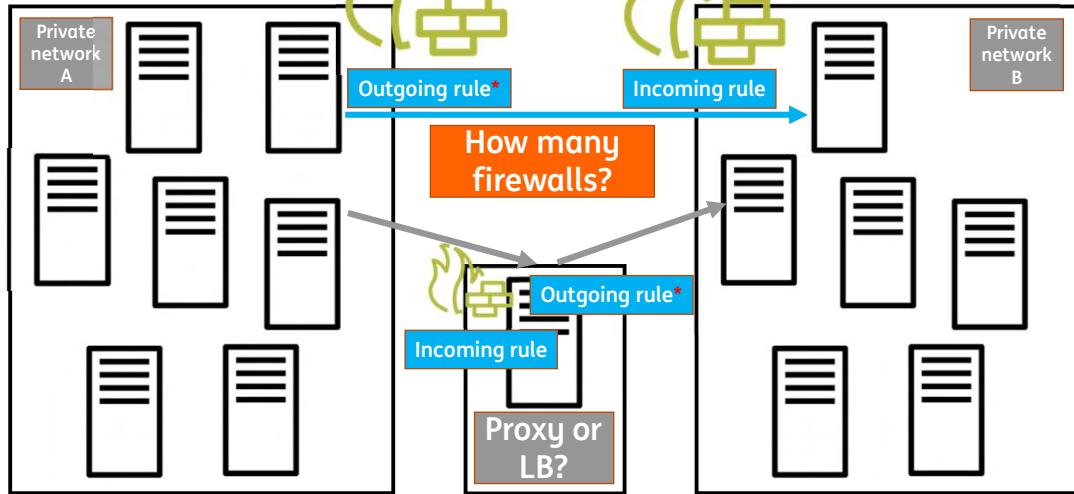
- source IP address/port number
- destination IP address/port number
- protocol

Firewalls can filter on **outgoing** and **incoming** traffic of a network zone.

Traffic within a zone is NOT filtered.

The firewall is using a **default deny policy** meaning if no **firewall rule** exists the communication is rejected.

Network Basics



How many firewall rules?

Even though it is one click or one request in the self-service portal of IPC, multiple firewall rules might be created



Application aware firewall

The virtual firewalls within IPC have Application Based Security i.s.o IP based security, meaning the firewalls are “**application aware**”.

An application aware firewall or next-generation firewall (**NGFW**) detects and blocks malicious traffic previous generations could not; it can decrypt, adds context (is **application** and user **aware**) and uses **DPI** (deep packet Inspection).

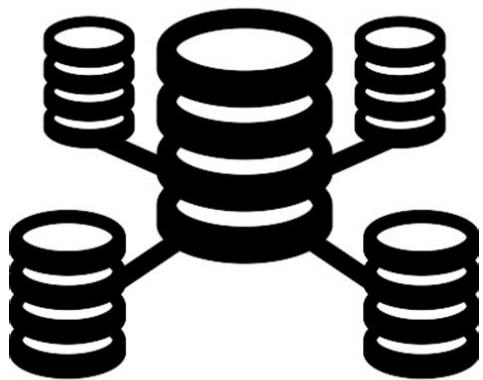
For example, if you want to allow Oracle traffic in the previous generation firewall you had to allow **port 1521**. In a NGFW you can define a rule to allow *Oracle*:

It won't be fooled by using a different port number, it recognizes the application!

14

Connectivity issues

**Always mention the source IP address,
destination IP-address, port and
networknames in communication
with network engineers**

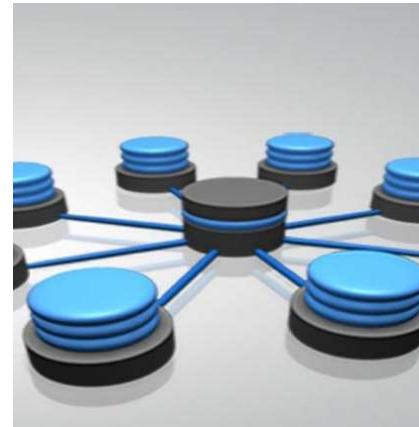


Data High
Availability
@ING

Data characteristics

Remember your data is your most **critical asset**. Your applications are just a means to access and manage your data. You must protect your crown jewels.

- Data has weight
- Data has value
- Data has importance
- Data has confidentiality, integrity & availability
- Data has lifecycle



2

Weight is related to **movement**: data with high importance and high volume has more weight than data with no importance and low volume.

Classifying your data is of utmost importance for governance and security purposes.

Enterprises can execute a BIA (Business Impact Analysis) to quantify the data.

Data weight

Example: AWS Snowball and Snowmobile...



Introducing AWS Snowmobile

- 45-foot long ruggedized shipping container
- Up to **100PB** of capacity
- Load data **S3** or **Glacier**
- Dedicated security personnel, GPS tracking, alarm monitoring, 24/7 video surveillance, and optional escort security while in transit
- Data encrypted with 256-bit encryption keys, managed through KMS



The size of the data is important for performance, and think of latency.
It is important for your choice of a datastore: DBaaS, SQLServer, Cassandra (KaaS), NAS (Network Attached Storage) or ECS S3 object store.

Data value: Confidentiality, Integrity and Availability (CIA)

At ING the value of our data is expressed in the **CIA** rating and **RPO**.

Confidentiality: Data cannot be disclosed by unauthorized individuals or systems

Integrity: The assurance of data accuracy and consistency over its entire life-cycle. Data cannot be modified undetectably; can you trust your data?

Availability: Data must be available when it is needed

And we have seen that the **RPO: Recovery Point Objective** describes the amount of data loss the business is willing to accept during a disaster (outage of one datacenter).

Preferred data services

To store business data several technologies are available at ING:

DBaaS (Oracle)
and SQLServer

KaaS
No SQL
datastores

ECS
S3 Object Store

NAS
Shared
filesystem

More services are available in the IPC, like *Redis* for example, but not preferred. Please discuss with your architect / BTA of your domain.

Which datastore should you use?

Databases: DBaaS and SQLServer

We need database software to **guarantee the integrity of the data**.

Using a **Database** (RDBMS) offers the following advantages:

- Use of **Keys** (indexes) to retrieve data and **search** the data
- Enables **relations** between the data
- Support of (exclusive) **locking**
- Support of **transactions: Rollback – Commit**
- Support for **SQL: Structured Query Language**



6

Nowadays when we talk about a DBMS actually we are talking about a RDBMS: a relational Database.

A Relational Database (**RDBMS**) is able to manage relationships between the data.

Often you have a database with a primary key and secondary keys.

For example a customer database, with a primary key of customer-id.

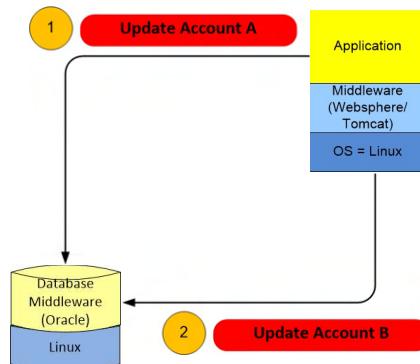
However you can search the data for all male customers, or customers with a birth date after 1980 etc. etc.

A relational database management system (RDBMS) is a kind of DBMS that was created in the 1970s and has become the dominant technology for managing databases since then. Its hallmark is a row-based table structure that connects related data elements to one another and provides transactional integrity to maintain data accuracy and consistency; another common RDBMS attribute is support for using the Structured Query Language (SQL)

Databases: Transactions

Happy flow for a moneytransfer
From account A to account B, NOT
using a Transaction

Account A and B are
Locked during the
Update.



7

In this example account balance is persistent data.

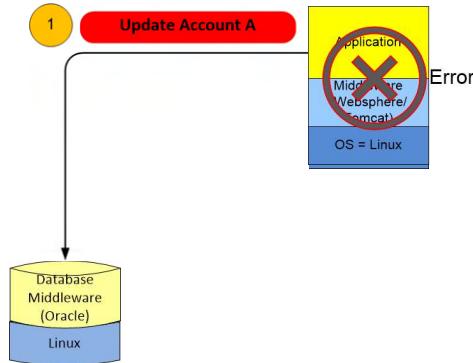
Databases: Transactions

Transactions

NON Happy flow for
a moneytransfer
From account A to account B, not
Using a Transaction:

Only account A is updated!

This is an integrity issue



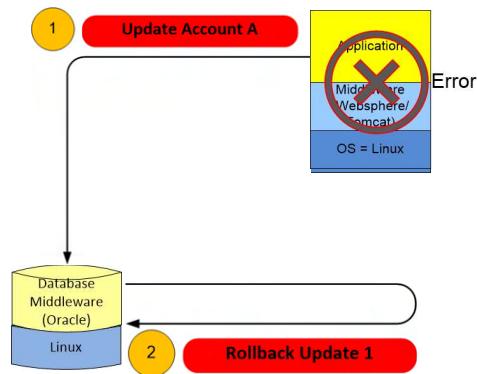
8

In this example account balance is persistent data.

Databases: Transactions - rollback

NON Happy flow using a Transaction:

Rollback executed by the database:
The data updated in action 1 is reverted / **restored** to the original state.



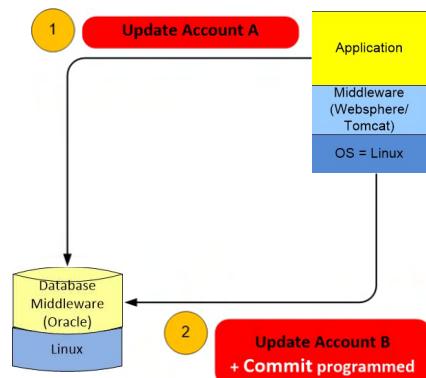
9

Application Server middleware like Tomcat/Websphere can also act as Transactional Component Middleware: The Application Server signals the DB to do a Rollback when the application crashes.

Databases: Transactions - commit

Happy flow for a moneytransfer
From account A to account B using
a Transaction.

The data change is finalised
AFTER both update 1 and 2
with the **Commit**



10

In this example account balance is persistent data.

SQL vs. NoSQL

It is much more difficult to guarantee the integrity and consistency of your data if you use a No-SQL datastore like KaaS (Cassandra), or ECS S3 object store.

For business data that is a SoR (System of record) use a database: DBaaS or SQLServer Always ON.

A SoR, system of record is the **authoritative** data source for a given data element or piece of information. It is the master data and represents the “truth”.

As we will see later, you can use a No-SQL data source as a “cache”: a copy of the SoR.



One purpose (p-code), one datastore

A datastore is considered **tightly coupled** to one purpose / p-code. Coupling typically refers to the degree to which software components/modules depend upon each other. **Tight coupling** is when an application or service is highly **dependent** on one another.

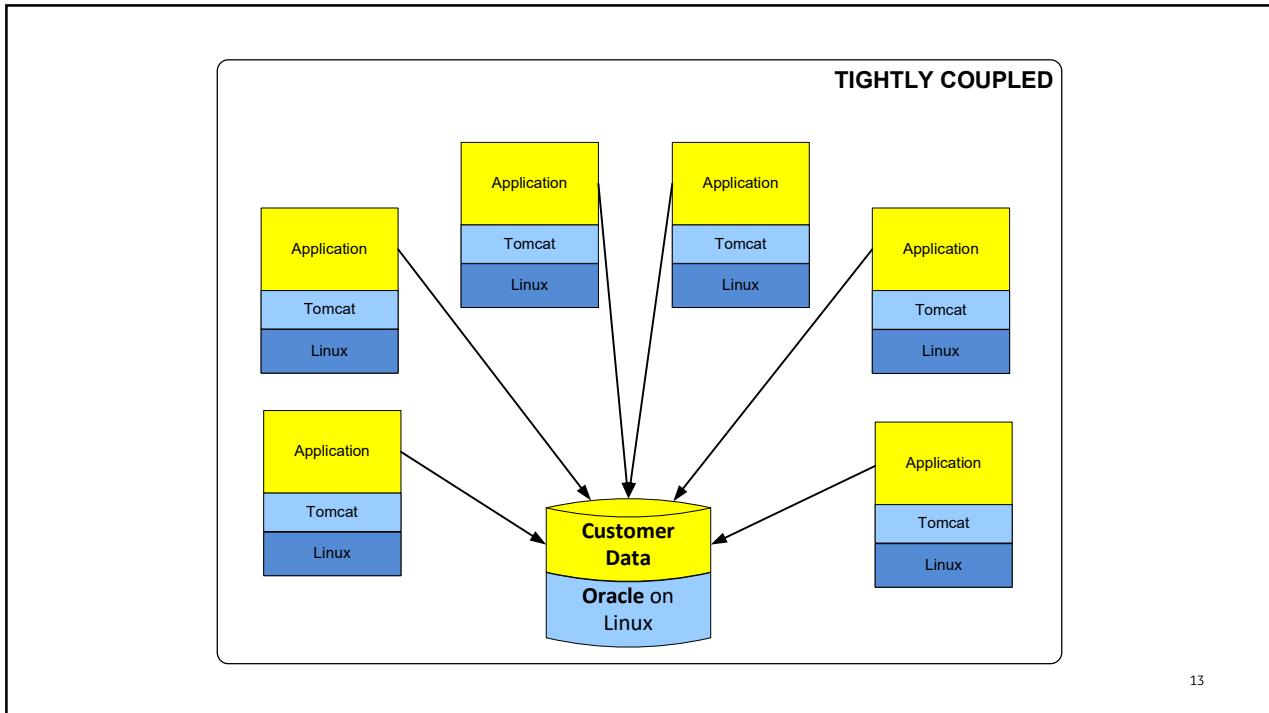
Loose coupling is achieved by means of a design that promotes **single-responsibility** and **separation of concerns**.

If a datastore is shared with multiple purposes:

- A change of the data structure leads to changes in multiple applications (in different squads and area's)
- A change of technology leads to changes in multiple applications (in different squads and area's)
- LCM must be coordinated with multiple applications (in different squads and area's)
- Responsibility and ownership might be unclear
- Load (performance/availability) is very difficult to manage



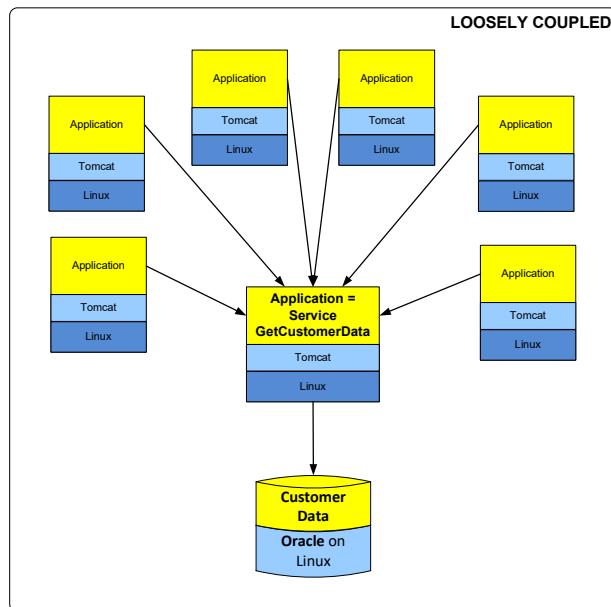
This dependency between squads and areas hampers our changeability and thus our agile WoW.



13

What if:

1. You need to upgrade Oracle?
2. You change technology, from Oracle to MariaDB?
3. The network location changes?
4. A failover takes place: are your requesters configured correctly?
5. The data model changes, need all requesters modification?



14

Data design principles

For business data that is a SoR (System of record) use a database: DBaaS or SQLServer Always ON.

One datastore (DBaaS, SQLServer, KaaS, ECS S3 and NAS) can be used by only one purpose/p-code

Agenda Data High Availability

Availability vs.
Integrity

Backup
considerations

RPO and zero
data loss

Stateless vs.
Stateful:
separate your
data

Commvault

DBaaS &
SQLServer
(DBMS)

KaaS (NoSQL)

ECS S3 Object
store

NAS
shared file
system

Archiving

1

Data Resilience

In IT we tend to focus on application or middleware availability. However, the most important thing is **zero data loss** and **data consistency and integrity**. The application is simply a means to read/manage the data.

Data resilience is about two things:

**Data availability:
backups**

**Data integrity: is
the data correct
and trustworthy?**

2

Data Integrity is the maintenance of, and the assurance of the accuracy and consistency of the data.

Data Integrity & backups

One of the worst things that can happen is that your data is corrupted. This is commonly caused by bad programming code. There even might not be a failure or error...

This is another reason why **backups** are always necessary.



But is there a guarantee that
the last backup is not
corrupted?

And how many backups, how long do you keep your backups? What is the **retention period**?

The **I** of your CIA-rating describes the integrity required.

Logical versus technical failure

Logical
failure

Technical
failure

4

Data Integrity is the maintenance of, and the assurance of the accuracy and consistency of the data.

Data Integrity

It is of the utmost importance that programmers write code that:

- Prevents bad data getting in the database. **Input validation** is essential.
- Look for corruption in the database as it is being read. A **regular integrity check** might be coded.
- Display as much information as is possible about any errors detected.

You need:

An **input / audit log** of all important input transactions. Who was the user, what transaction was he/she doing, what data was entered?

You must be able to relate the input (log) to the transaction in the database.

5

In a distributed environment you must rethink your strategy for error reporting.

Backups

The RPO (Recovery Point Objective) describes the amount of tolerable data loss

Creating a backup of your data is the obvious way to make your data highly available.

In general three basic backup schemes are possible:

- A full backup
- An incremental backup
- A differential backup
- Continuous (or real-time) backup

Depending on used technology different terminology and mechanisms are available.

6

1. A full backup: a complete backup of all data
2. An incremental backup: saves only newly created or changed data since the last backup, regardless of whether it was incremental or full. Restoring incremental backups can take a long time when the full backup was a long time ago: the full backup and several incremental backups must be loaded
3. A differential backup: saves only newly created or changed data since the last **full** backup. It will take more storage since each backup stores all changed data since the last full backup. However restoration is faster
4. Continuous data protection, also called continuous backup or real-time backup, refers to backup of computer data by automatically saving a copy of every change made to that data, essentially capturing every version of the data that the user saves. It allows the user or administrator to restore data to any point in time

Considerations backup - restore

An important factor to consider is the **performance** of backup and recovery.

- How much time is needed to create and/or load a backup?
- How is performance affected during a backup?

The time needed to restore probably means downtime (RTO?):

Your data is not available, therefore your service/application is not available.

Size of the data?

Technology used?

Backup scheme used?

Synchronous or asynchronous?

7

The performance of recovery determines the frequency of creating backups and which kind of backup.

Many products offer *incremental* backups: backup only the data changed since the last backup.

Ransomware & backups

Ransomware attacks now account for over half of all malware attacks (51% in Q3 compared to 39% in Q2 in 2020)

For the most critic hardware (IaaS) le

The DarkSide ransomware threat (*Colonial Pipeline attack*) ransomware-as-a-service model that DarkSide used very successfully

Ransomware & backups

*Ransomware attacks now account for over **half of all malware attacks** (51% in Q3 compared to 39% in Q2 in 2020)*

The **DarkSide** group (**Colonial Pipeline** attack) offers [ransomware-as-a-service](#). Ransomware is big business!

Ransomware & backups

Create backups to mitigate ransomware attacks

Place your backup in another network segment (network segmentation): default for Commvault and DBaaS

RPO Zero?



Zero Data Loss is very hard to guarantee

Even our most important data store, DBaaS – Oracle, offers a RPO of 2 minutes. Meaning you might lose 2 minutes of transactions... What are your TPS?

For the most critical data use [multiple solutions](#) on the application, middleware (PaaS) and hardware (IaaS) level.

Mitigate the risk of data loss: solve it in your application

Keep an input
log in the top of
your chain

Write your
own integrity
check

Write to
multiple
datastores

Test, test, test

Data High Availability

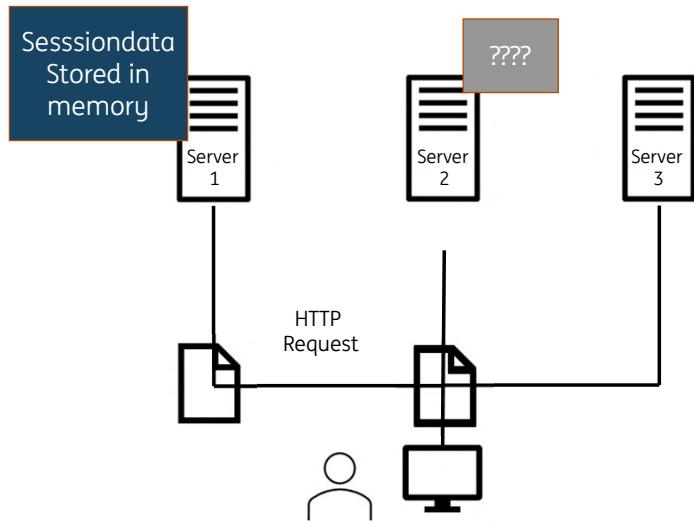
We can divide data in three categories:

Transient data (in
memory or in the
network)

Persistent data on
disk

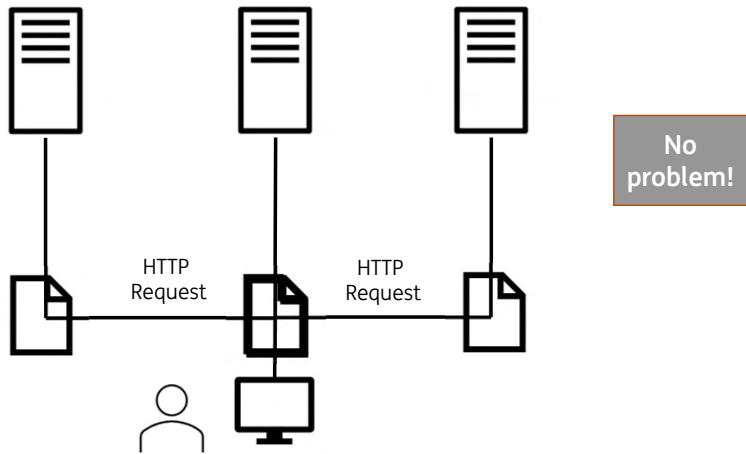
Persistent data
managed by a
special middleware

Stateful sessions example



14

Stateless



15

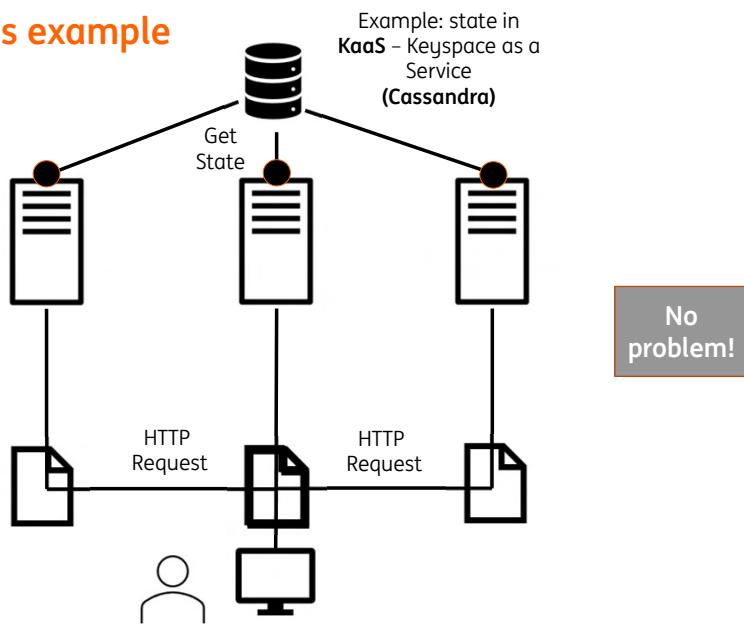
HA and Stateless

If your application or server is **stateless** high availability is **much, much easier**



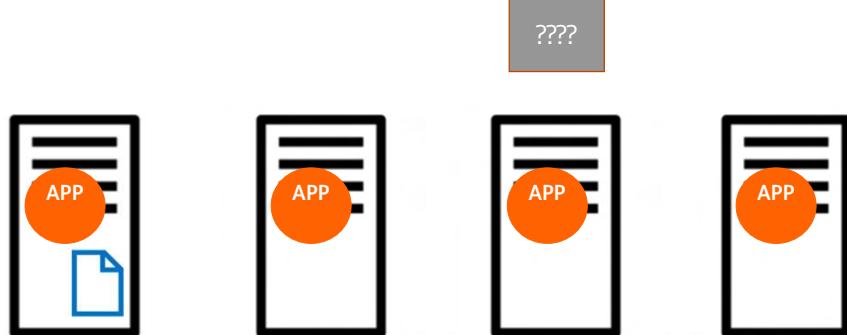
Separate your data from your application (server)

Stateless sessions example



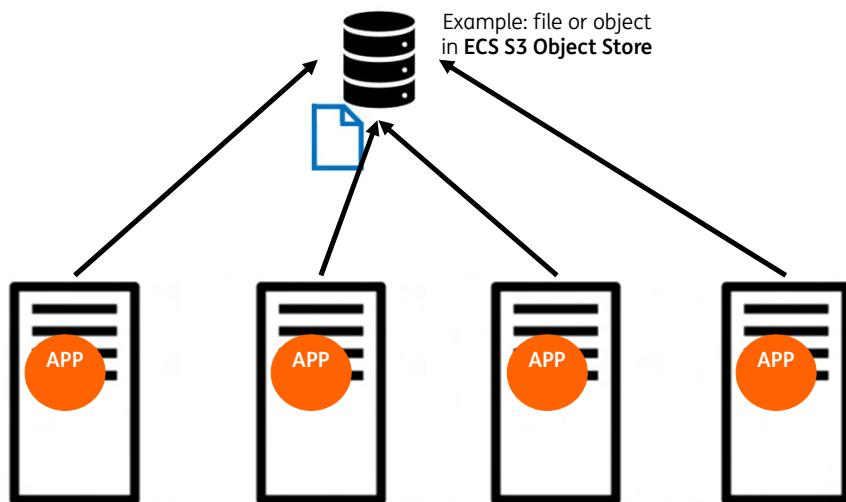
17

Stateful – files example



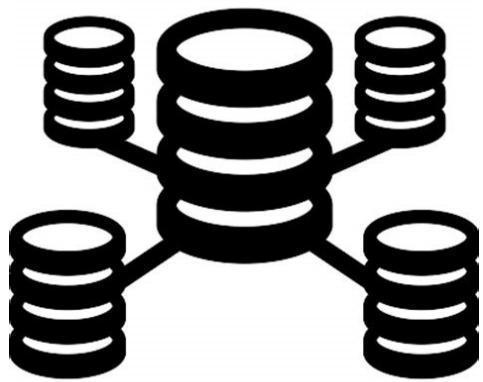
18

Stateless – files example



19

Store the data file for example in the Object Store. Or on a separate machine and copy the data to the application servers (then you also make the data HA!).



Data High
Availability
@ING

Non-persistent data

Data in memory, **non-persistent** data, is usually **lost** during a failure.

- The **request-reply pattern** often uses non-persistent data: the requester must be able to **reproduce** the request.
- There are (Java) application libraries to replicate data in memory, however that is not recommended.
- Another application option is to persist the data in a fast NoSQL datastore: **KaaS** at ING, Keyspace as a service (Cassandra).
- On the Infra level a technology called **Vmotion** (VMWare) is used to move VM's **including memory**. This functionality is not available as a self service for the consumer.

Preferred ML data services

For **persistent data** several technologies are available / preferred within ML:



More services are available in the IPC, like *Redis* for example. Please discuss with your architect / BTA of your domain which technology is preferred.

Messaging (Kafka / MQ / Tibco EMS) and filetransfer – XFB, to be discussed separately.

3

Non-persistent data is usually lost: the requester/user needs to resend his request.

On the Infra level a technology called Vmotion (VMWare) is used to move VM's **including** memory. This functionality is not available as a self service for the consumer.

Because you can, in the IPC portal?



Because you can, doesn't mean it
is allowed (or preferred)



A snapshot preserves the virtual machine just as it was when you took the snapshot - the state of the data on all the virtual machine's disks and whether the virtual machine was powered on, powered off or suspended.

Backup using Commvault for SAN – SSD – SQL

In the IPC the data of your VM and your attached storage (SAN) is stored on (fast) SSD.

Commvault is the standard tool used to manage your backups in IPC for your VM's, attached storage and SQLServer.

Backup is NOT enabled by default

In-guest backup can be configured

Maximum is 8 TB, with backup 6 TB (?)

Restore only to the same VM

If you use more than approx. 4 TB and you have enabled a full backup, please test carefully!

With backup enabled 2 TB (of the total 8 TB) is reserved for the backup process.

If you exceed the 6 TB or this 2 TB during the backup process there is a risk of an unrecoverable server: your backup is also corrupt: you have lost your server...

This depends on the performance of the backup and how dynamic your data is changed during the backup.

5

Second day operations Commvault

Schedule your Backup

Set retention to
31 or 14 days

Set SLA from
Platinum to
Gold (without
replication)

Suspend and
Resume your
backup

Backup – Archive?



A backup with Commvault is NOT
meant for Archiving!

Snapshot versus backup with Commvault

In the IPC two self service functionalities are available:



Keep in mind a snapshot is not a backup. Snapshots are not meant to exist long term and can lead to performance issues when left in place. They use the same storage infrastructure as the parent disks (with max. 6 TB).

An existing snapshot will initiate a process to keep a (growing) delta file which affects performance and might exceed storage limits and thus lead to a corrupt server. This background process is only stopped when no snapshot exist.

A snapshot preserves the virtual machine just as it was when you took the snapshot - the state of the data on all the virtual machine's disks and whether the virtual machine was powered on, powered off or suspended.

Patching & backup



A backup or snapshot is NOT executed by default before patching - *PatchMeNow!*

This is the responsibility of the consumer

A snapshot preserves the virtual machine just as it was when you took the snapshot - the state of the data on all the virtual machine's disks and whether the virtual machine was powered on, powered off or suspended.

DBMS ML standards

A database belongs to
one application

A database belongs to
one private network

All connections to the
database must be
encrypted using TLS
1.2

Production and non-
production
environments must
always be hosted in
different databases

Use DBaaS for (critical)
business data

DBaaS: PDB

The DBaaS platform is running on a dedicated infrastructure with built-in redundancy on multiple components (power, network, servers, storage) to provide a high level of resilience and is optimized to run Oracle databases.

You do NOT get your own VM with Oracle installed in your own Private Network (PN).

You get your own Pluggable Database (**PDB**). This means you get your own private database with guaranteed service levels in a larger (shared) container database.

Not possible to log on
to the database
server: use SQL Net
(1521)

Not possible to use
the file system of
the database
server: use DBFS

DBaaS service levels

ING DBaaS offers 2 service levels:



The **Thin** service level offers high availability within one datacenter, while a **Normal** adds high availability over the two datacenters. Both services are supported but for the production environment the Normal service is a must. For Development and Test a thin service is required.

Please see the *DBaaS blueprint* on our Confluence space.

DBaaS Application Continuity



The consumer is responsible to configure Application Continuity (AC)

Application Continuity (AC) is a feature that masks database outages from end users and applications by recovering the in-flight work for impacted database sessions following outages. Application Continuity performs this recovery beneath the application so that the outage appears to the application as a slightly delayed execution. By enabling application continuity, the DBaaS service provider gets the possibility to patch the DBaaS service without downtime for the applications using the DBaaS service.

Enabling Application Continuity is simple.

- Use an Oracle driver version 12.2 or later.
- Open port 6200 of the firewall (In the future this will be done automatically when the Database is requested)
 - Create a firewall rule where the source is DBaaS and the destination is your Private Network.
 - Select as template “DbaaS failover notification”
- Use the connection string sent when the database was created.

DBaaS Data high availability

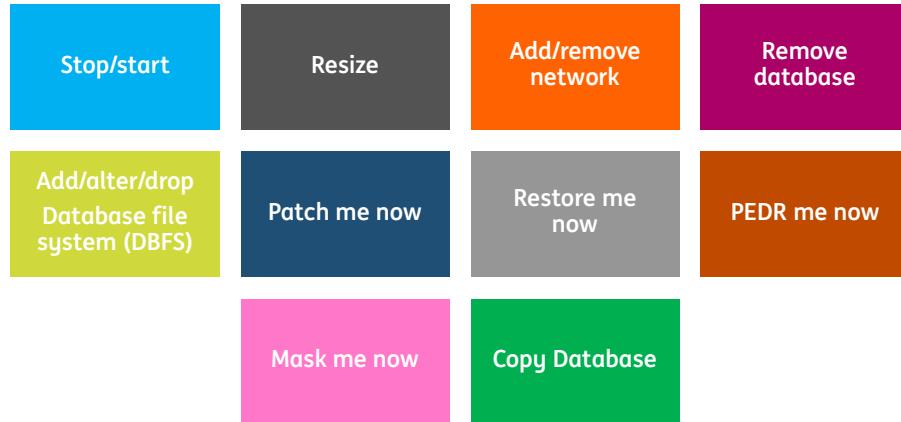
DBaaS offers *forever incremental backups* using ZDLRA (Zero Data Loss Recovery Appliance).



**Even with the dedicated hardware
and state of the art Oracle
software**

RPO = 2 minutes
(local and disaster)

Second day operations DBaaS



15

SQLServer

This service is based on cloud based virtual machines (in your Private Network) running a SQL Server instance.

Two services are available:

Microsoft SQL
Server 2016
Single server, non-
HA

MS SQL Server
2016 AlwaysOn
(HA)

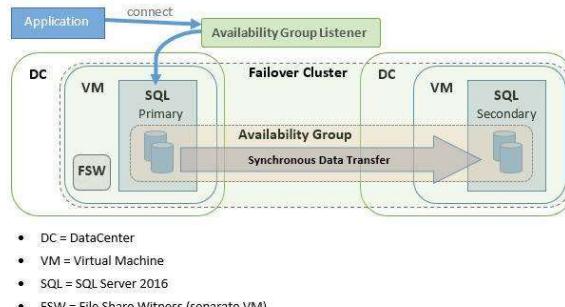
This SQL Server offering includes the ETL component SSIS (Integration Services) by default

SQLServer AlwaysOn

Microsoft SQL Server AlwaysOn has the purpose to enable seamless database failover capability from one SQL Server to another spread **across different datacenters**.

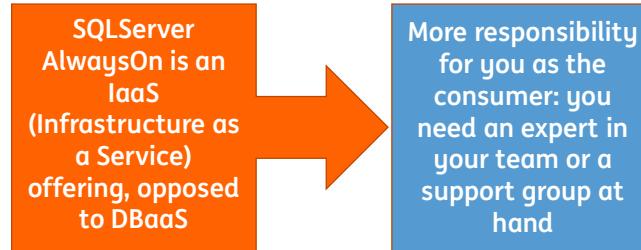
AlwaysOn will contain either 2 or 4 nodes (VMs) on which Microsoft SQL Server 2016 installed. 1 VM is added to the setup which will act as a **Failover Witness** (containing a shared file).

Your application needs to connect to a Availability Group listener



Setup of AO with both nodes in a single datacenter is
NOT supported.

SQLServer AlwaysOn



For example, self service functions specific for SQL Server are not offered in the IPC portal (except on IaaS/Windows level).

Self Service software (made available by the SQL Server team of the infra provider/IPC) needs to be installed by the DevOps teams themselves.

NoSQL vs. SQL

With **SQL** databases, all data has an inherent structure. A conventional database like Microsoft SQL Server, MySQL, or Oracle Database uses a *schema*—a formal definition of how data inserted into the database will be composed.

With **NoSQL**, data can be stored in a schema-less or free-form fashion. Any data can be stored in any record.

Characteristics **NoSQL** data stores:

Transactional processing is not supported

Simpler horizontal scaling

Faster

Eventual consistency

Often caching capabilities included

1

- A NoSQL DB usually does NOT offer transactional processing (rollback, commit and locking)
- A NoSQL DB offers simplicity of design, simpler "horizontal" **scaling** to clusters of machines (which is problematic for relational databases).
- The lack of structure used by NoSQL databases make some operations much **faster**
- NoSQL databases offer a concept of "**eventual consistency**".
- NoSQL databases usually support **auto-sharding**.
- Many NoSQL database technologies have excellent integrated **caching** capabilities.

Most NoSQL databases offer a concept of "**eventual consistency**" in which database changes are propagated to all nodes "eventually" (typically within **milliseconds**) so queries for data might not return updated data immediately or might result in reading data that is not accurate, a problem known as stale reads.

In SQL databases for instance, a given column in a table may be restricted to integers only. As a result, the data recorded in the column will have a high degree of normalization. A SQL database's rigid schema also makes it relatively easy to perform aggregations on the data, for instance by way of JOINs.

NoSQL = “cache”



A NoSQL datastore like KaaS should never be used as a SOR (System of Records) for business data.

Data in a NoSQL datastore must be restored from the SOR, usually a DBMS

Keyspace as a Service (KaaS)

At ING KaaS is offered, which is implemented with *Cassandra*.

It is a “hotel” : you can request your own **Keyspace**, not your own machines (or VMs).

Multiple
replicas (spread
over 2
datacenters
possible)

Active-active
over two DC's is
possible

Consistency
levels
configurable

Encryption
data in
transport on
roadmap

Point-in-time
restore
responsibility
of the squads

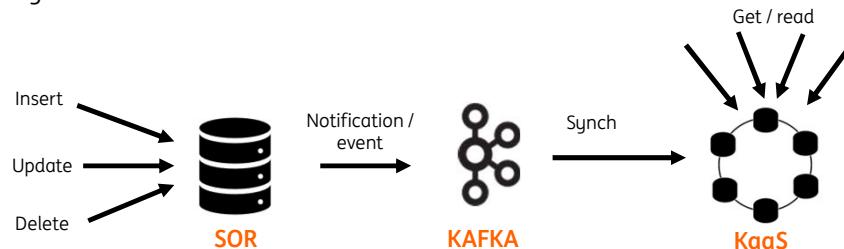
RPO depends on configuration.

Additional **backups** (snapshots) are on request: question is, do you need an extra backup?

Consistency levels: ALL, ONE,
EACH_QUORUM, QUORUM and
LOCAL_QUORUM.

KaaS – use case CQS

A common model used to meet high requirement for data retrieval is to use a “cache” for retrievals/gets.



For update/delete/insert a DBMS is used as SOR, ensuring consistency and integrity (**ACID**)
Using notifications (events on Kafka) a cache, being KaaS / Cassandra, is kept in sync
KaaS is used for the “Gets” offering high performance and availability.
This is also called **Command Query Separation (CQS)**.

BE AWARE OF THE RISK THAT THE CACHE CAN BE OUT OF SYNC WITH THE SOR

4

Keep in mind that there is a risk the “cache” is out of sync.

The DBMS is the SoR (Source of Record).

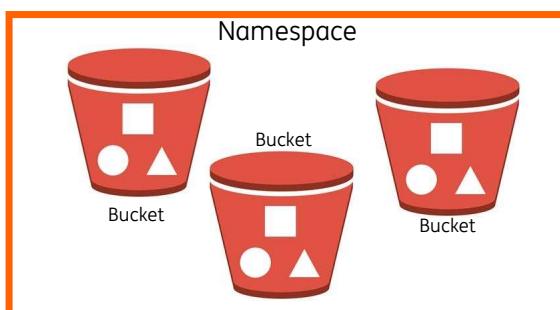
ACID (Atomicity, Consistency, Isolation, Durability)

This model to separate updates from reads is also called **CQS command query separation (or CQRS)**

ECS (S3) Object Store

ECS Object Storage (Dell Elastic Cloud Storage – ECS) is available in IPC and is a very low-cost storage service.

Within a tenant you can create your own **Namespace** with Self Service and within a Namespace you can configure your **Buckets** (no ordering of VMs in your PN).



The S3 API is the standard interface for Object Storage

Both data in transport and at rest can be encrypted. Access is secured with secret codes or hashes.

5

Object storage is a data storage architecture that manages data as objects, opposed to other storage architectures like file systems which manage data as a file hierarchy, and block storage which manages data as blocks within sectors and tracks

Here are some typical object storage characteristics:

- Data is stored as objects versus traditional blocks
- Application developers can easily access objects using simple S3-compatible API calls through “GET” and “PUT” requests, without complex directory structures.
- Objects can include backups, archives, videos, images, logs, HTML files, and more
- It’s unstructured by nature because there is no format to the way data is stored.
- Unlike the directory structure found in traditional file systems, it utilizes a flat list of objects stored in “buckets.”

- Objects are stored using unique IDs rather than filenames, which drastically reduces the overhead required to store data.
- Objects are stored with user-defined metadata, making it easier to find objects at scale
- Objects can be terabytes or even a few kilobytes in size and a single bucket can hold billions of objects.

Cloud ready?



Over the years, due to the wide adoption of Amazon S3, the S3 API has become the de facto standard interface for almost all storage providers

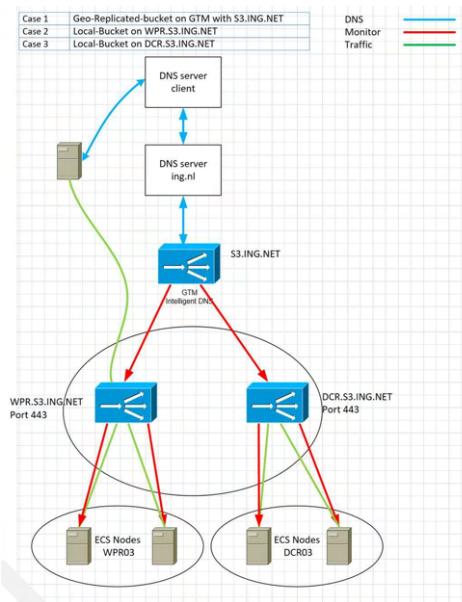
More and more use cases for Object storage are coming up. By 2021 (Gartner) it is expected that more than 80% of enterprise data will be stored in scale-out storage systems in (private-)cloud data centers, up from 30% today.

Object Store service levels

Local Object Storage WPR
(or DCR)

Replicated Object Storage
WPR -> DCR

Replicated Object Storage
DCR -> WPR



ECS (S3) Object Store Second day operations

ECS add
Objectuser

Change S3
Password

Extend
Namespace
(max. 50 TB for
now)

Retrieve Details
of Namespace

Remove Elastic
Cloud Storage

With your object user and secret and access to the S3 API you can create (extra) buckets yourself.

ECS (S3) Backup and RPO

There is no backup available: use versioning or retention settings

Feature	Local Object Storage	Replicated Object Storage
Storage	From GB up to multiple PB	From GB up to multiple PB
Local Recovery Time Objective (RTO)	8h	10 minutes
Local Recovery Point Objective (RPO, data loss)	Max 10 min.	Max 10 min.
Disaster Recovery Time Objective (RTO)	N/A (No DR)	Read immediate, write 10 minutes
Disaster Recovery Point Objective (RPO, data loss)	N/A (No DR)	maximum 15 minutes data loss

ECS (S3) Object Store use cases

There are numerous use cases for the object store, however usage is rather new within ING (end of 2020).

Please consult your BTA's and architects for use cases.

One thing to consider is management (and exchange?) of keys and secrets.

NAS: Network Attached Storage

You can request a replicated or non-replicated share per DC in the IPC self service portal.
No extra machines/VM's are provisioned in your Private Network.

The implementation of this NAS File Sharing is on a *Isilon* NAS cluster that (when requested) replicates its data asynchronous (**with at least 10 minutes delay**) to another NAS cluster that is in a second datacenter.

Mount with client software	No locking (WORM)	Protocol supported NFS, SMB soon	No backups	NAS is not meant for backups/archiving
----------------------------	-------------------	----------------------------------	------------	--

11

The shared storage has to be mounted by the customer with appropriate client software.

Network File System (NFS) is a distributed file system protocol allowing a client to access files over network much like local storage is accessed. Stored data won't be locked (WORM - Write Once Read Many) and can be deleted or modified at any level by the customer like normal filesystem.

NAS share protocol supported: NFS, (SMB will soon follow)

This File Share (NFS) service is used when you want to store data for FileShare purposes only.

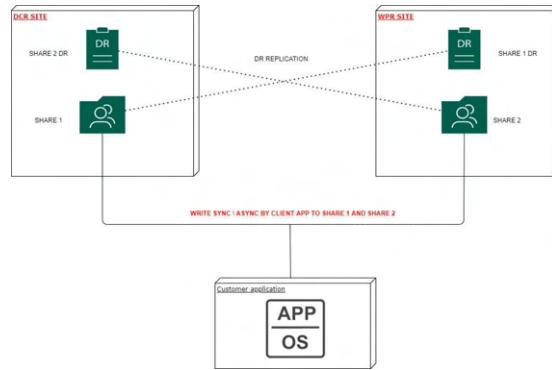
The implementation is multitenant, including isolation (Security, Restore, DR. Performance best effort) between tenants and meets the ING Security Guidelines.

NAS: Backup and RPO

Backup is not part of this offering.

The minimum recommendation for backup is :

- Order 2 shares for one purpose (one in DCR , second in WPR)
- Write data SYNC / ASYNC to both shares by client App/OS



12

This service offers a logical error protection capability.

NAS: use cases and ML guidelines

Use NAS only
when S3 Object
Store is not
possible

Use NAS only in
the same
networkzone
(PN)

NFS (and SMB) are considered “tightly coupled” protocols: there is a direct dependency / connection between the application (-server) and the NAS (as with JDBC, a DBMS).

HTTP(s) is considered a “loosely coupled” protocol and more secure.

Archiving

One of the differences between Archiving and File Share functionality is that for archiving often requires WORM (Write Once Read Many) compliancy reasons.

It is called a WORM share, compliancy mode or Smartlock.

Uses same technology as NAS File sharing: Isilon

For now: no self service, use request form

WORM on request

More than 5TB: order 3 months in advance

Quota's:
Hard, advisory

14

SmartLock compliance mode enables you to protect your data in compliance with the regulations defined by U.S. Securities and Exchange Commission rule 17a-4.

With SmartLock we talk about retention periods or better how long data can not be changed or deleted from the Isilon.

A retention period is the length of time that a file remains in a WORM state before being released from a WORM state.

You can request a minimum and a maximum retention period for a SmartLock directory to prevent files from being retained for too long or too short a time period. Smartlock is always defined for a directory. So all files in that directory inherits the setting.

Quota's:

Hard

Limits disk usage to a size that cannot be exceeded (100% full). If an operation, such as a file write, causes a quota target to exceed a hard quota, the following events occur:

- the operation fails (no more writes).
- a notification is issued to specified recipients.

Writes resume when the usage falls below the threshold.

Soft

Allows a limit with a grace period that can be exceeded until the grace period expires. Soft-quota's are not used at ING.

Advisory

An informational limit that can be exceeded (80% full).

When an advisory quota threshold is exceeded:

- a notification is issued to specified recipients (writes still allowed).

Advisory thresholds do not prevent data writes.

No “DR capable” machines



Do not order “DR capable” machines
in IPC

DATA HA: Key takeaways

Availability is not the same as Integrity

Keep the performance of backup / restore in mind

RPO Zero = solve it in the application

Separate your data from your application (server)

Because you can, doesn't mean it is allowed

Enable backup on your host when needed

Always remove a snapshot ASAP

Create a backup before patching

use DBMS only as SOR for business data

With a HA DBMS configure your App correctly

Use S3 for your data whenever possible

Be security, cost and cloud aware



do your thing



ING Container hosting Platform ICHP

Solutionists DBNL



What is an Application container?



Containers are a solution to the problem of how to get software to run reliably when moved from one computing environment to another: **portability**.



A container consists of an **entire runtime environment**: an application, plus all its dependencies, libraries and other binaries, and configuration files needed to run it, bundled into one package.

Docker (as an application container) and **Kubernetes** (as an orchestrator) dramatically reduces the burden of deployment: deploying 100 apps or services is no longer 100 times the work of deploying a single app. For many companies it results in a dramatic **reduction in the cost** of adopting microservices.

2

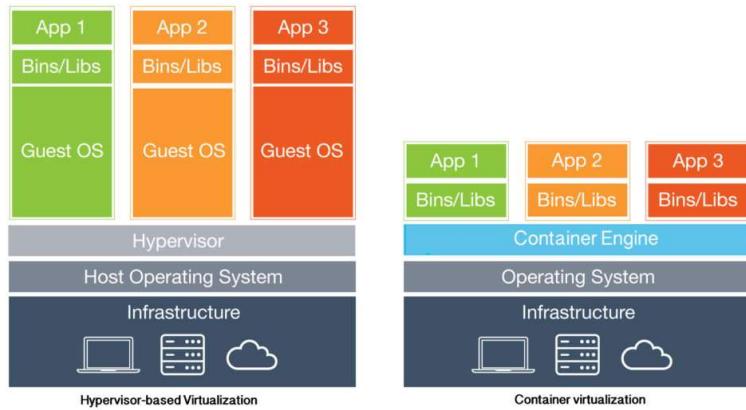
This move might be from a developer's laptop to a test environment, from a staging environment into production, and perhaps from a physical machine in a data center to a virtual machine in a private or public cloud.

By containerizing the application platform and its dependencies, differences in OS distributions and underlying infrastructure are abstracted away.

Simpler deployment is possible not just because Docker and Kubernetes provide powerful abstractions at all the right levels, but because they standardize the patterns for packaging and deployment across the entire organization.

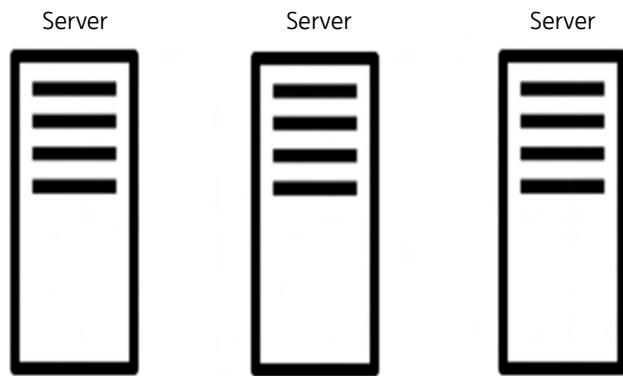
Difference Application Containerization - Hypervisors

Containers virtualize at the operating system level; **Hypervisors** virtualize at the hardware level.



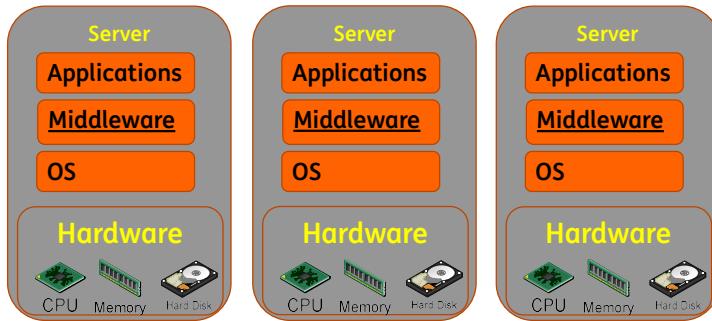
3

Traditional architecture



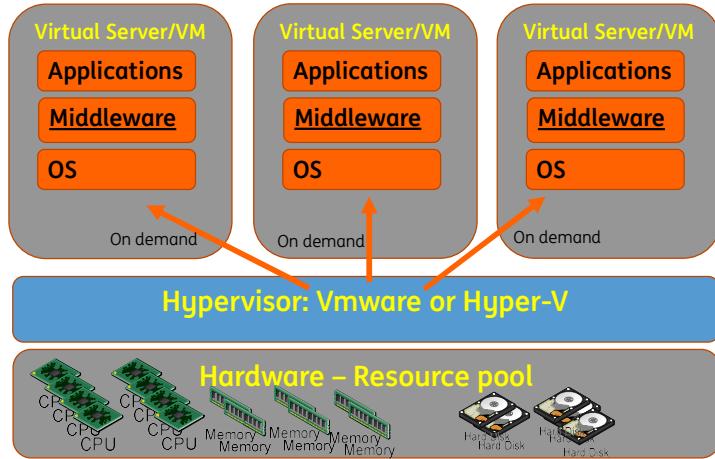
4

Traditional architecture



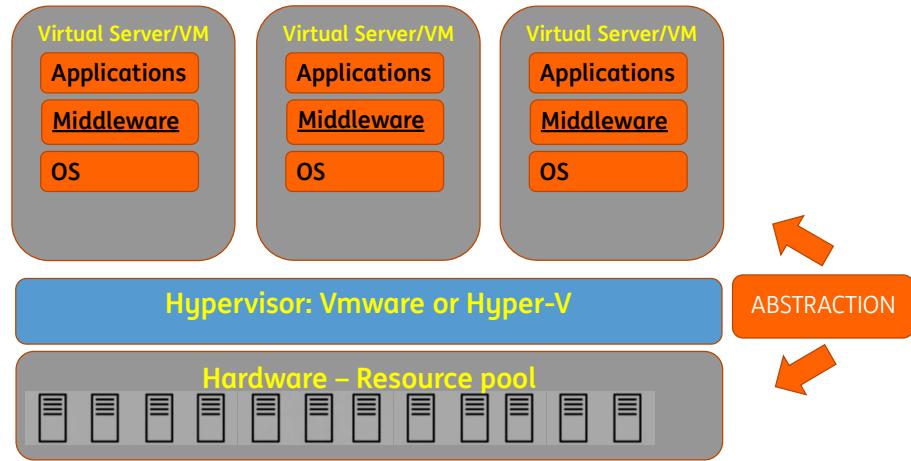
5

Virtual architecture



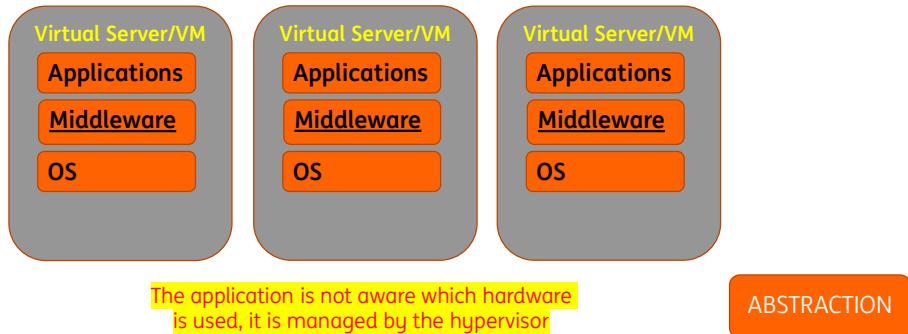
6

Virtual architecture



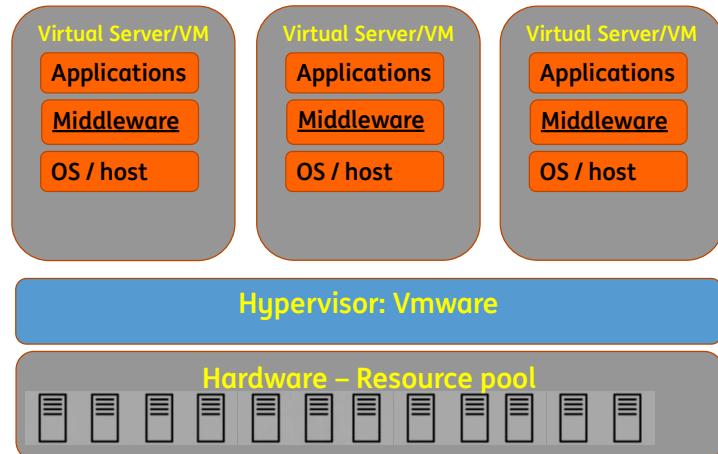
7

Virtual architecture



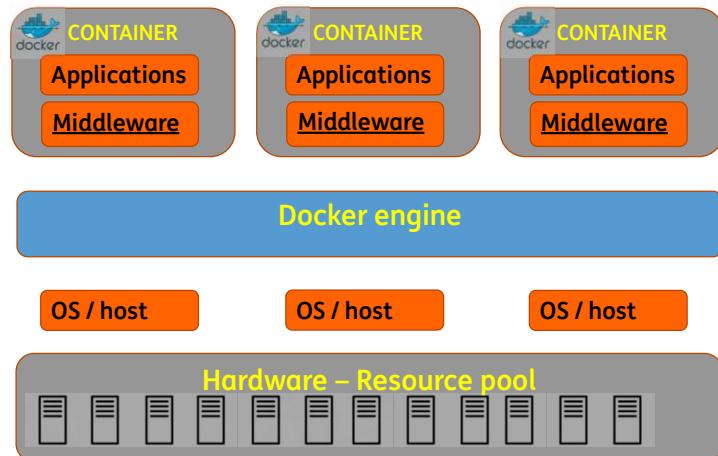
8

Virtual architecture



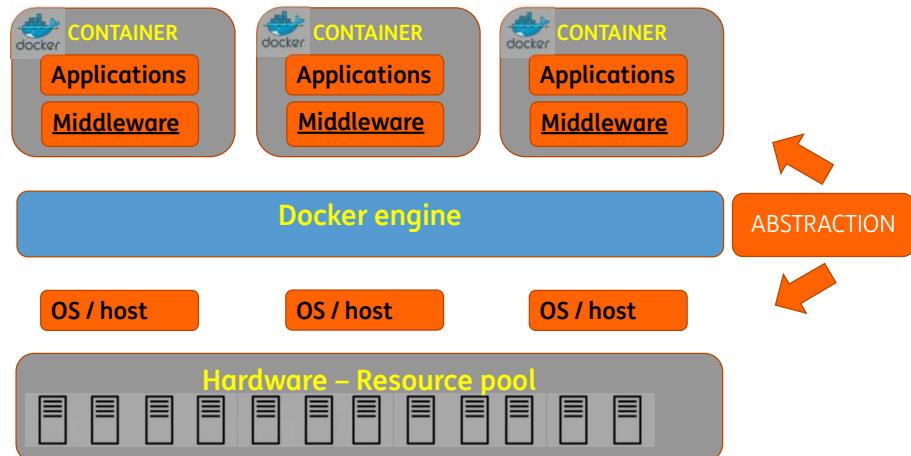
9

Application container architecture : Docker



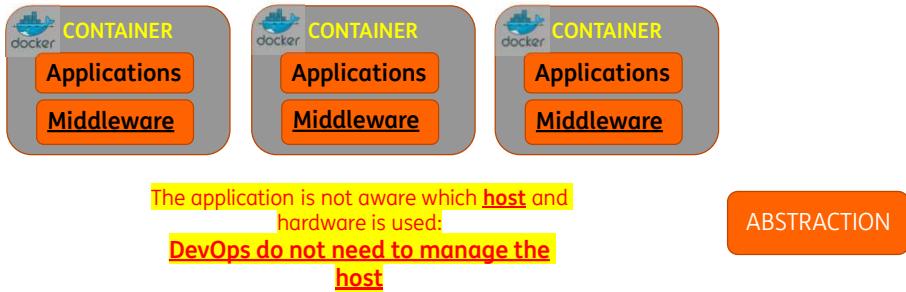
10

Application container architecture



11

Application container architecture



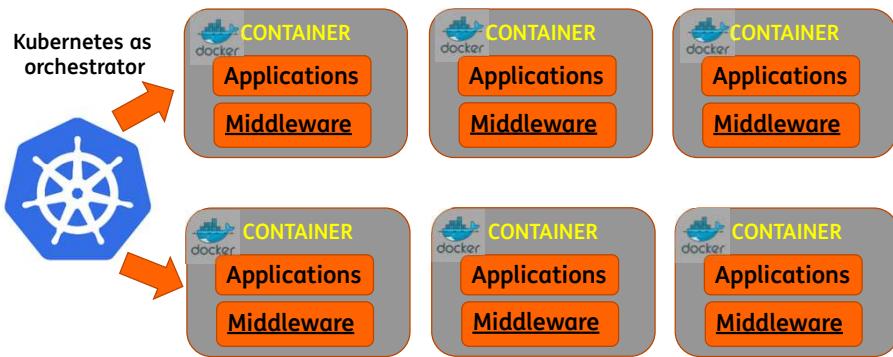
12

Containers are a solution to the problem of how to get software to run reliably when moved from **one computing environment to another**. A container consists of an entire runtime environment: an application, plus all its dependencies, libraries and other binaries, and configuration files needed to run it, bundled into one package.

By containerizing the application platform and its dependencies, **differences in OS distributions and underlying infrastructure are abstracted away.**

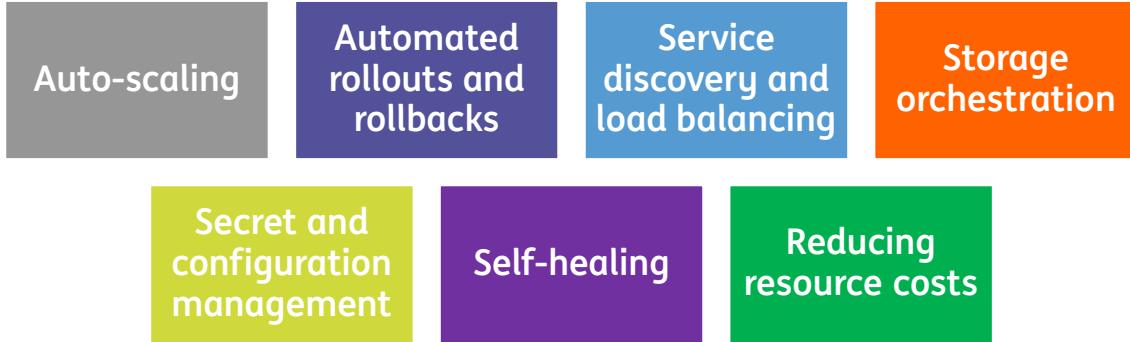
Docker and Kubernetes dramatically reduces the burden of deployment: deploying 100 apps or services is no longer 100 times the work of deploying a single app. For many companies it results in a dramatic **reduction in the cost** of adopting microservices.

Docker & Kubernetes



13

Kubernetes features



14

- Horizontal scaling: scale your application as needed from command line or UI.
- Automated rollouts and rollbacks: roll out changes that monitor the health of your application—ensuring all instances don't fail or go down simultaneously. If something goes wrong, K8S automatically rolls back the change.
- Service discovery and load balancing: containers get their own IP so you can put a set of containers behind a single DNS name for load balancing.
- Storage orchestration: automatically mount local or public cloud or a network storage.
- Secret and configuration management: create and update secrets and configs without rebuilding your image.
- Self-healing: the platform heals many problems: restarting failed containers, replacing and rescheduling containers as nodes die, killing containers that don't respond.
- Reducing resource costs: multiple containers can share the same OS and network connection. This is much more efficient when it comes to resource utilisation than creating a virtual machine with its own OS for each application.

Benefits of Containers

Shorter time to market

Portability (cloud)

Increased Availability / security

Mature Ecosystem
–
Docker/Kubernetes

Reducing resource costs

Immutable servers

Application containers introduce the concept of **immutable servers**.

Any application change is a redeploy by an automated CI/CD pipeline

With an immutable server, you make each change to a base image, and then you know that all instances created from that image are consistent.



Any (manual) change to a running system introduces risk.

Having **no access** can simplify security procedures (and therefore speed up the usability of a newly provisioned server/runtime environment).

16

No Access means no access: even during an **incident/service disruption!**

Or you need a mechanism (**Break the glass**) where you can stop your server and isolate your server for further investigation: then, however, it must be destroyed: it may never be brought back up again!

With Continuous Delivery of software, it's safer to compile a given version of an application into a deployable artifact only once, and know that you are deploying and running a consistently built application in all environments.

Any changes that are needed can be made to the base image, tested, and then rolled out.

Not having access into the servers puts the right incentive on the team to collect and store logs and system metrics externally. This way, debugging can happen while the server is long gone.

ICHP and Cloud Native

ICHP is the **Cloud Native** platform of ING.

It has been designed according to Cloud Native Design Principles ("The 7 Commandments"*)�

The point is: **ICHP is NOT a good fit for ALL applications.**

The Holy Trinity (for Cloud Native Apps)

- **Immutable**
- **Short Living**
- **Stateless**

Maybe most important: a container must be able to be destroyed, restarted without impact on business logic (**idempotent**).

To use the full potential of an orchestrator like Kubernetes, DevOps can hand over control of stopping and starting (rollouts, rollbacks), routing and scalability of their application to Kubernetes



17

* Google: PRINCIPLES OF CONTAINER-BASED APPLICATION DESIGN

CNCF Cloud Native Definition v1.0

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

*These techniques enable **loosely coupled** systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.*

For more information, see the Cloud Native Computing Foundation (“CNCF”) Charter at:

<https://cncf.io/about/charter>

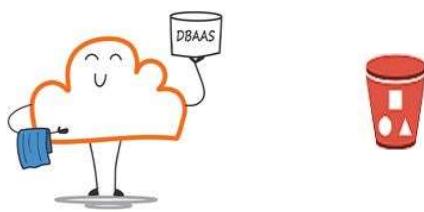
Stateless?

The application must not depend on data in the container.

However the application can use data outside the container, outside the Kubernetes cluster.

For example: **DBaaS, ECS S3 Object store, Kafka** etc.

Please review the design principles mentioned in the Data High Availability modules (one datastore-one App, SoR etc.) of this eLearning series.



When not to migrate to ICHP



When not to migrate to ICHP

If your application cannot recover correctly from a crash or restart, for example:

- A batch application that restarts during processing of a file, does it continue where it left off? Or does it start over and causes data inconsistencies (double processing?)
- An application that depends on data stored in the container, when it is restarted this data is lost, does it cause data inconsistencies?
- A monolithic (COTS) application, can it managed by an orchestrator (scaling, routing etc.)? How quickly can it be restarted? Are all changes done using a pipeline?

If network access is not controlled

- Does the application support endpoint isolation or can any application in the Kubernetes cluster access the endpoint/API (with a route created without accounting or an internal traffic policy)?

This is especially critical for external facing applications.

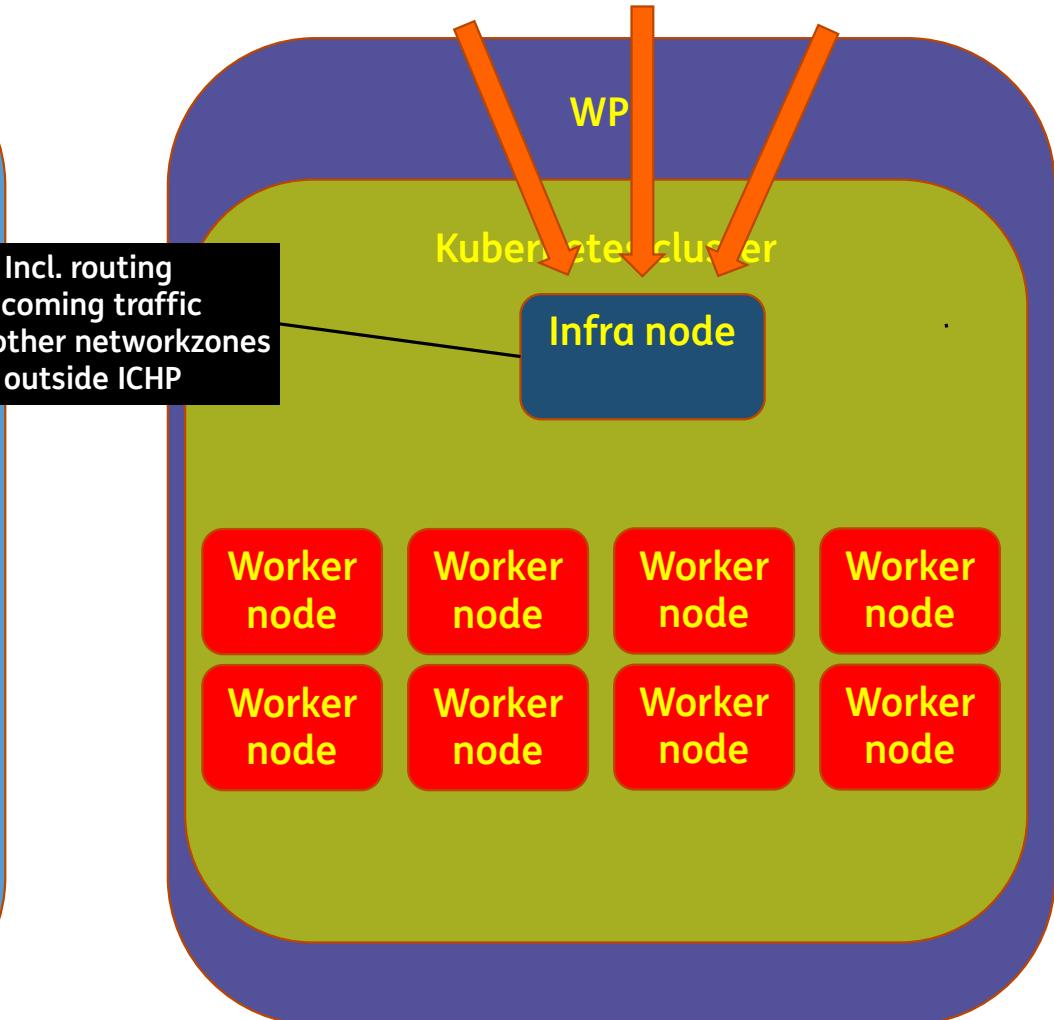
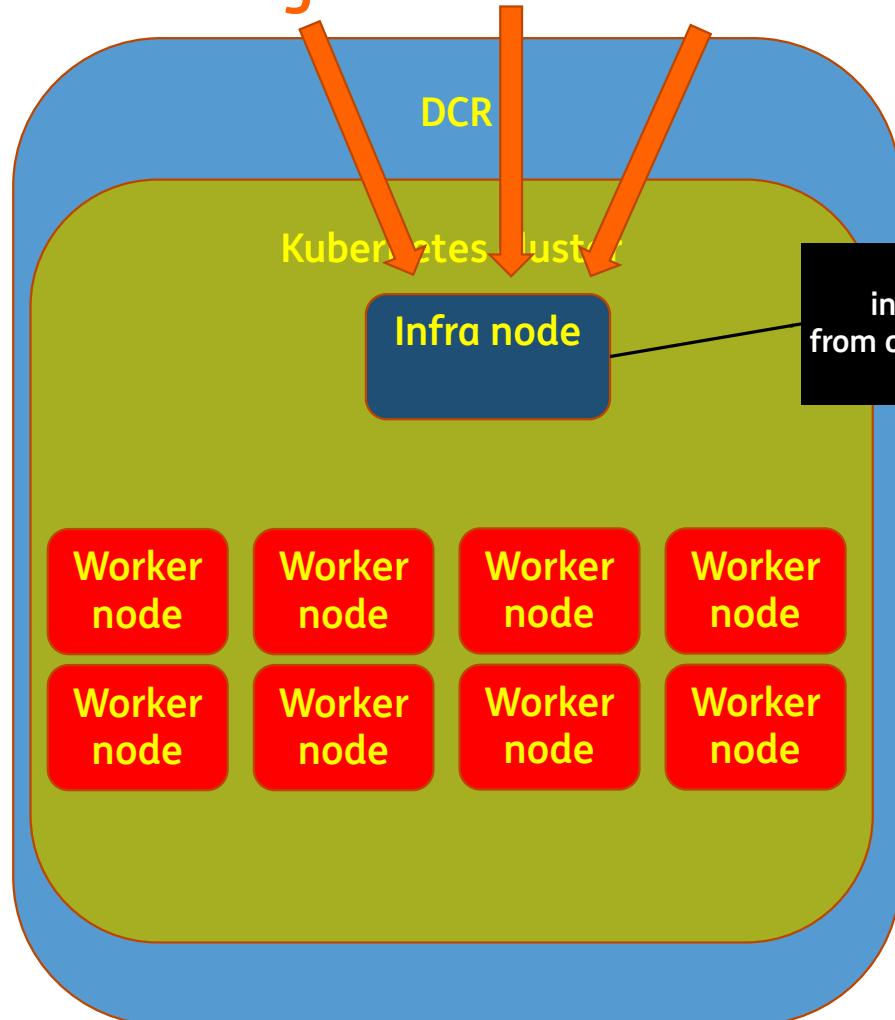


ING Container hosting Platform ICHP

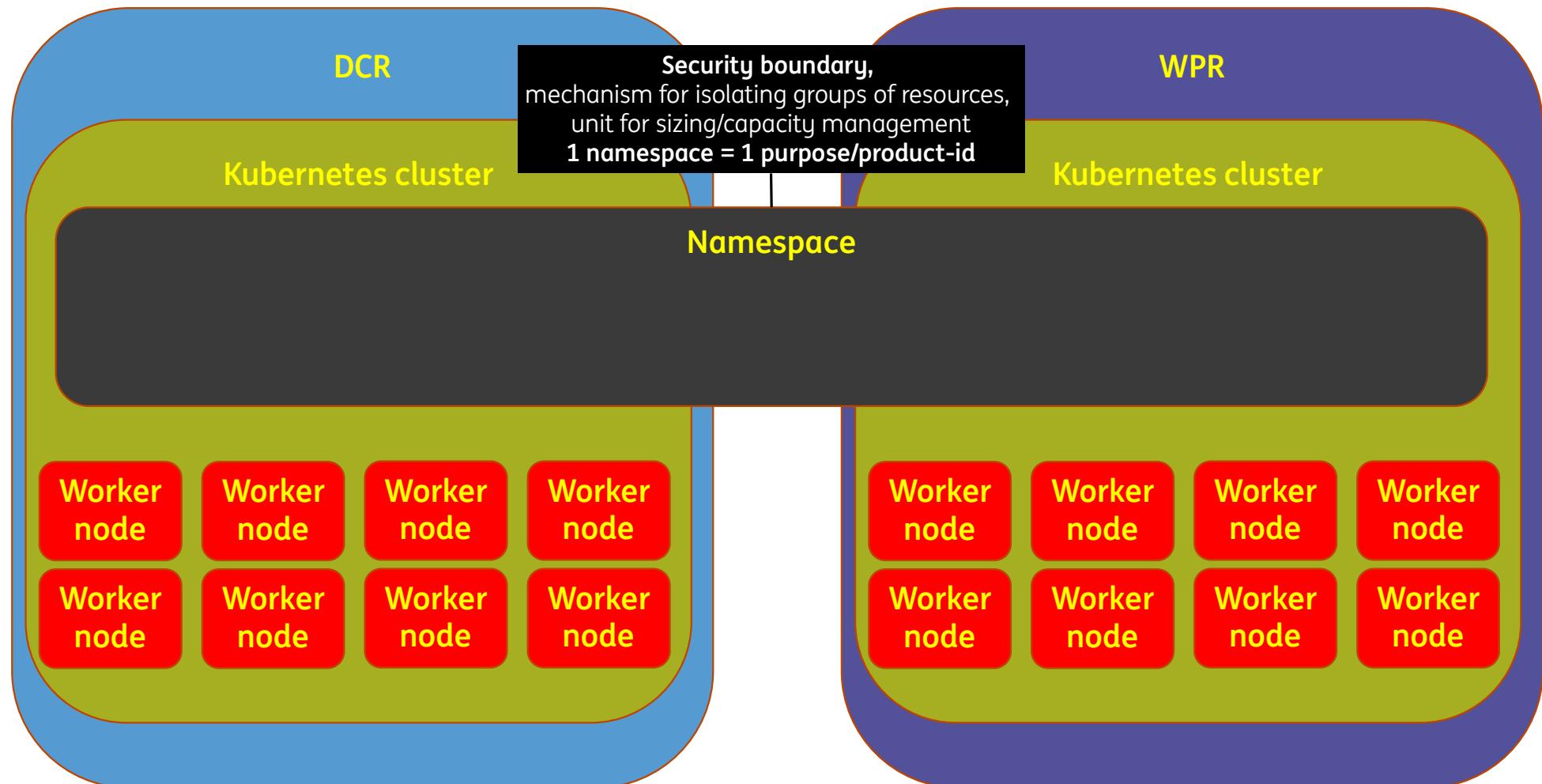
Solutionists DBNL



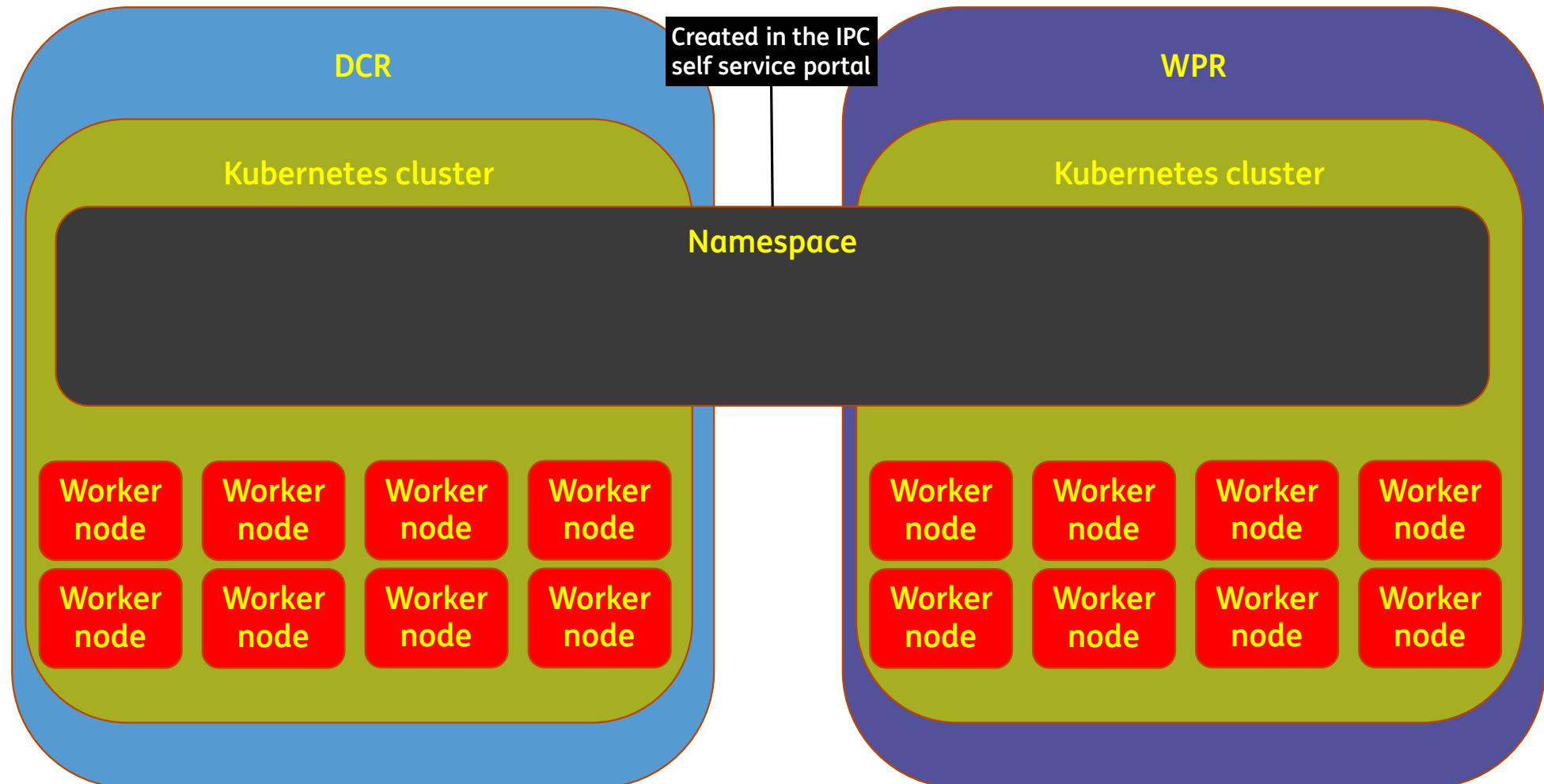
ICHP Design essentials



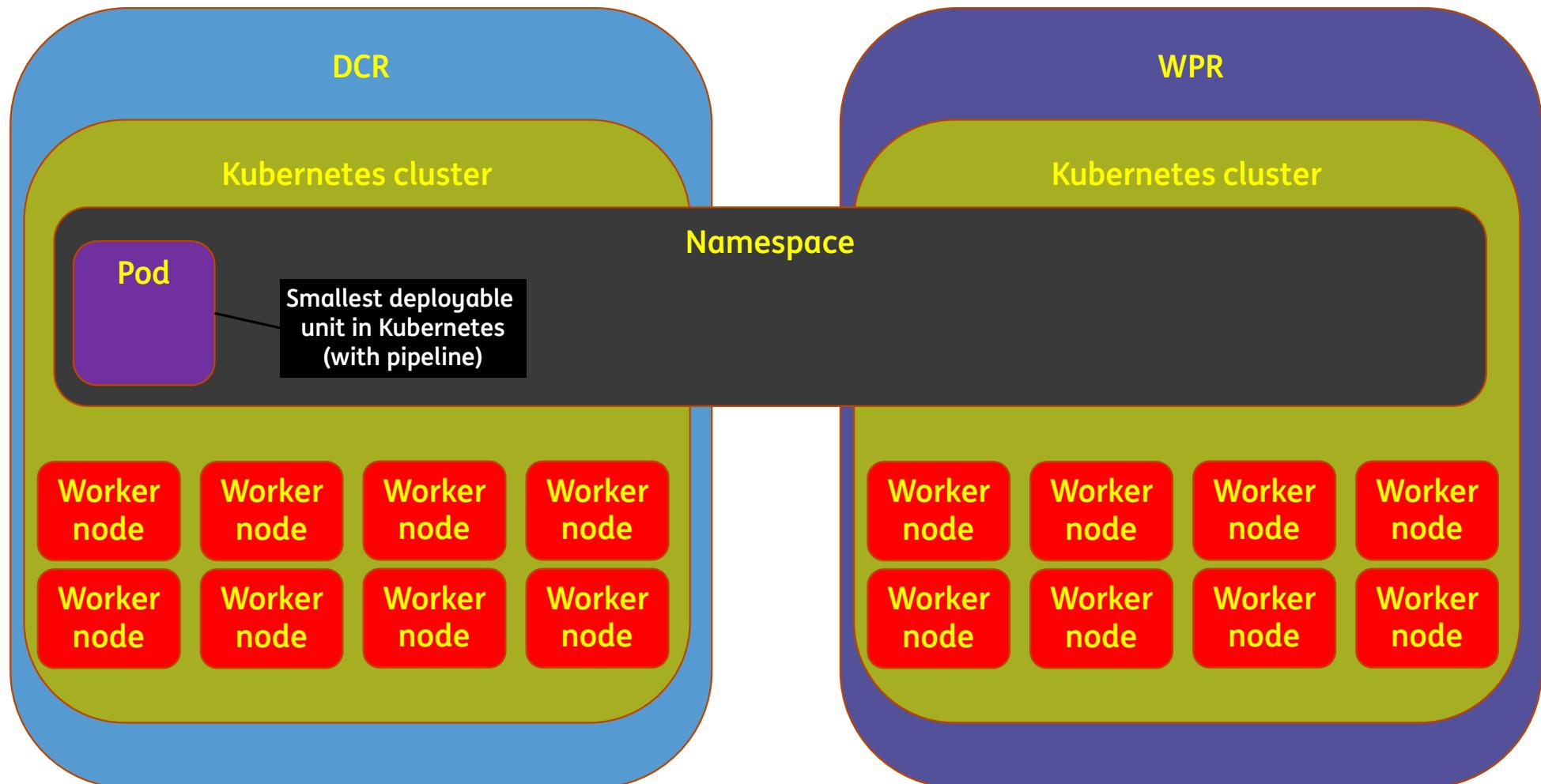
ICHP Design essentials



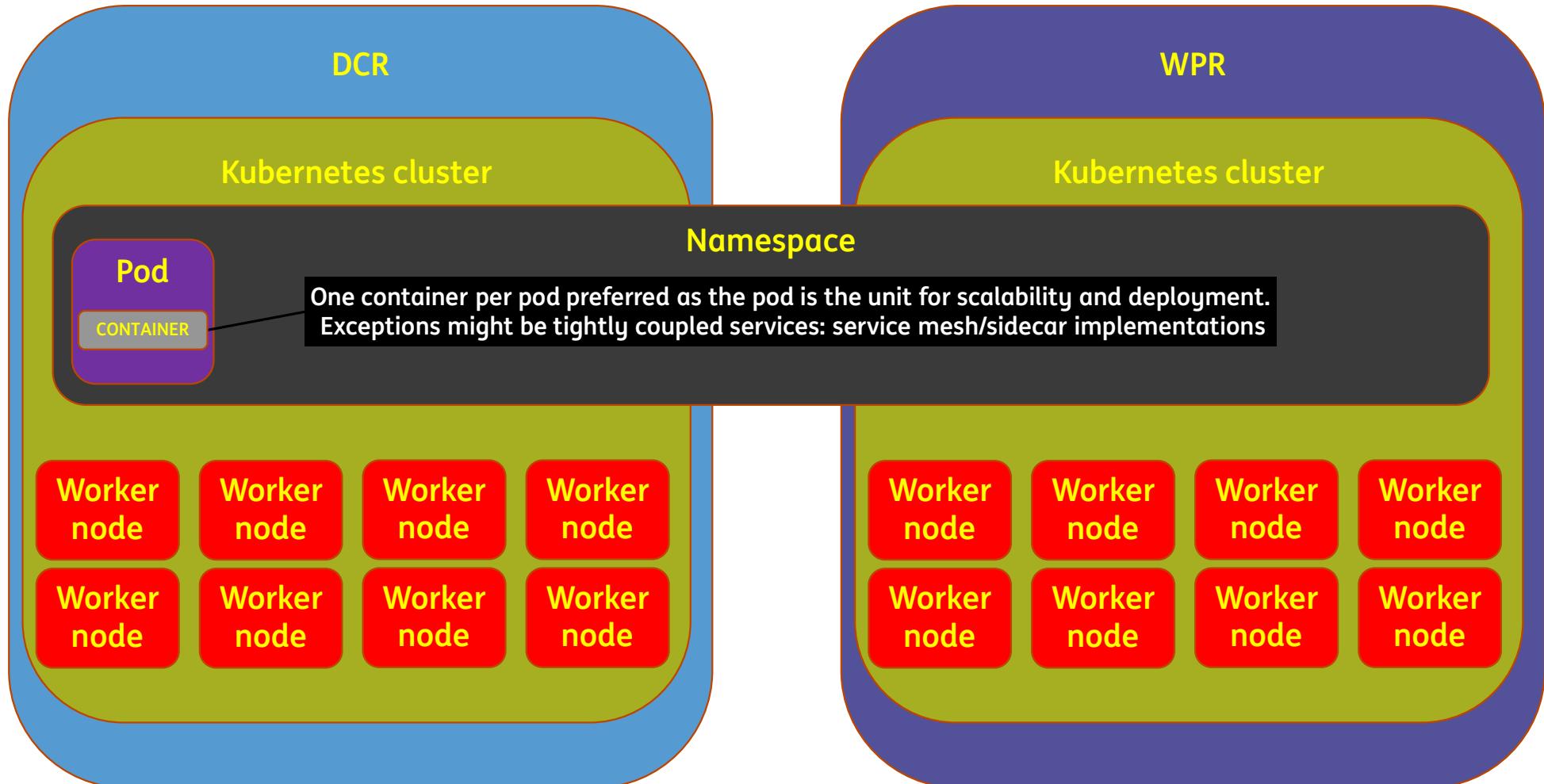
ICHP Design essentials



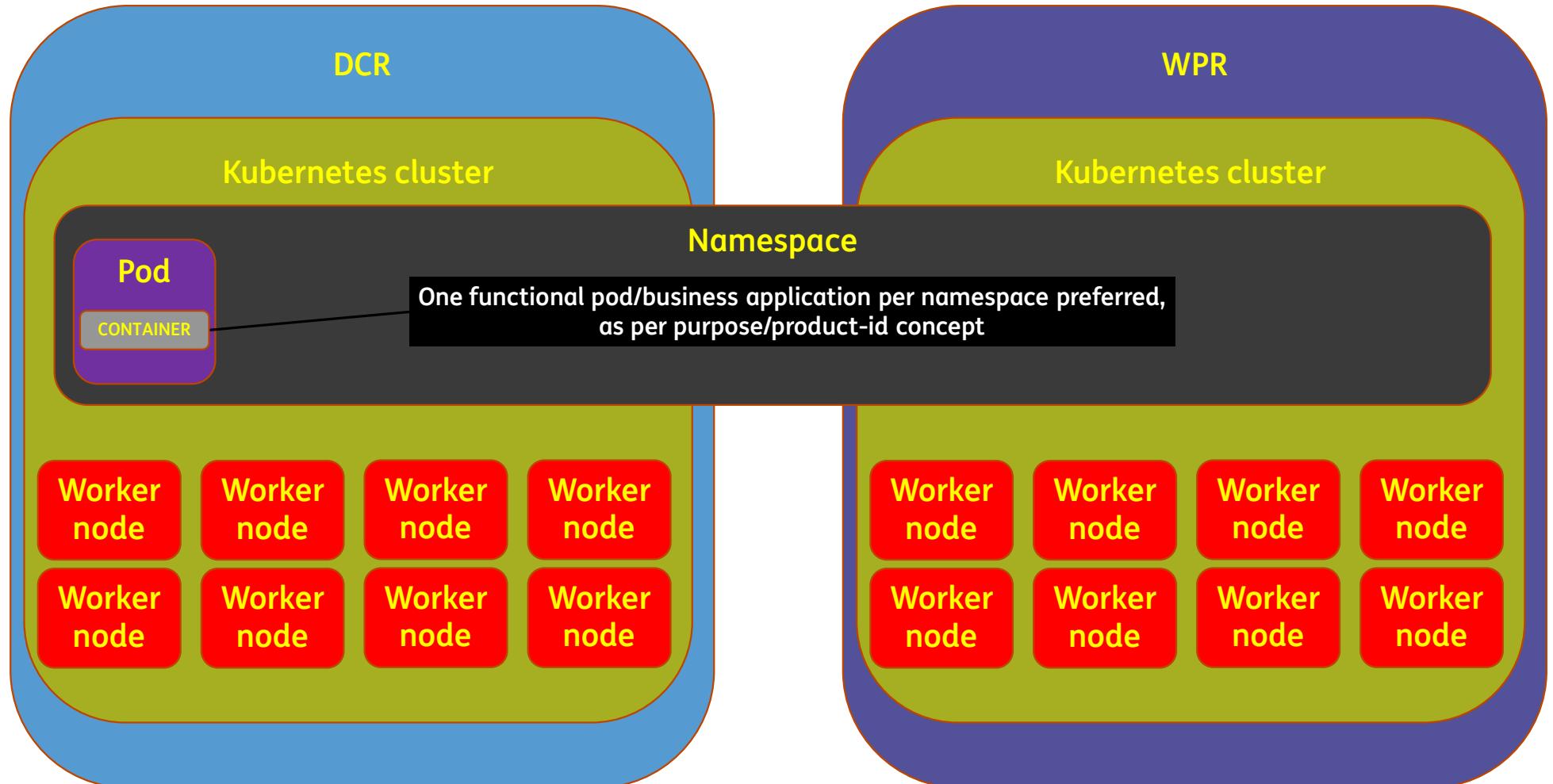
ICHP Design essentials



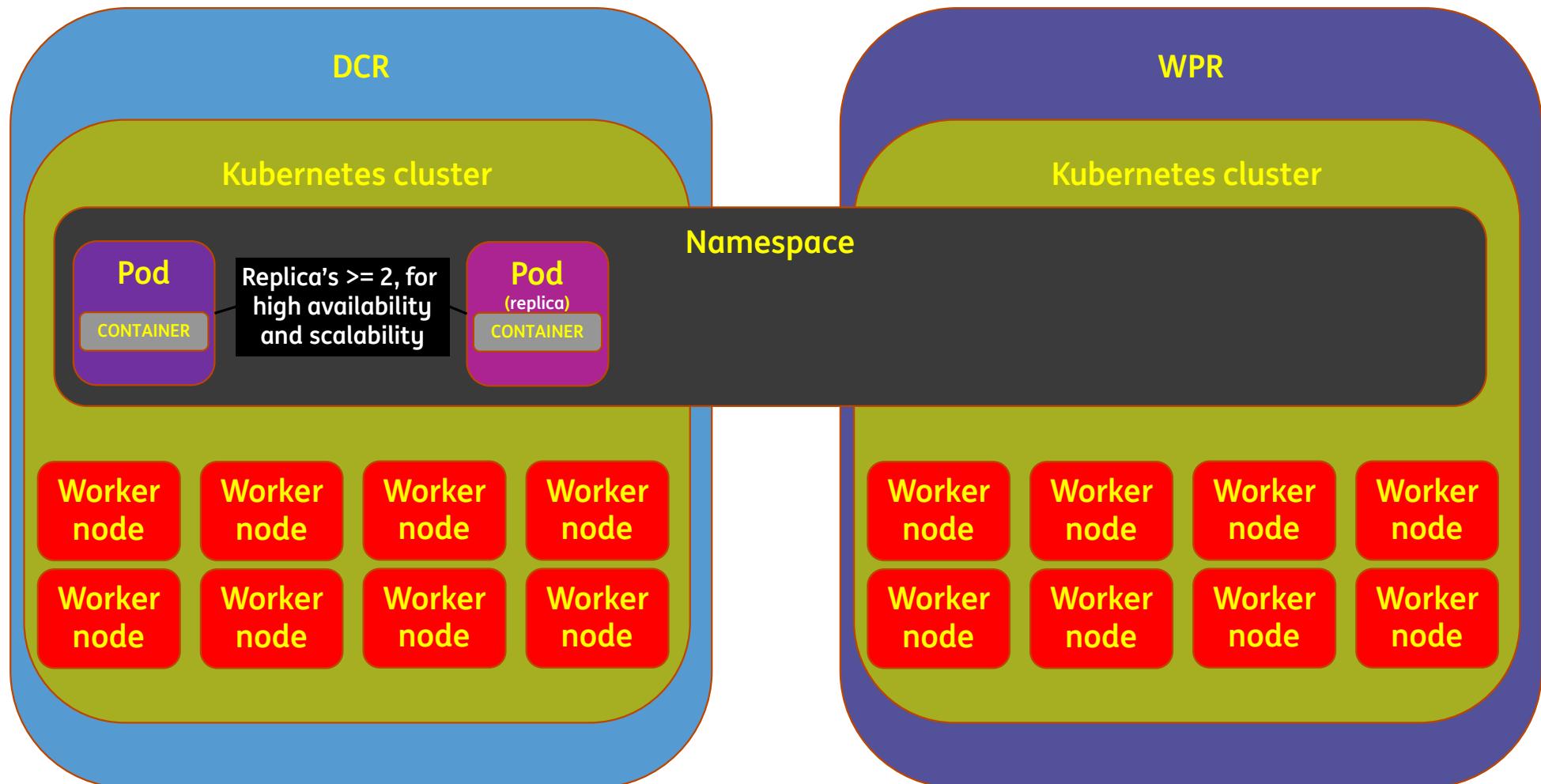
ICHP Design essentials



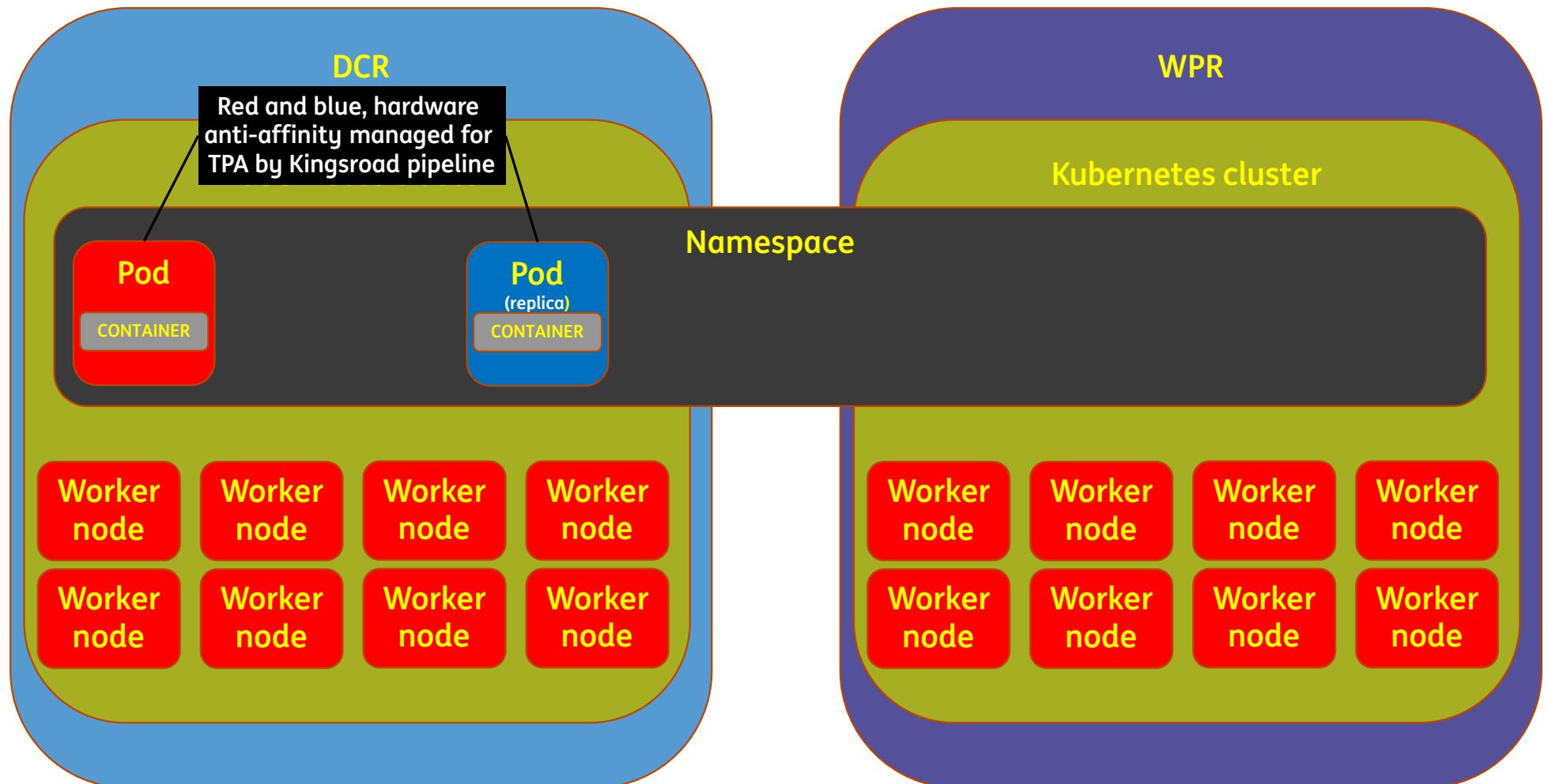
ICHP Design essentials



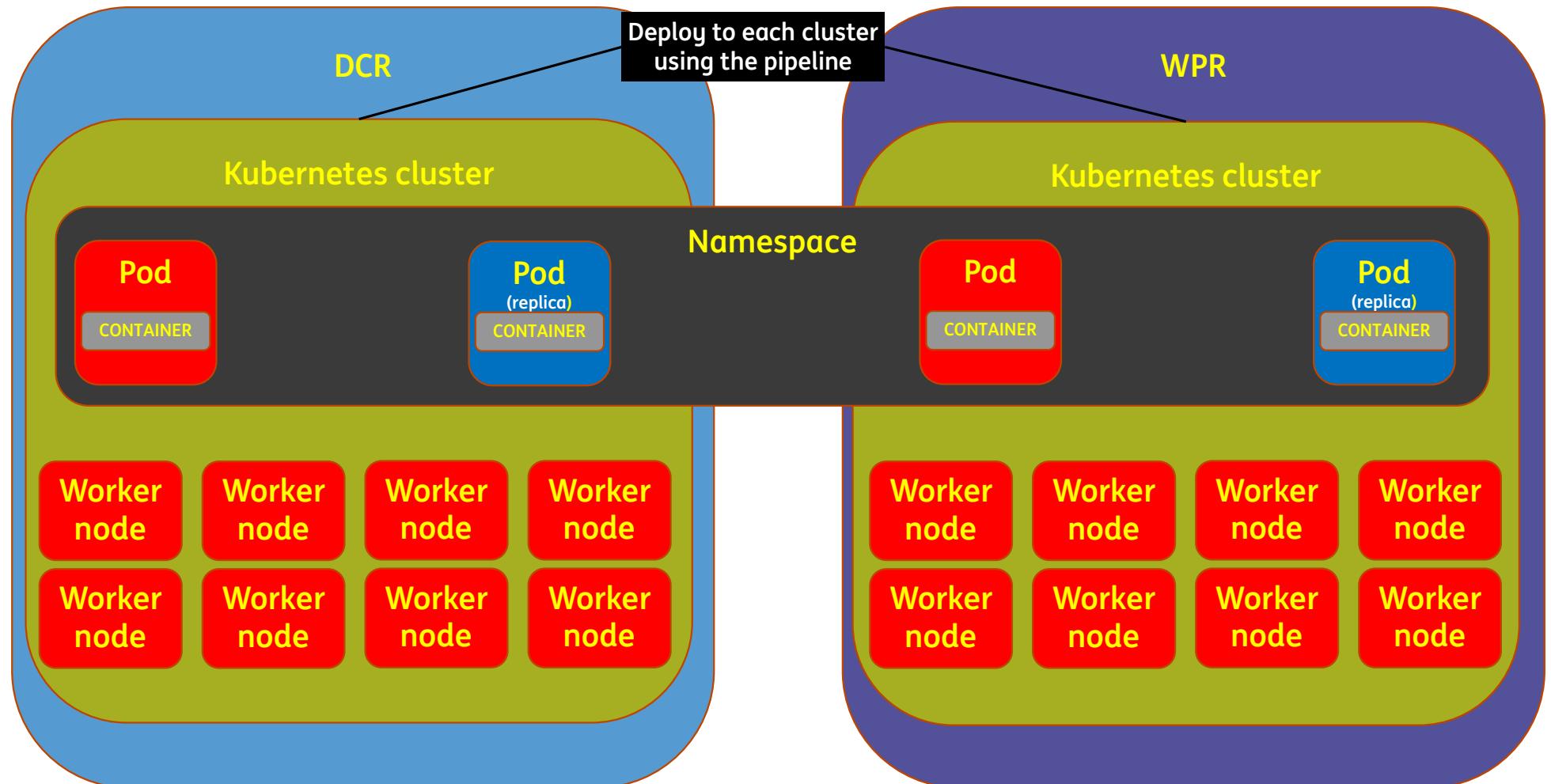
ICHP Design essentials



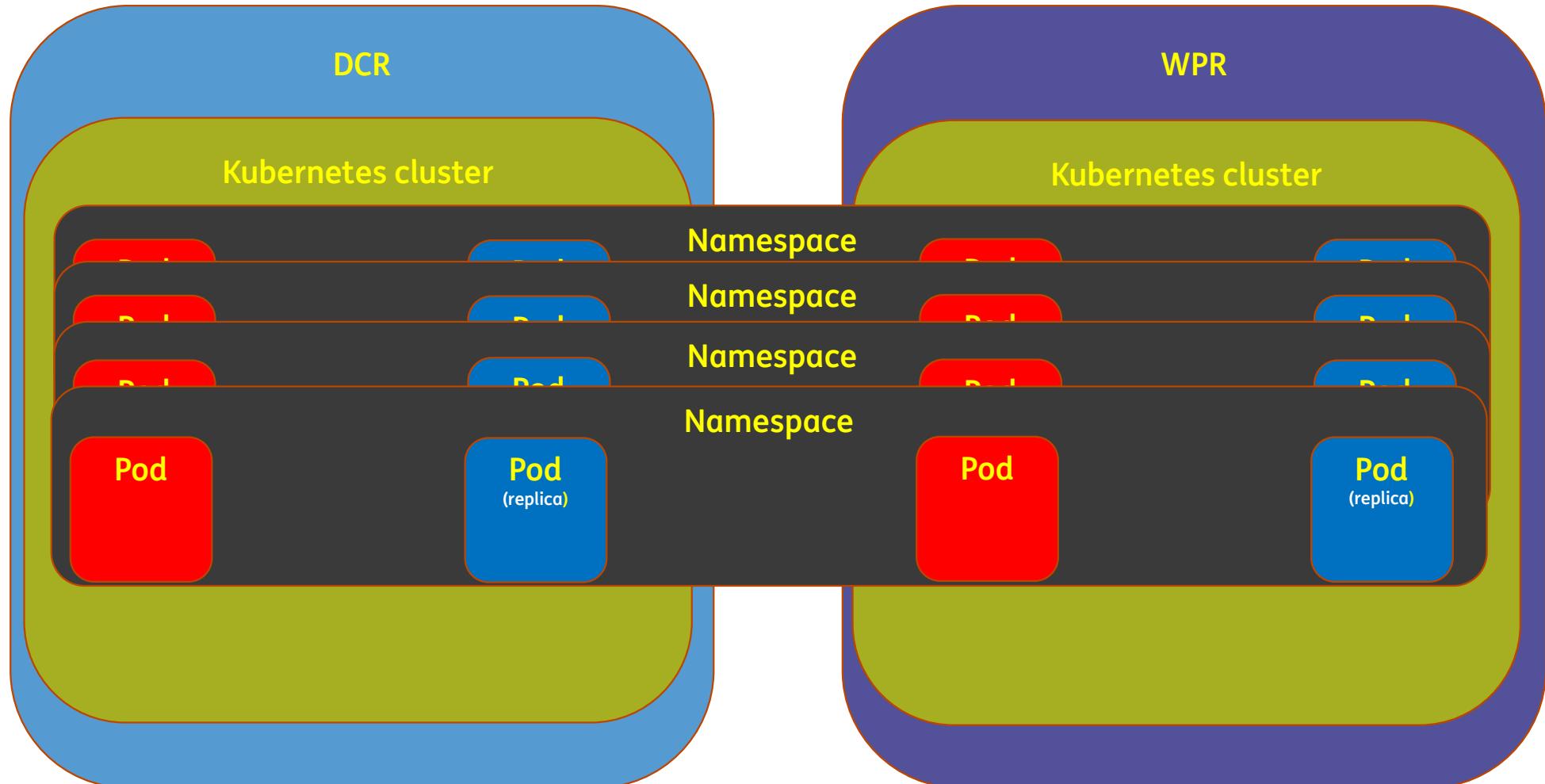
ICHP Design essentials



ICHP Design essentials



ICHP Design essentials



Purpose – product-id



The goal of the concept of a purpose / product-id is to enable (automated) migration of resources from one owner (squad) to another, without the necessity of destroying and/or recreating resources.

These resources involve the physical resources (hosts, network assets) as well as access control resources (NPA's, Solar etc.).

A namespace can be moved from one product-id to another.

However, migration for all resources per purpose is not available yet.

Purpose / product-id – business CI - changeability

Best practice is one on one, **1 Application = 1 purpose / product-id.**

This is most flexible for changeability (**future** changes of ownership/product-id).

One application can be multiple **tightly coupled** microservices.

Security and access controls (CIA rating / DRA, Solar) are **per business CI** in Snow (CMDB).

Having one business CI for your API or having one business CI with multiple application components (= your API's/microservices) has an impact on a **future** change of product-id.

The difficulty is the “future” : we are not fortunetellers...



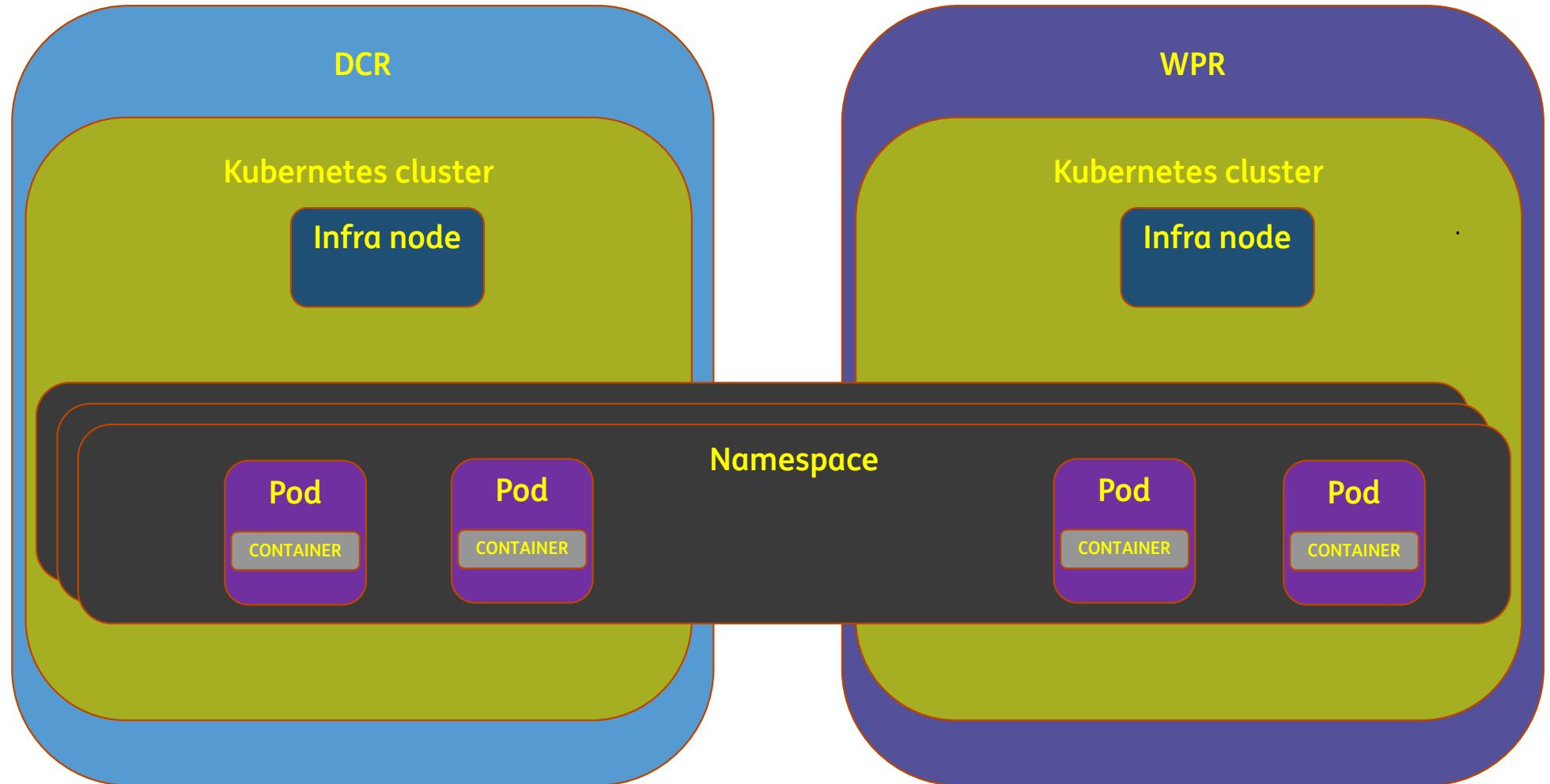
Kubernetes Network policy

ingress:
- **from:**
egress:
- **to:**

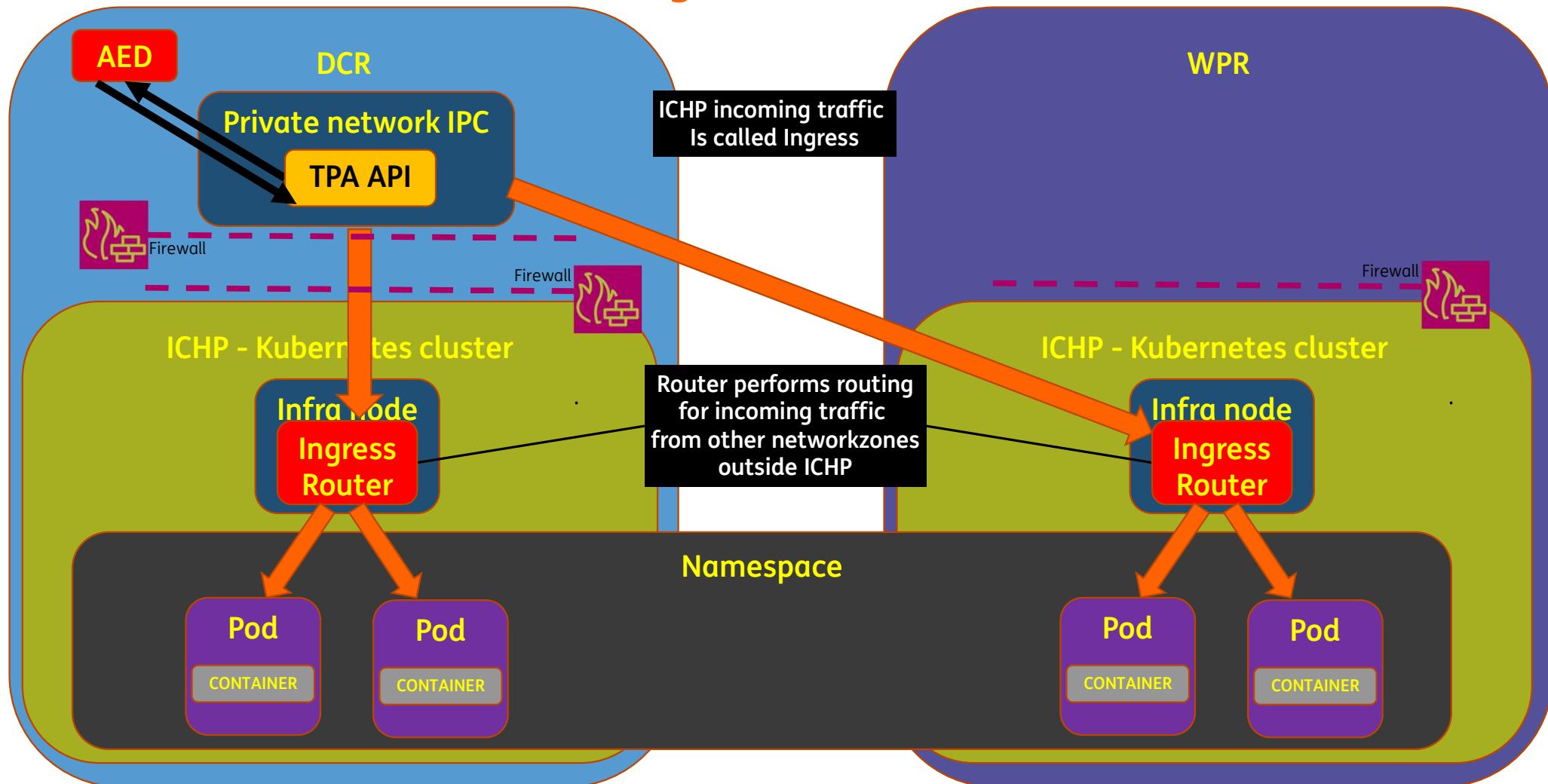


Network basics
ICHP

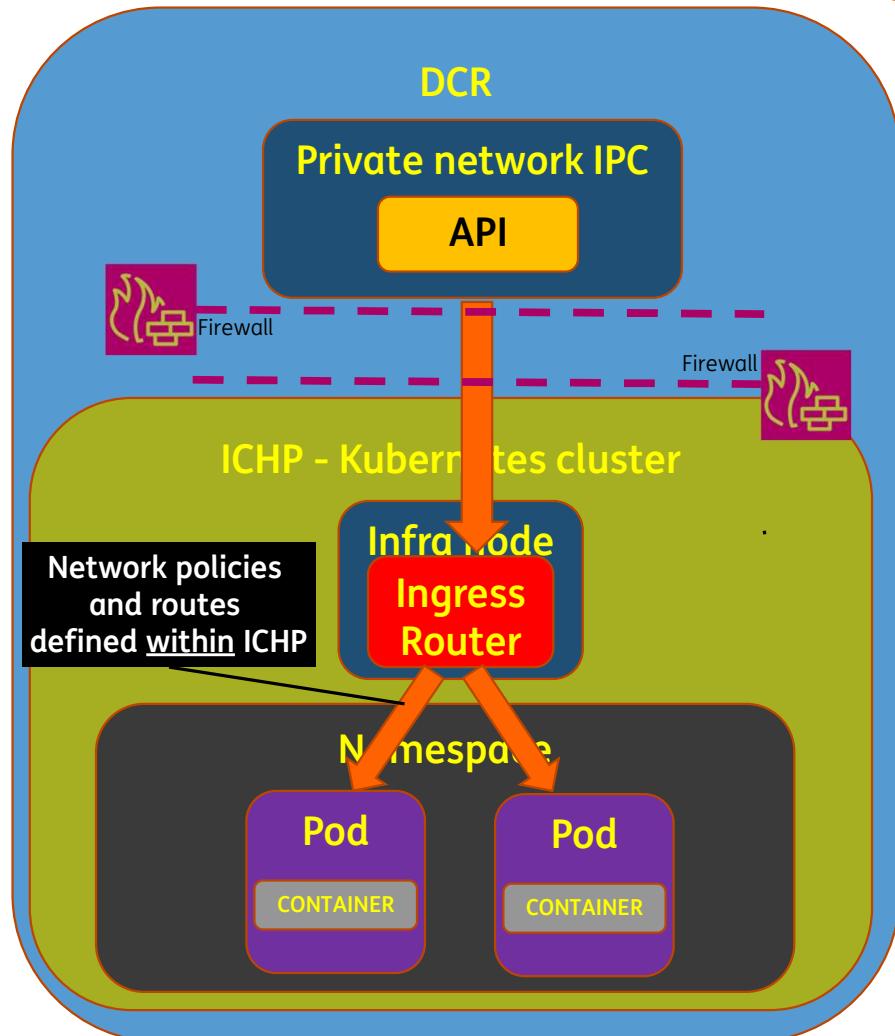
ICHP Design essentials



ICHP Network essentials - Ingress



ICHP Network essentials - Ingress



Network traffic to and from an ICHP namespace works a little bit different:

ALL incoming traffic (called **Ingress**) to ICHP has an Infra node (also known as ingress routers) as destination.

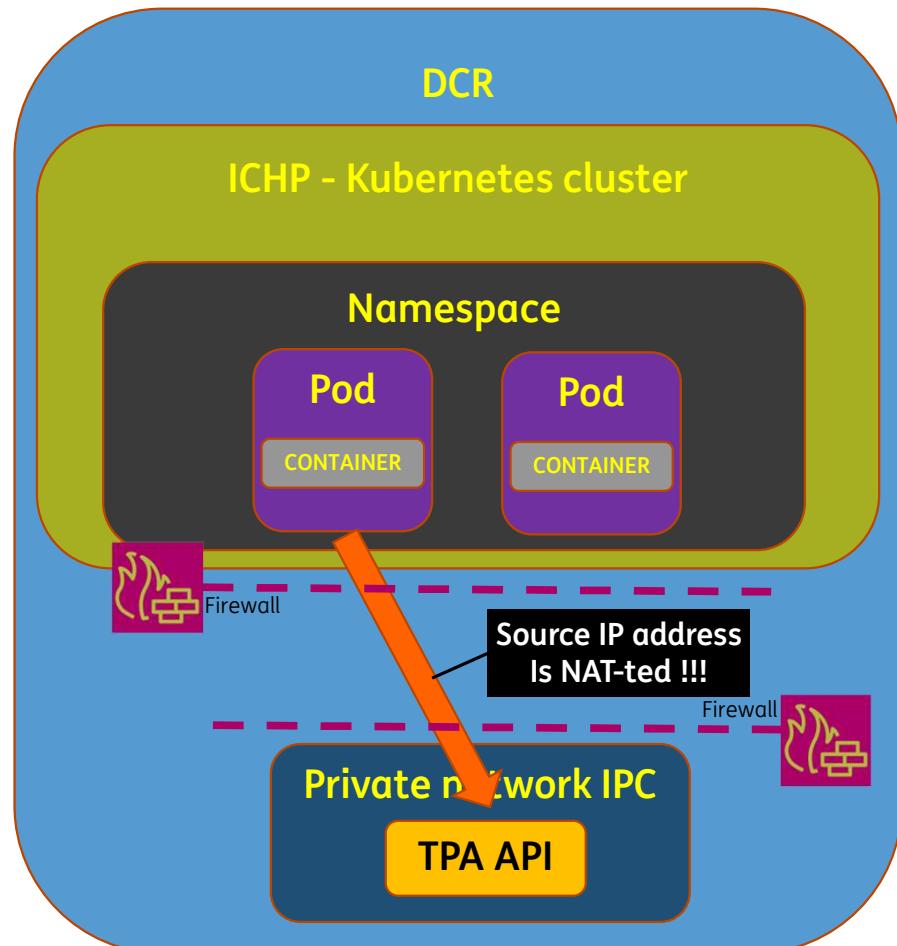
A firewall is used for the whole ICHP cluster but namespaces in ICHP are not traditional network zones, ie. they do NOT have a firewall. A network policy and route must exist in ICHP: these are managed by the Kingsroad pipeline.

For **traffic** from a network zone / a PN outside ICHP to a namespace within ICHP you need to configure **the ingress routers as destination** in the self-service portal, for example destination:

- *OpenShift Production on BMaaS DCR (new)*

ALWAYS check the latest ICHP network info on confluence/Forge!

ICHP Network essentials - Egress



Outgoing traffic from ICHP is called **Egress**.

To enable automation of outgoing traffic flows from an ICHP namespace to a private network in IPC:

- namespaces are defined **as** Private Networks in the self-service portal.
- **the IP addresses of your namespace (pods) are NAT-ted:** your outgoing IP-address is changed!

It is NOT 172.X.X.X

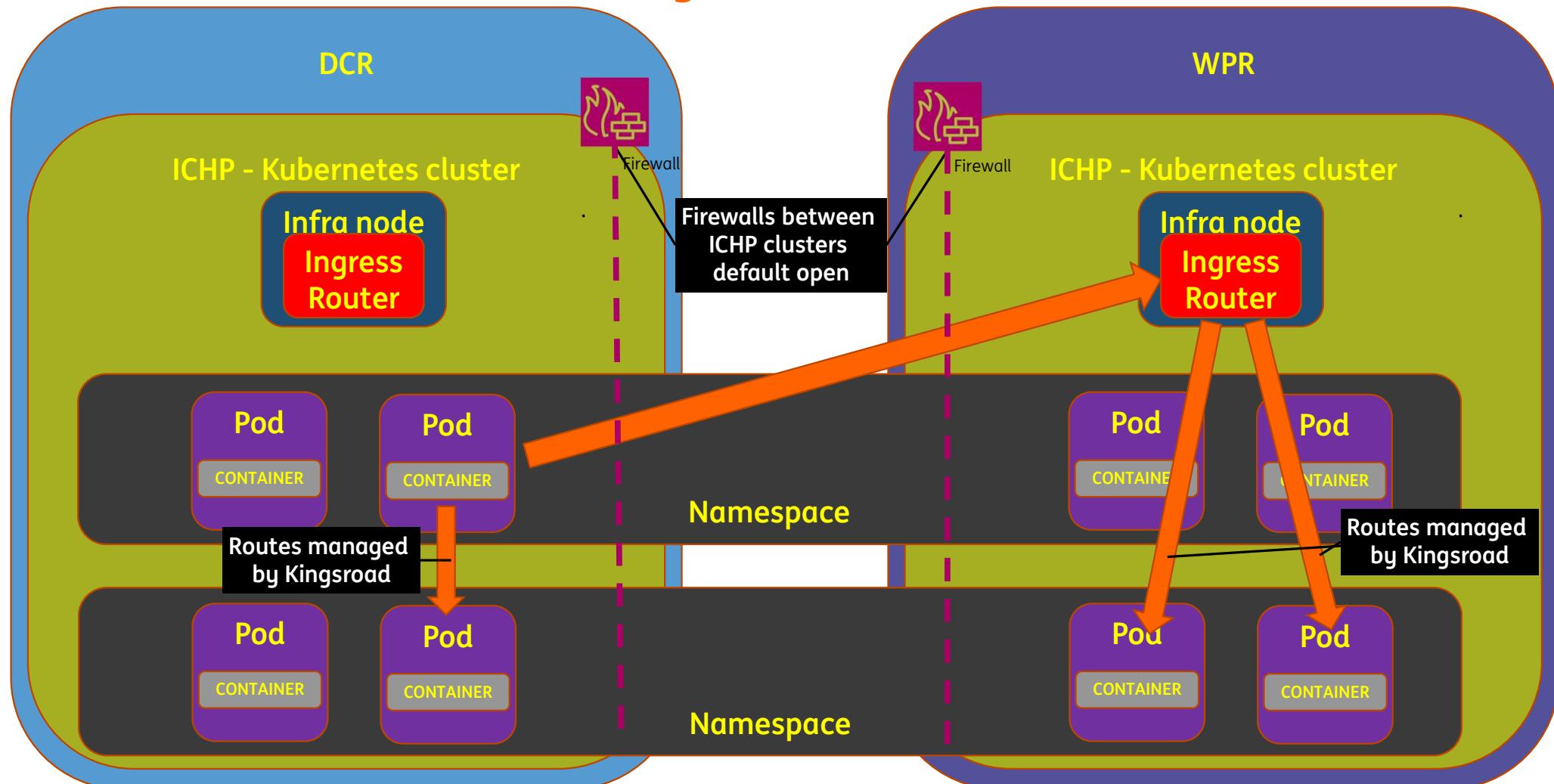
But 10.X.X.X

Do a NSLOOKUP, for example:

nslookup [aonlapi-prd.pod.dcr.prod.ichp.ing.net](#)

The **ICHPv2** DTA egress IP's resolve to DNS entries:
<namespace>-<environment>-1-<cluster number>[.egress.ichp.ing.net](#)

ICHP Network essentials - Ingress



ICHP connectivity issues

For ICHP network flows remember:

1. for Egress, NAT-ting is used for the source IP
2. For Ingress, the Ingress routers are used*

In ICHP V2 multiple Ingress routers are used for different purposes (TPA, non-TPA etc.), please check the Forge/Confluence

API Chain complexity



REST API

API chain complexity

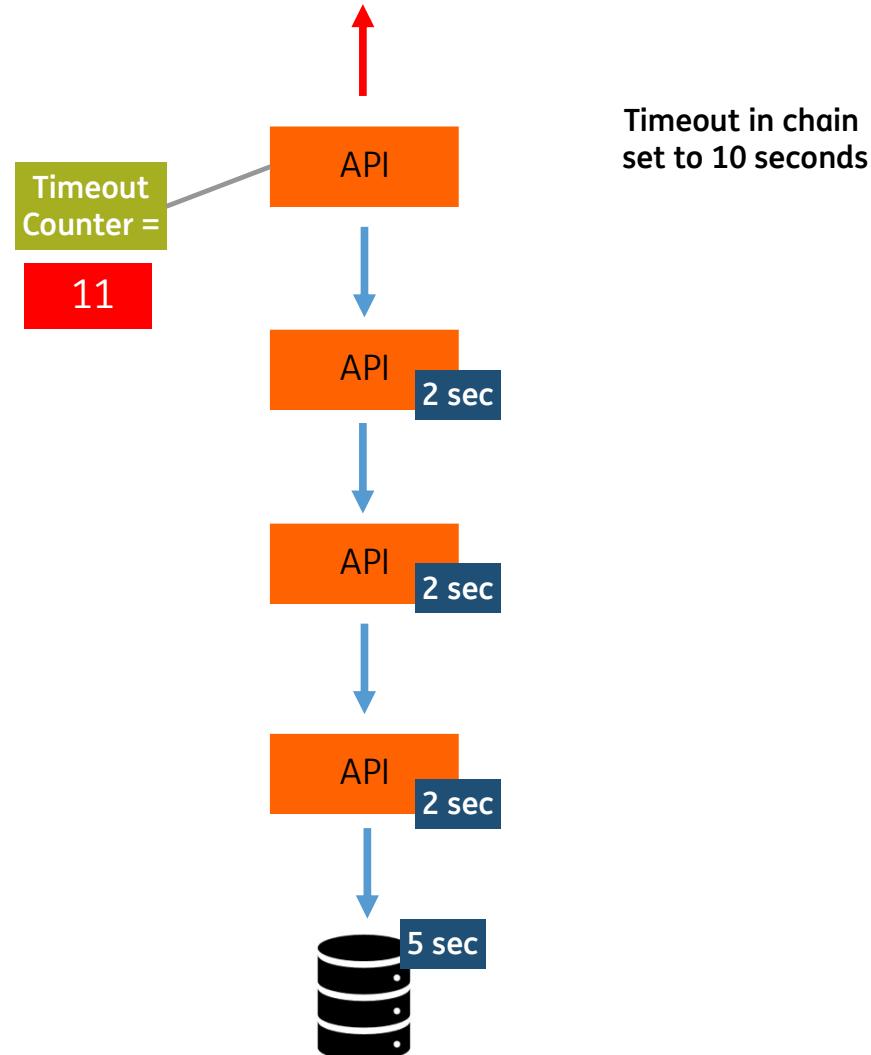
Especially where an **RR** Application Chain is concerned timeouts must be checked and tuned when necessary.

Incorrect timeouts in an application chain might lead to unexpected results or might even threaten the integrity of your data when a transaction involved.

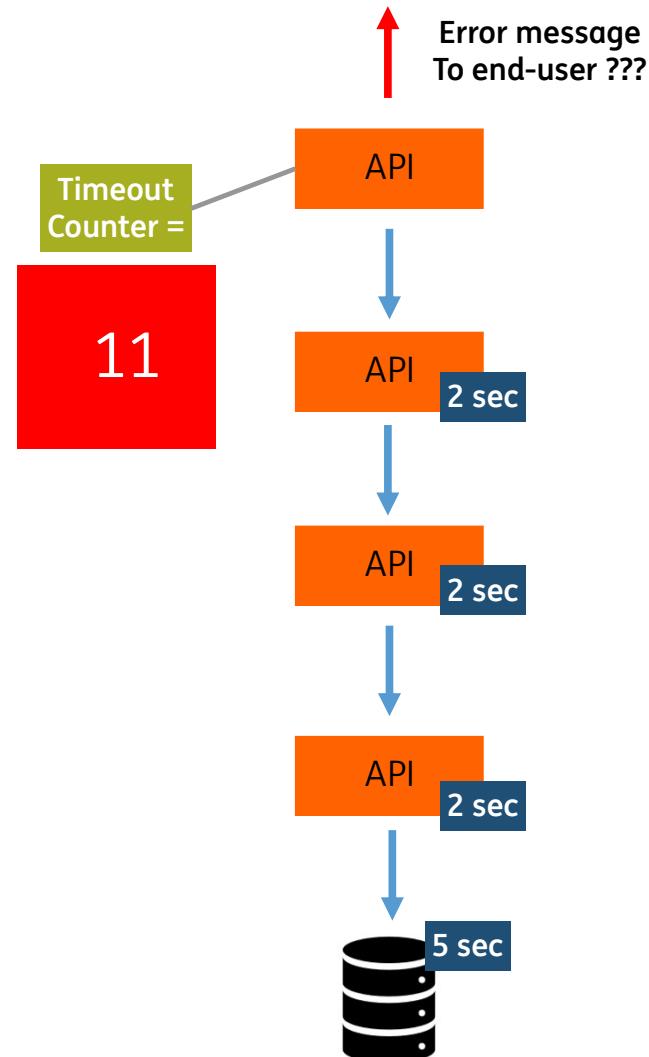
In this context I mean with a transaction:

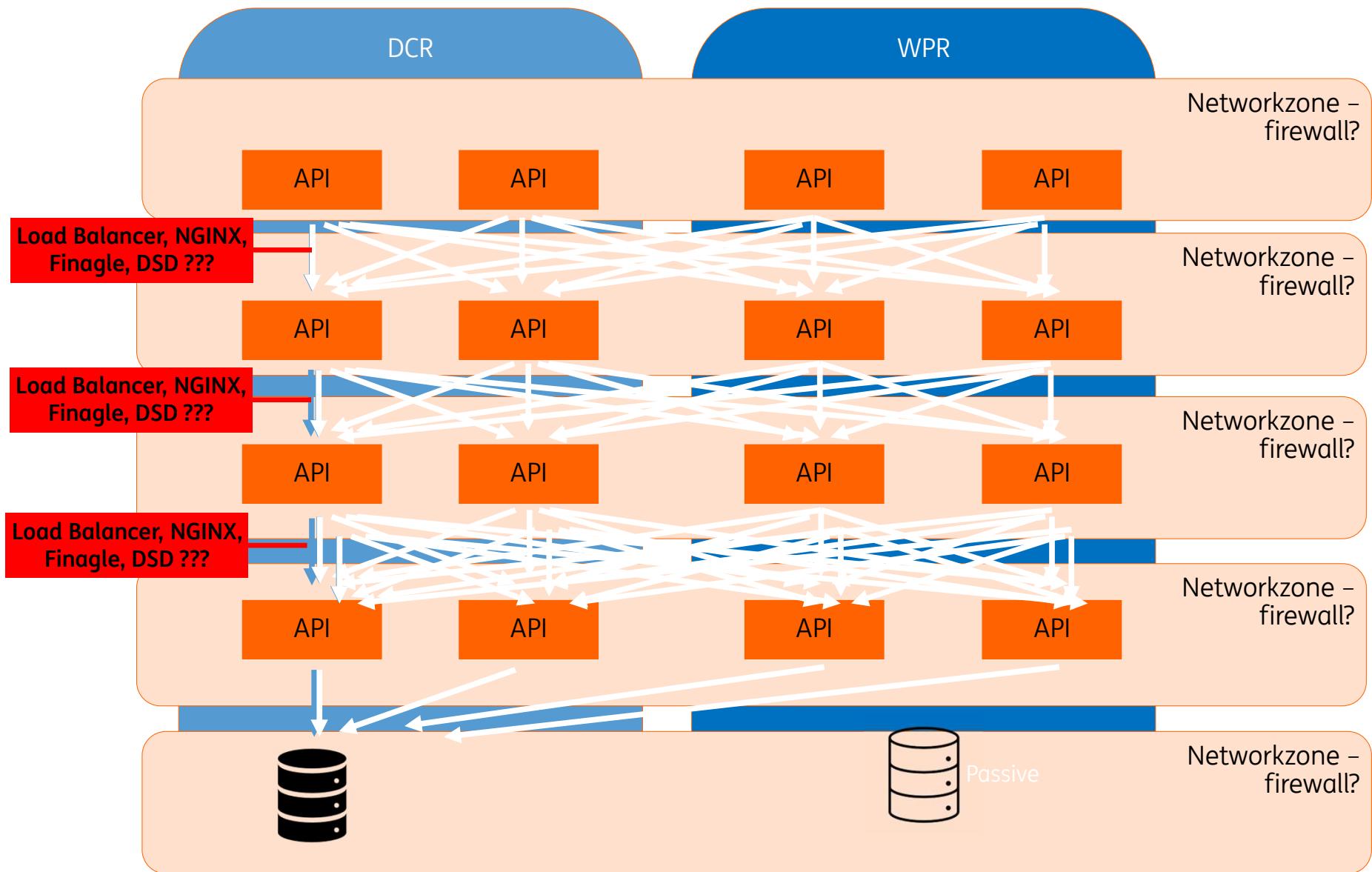
- A change, INSERT-UPDATE-DELETE, of data, not a READ/ GET of data.

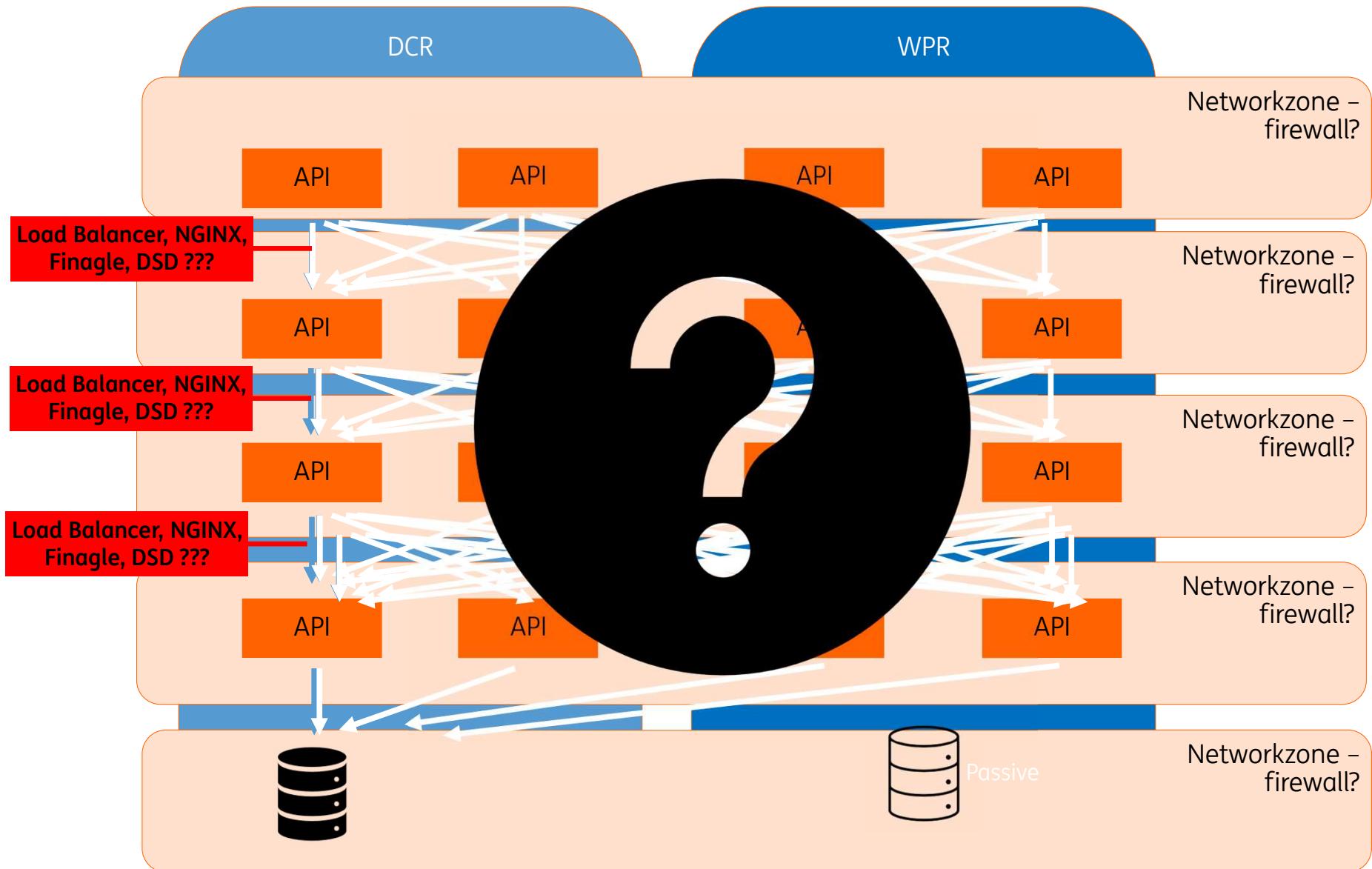
API chain timeouts



API chain timeouts







API chain complexity

It is notoriously difficult to handle a transaction in a long API chain and/or distributed environment

API chain complexity

Do you know your requesters?

Do you know your flow?
Including networkzones, firewalls etc.?

Do you know the complete chain from top to bottom?

Do you know the timeout settings and failover mechanisms used?

Can you observe the transaction in the chain?

Do you know the business functions you are part of?



SRE Observability: Tracing

It is crucial you can **observe** or **trace** the path of data requests across applications and networks.

A simple request to load a webpage might bounce between a dozen different applications.

At ING we have TracING, as part of the Merak libraries.

Make sure TracING is enabled in all your services!

API Chain complexity



REST API

API chain complexity

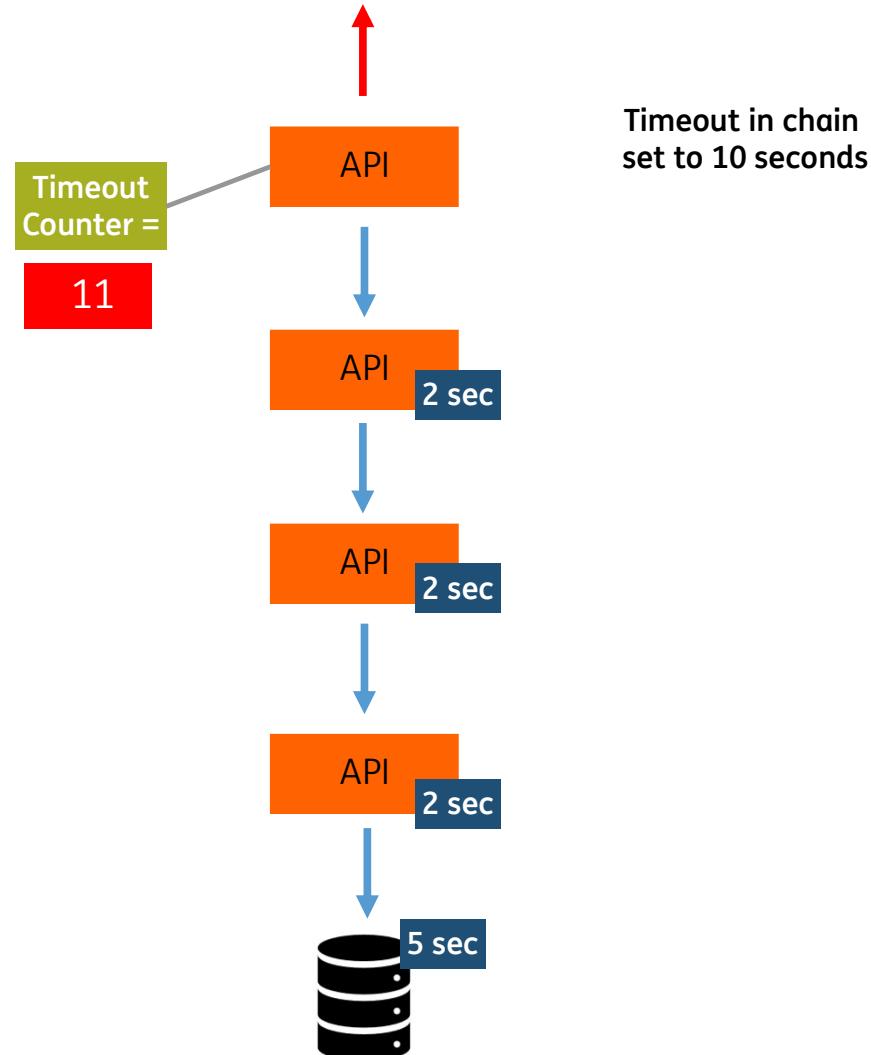
Especially where an **RR** Application Chain is concerned timeouts must be checked and tuned when necessary.

Incorrect timeouts in an application chain might lead to unexpected results or might even threaten the integrity of your data when a transaction involved.

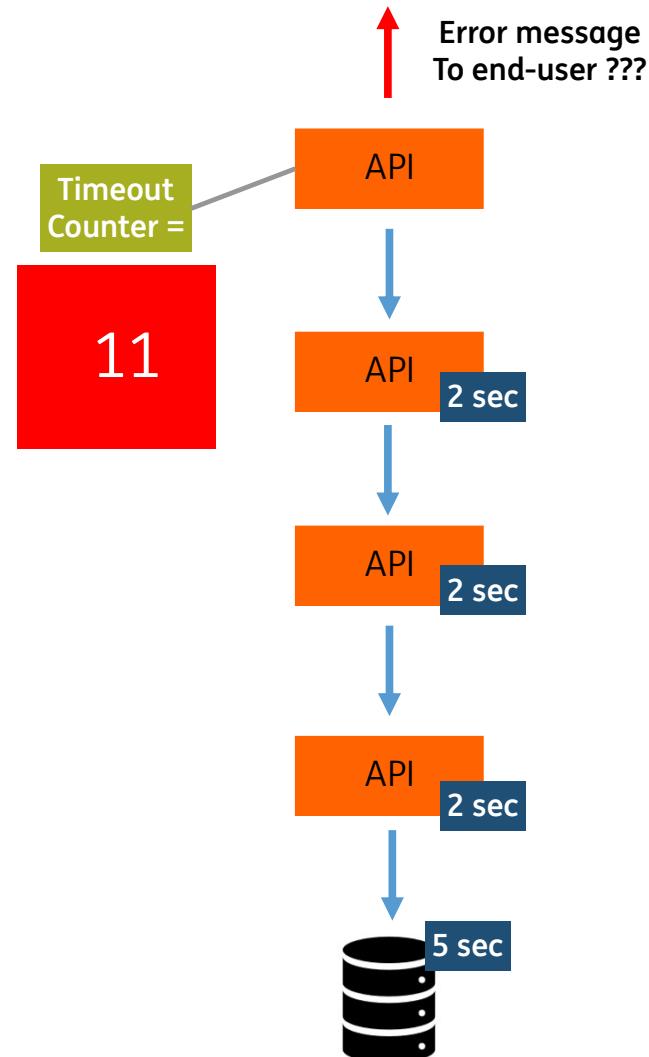
In this context I mean with a transaction:

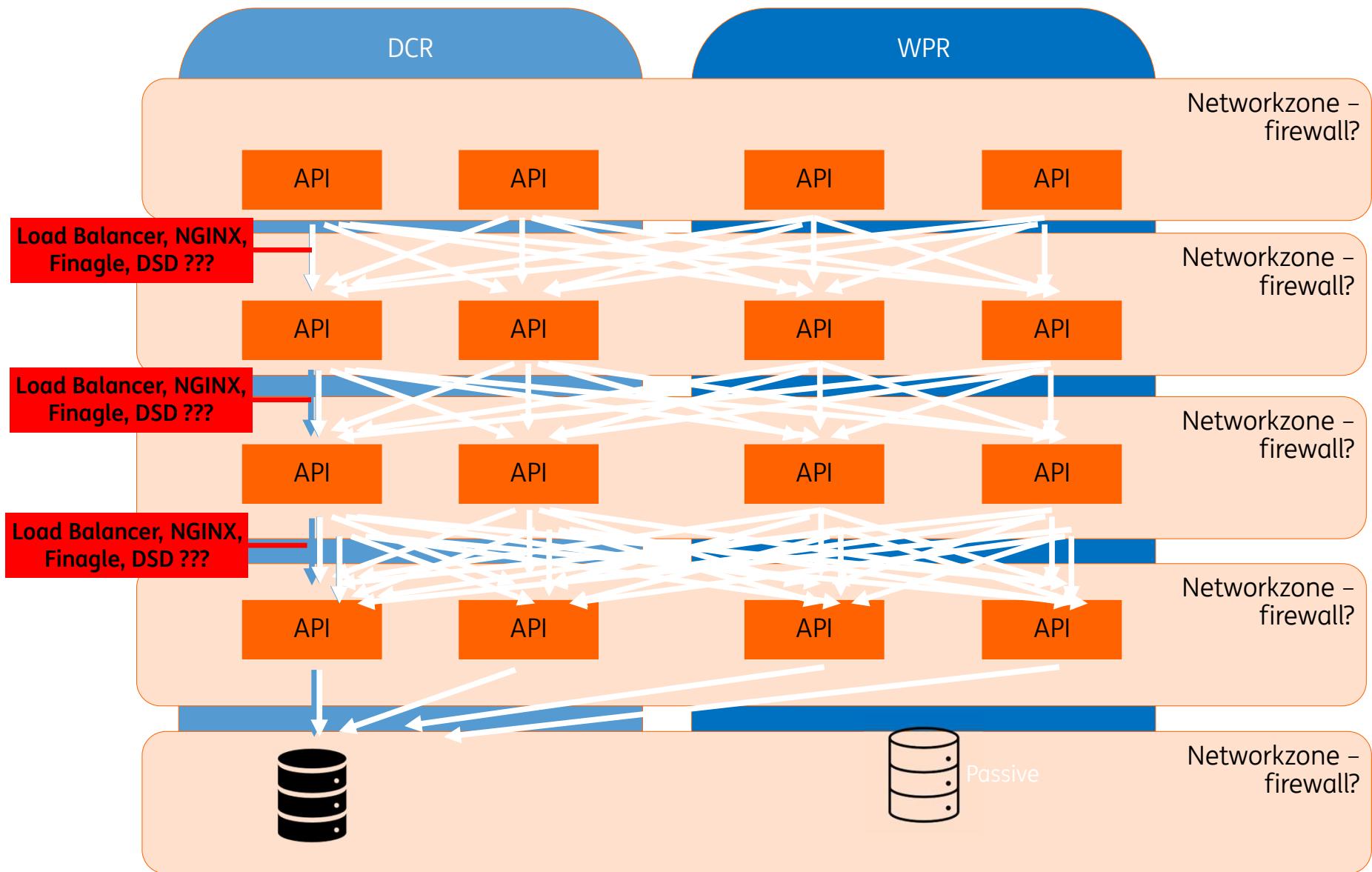
- A change, INSERT-UPDATE-DELETE, of data, not a READ/ GET of data.

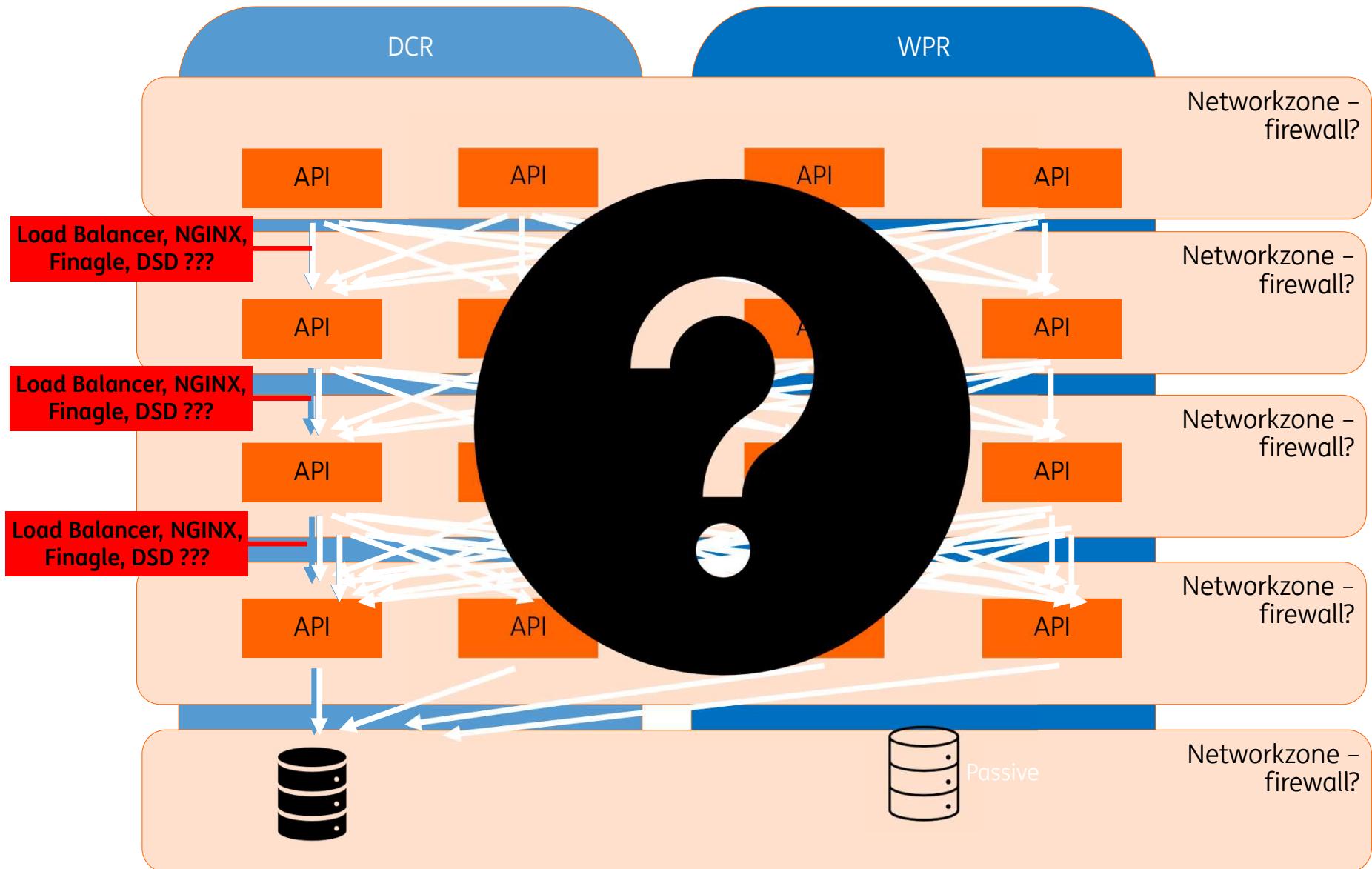
API chain timeouts



API chain timeouts







API chain complexity

It is notoriously difficult to handle a transaction in a long API chain and/or distributed environment

API chain complexity

Do you know your requesters?

Do you know your flow?
Including networkzones, firewalls etc.?

Do you know the complete chain from top to bottom?

Do you know the timeout settings and failover mechanisms used?

Can you observe the transaction in the chain?

Do you know the business functions you are part of?



SRE Observability: Tracing

It is crucial you can **observe** or **trace** the path of data requests across applications and networks.

A simple request to load a webpage might bounce between a dozen different applications.

At ING we have TracING, as part of the Merak libraries.

Make sure TracING is enabled in all your services!

API Chain complexity

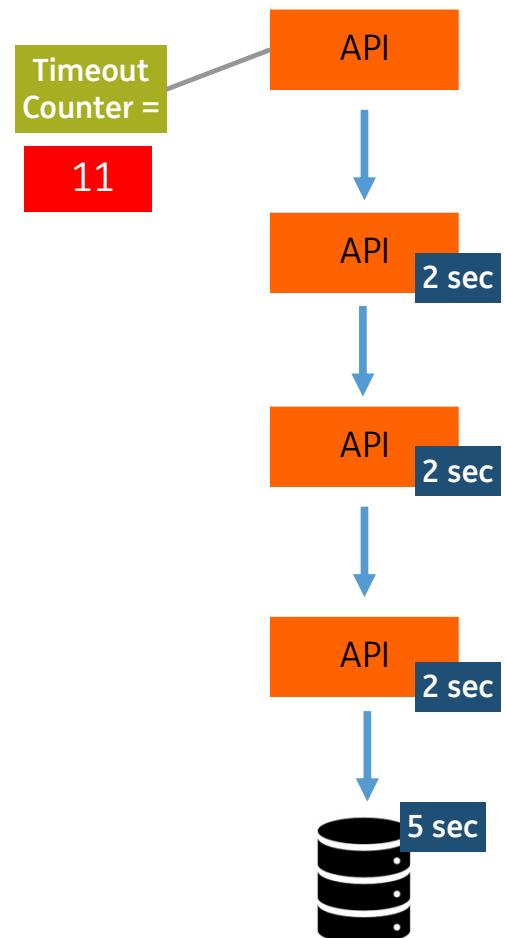


REST API

Major incidents – lessons learned

- *Timeouts usually setup incorrectly, using global defaults*
- *Timeouts too long, too many retries*
- *Systems collapse with “retry storms”*
- *Services doing work that can never be used : spending resources to which the clients are not listening anymore (due to timeouts)*
- *Some squads do not know the (network) flow*

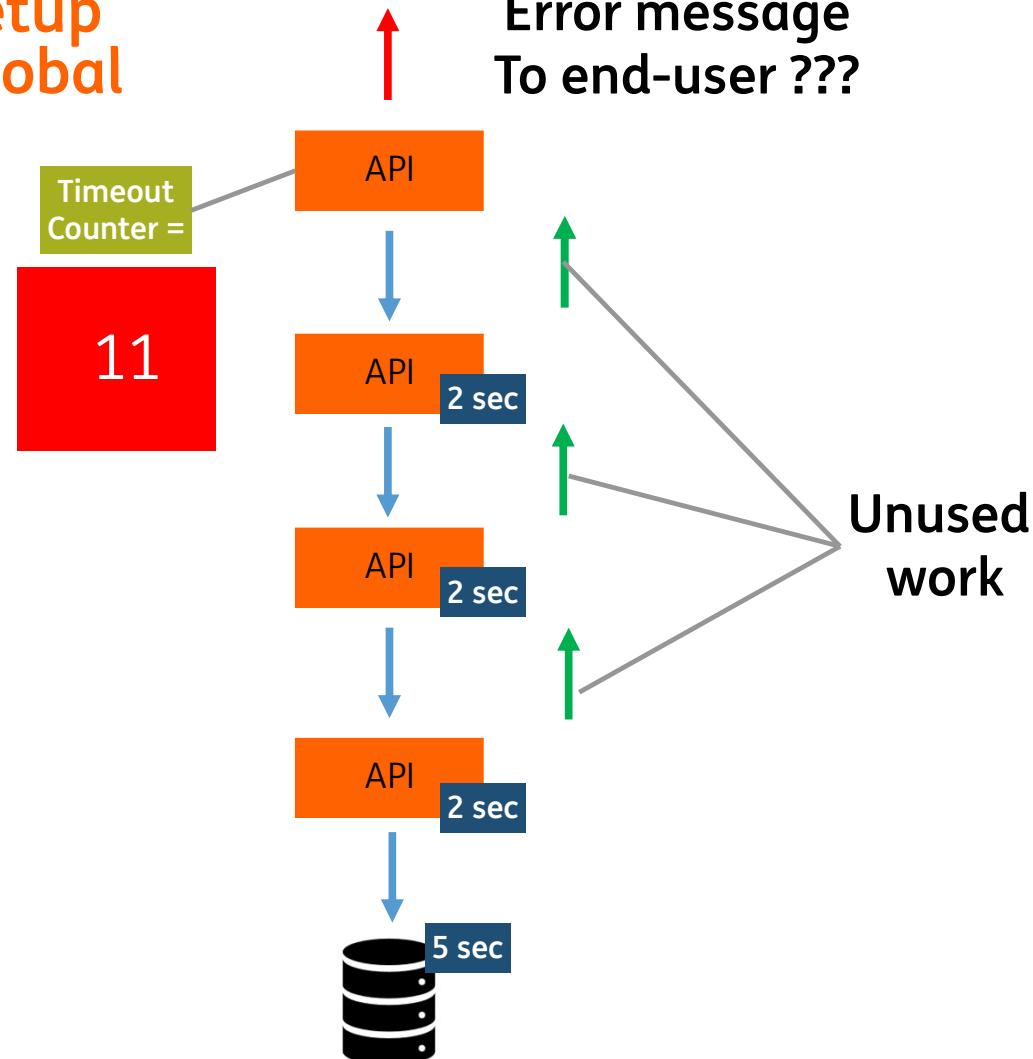
**Timeouts usually setup
incorrectly, using global
defaults**



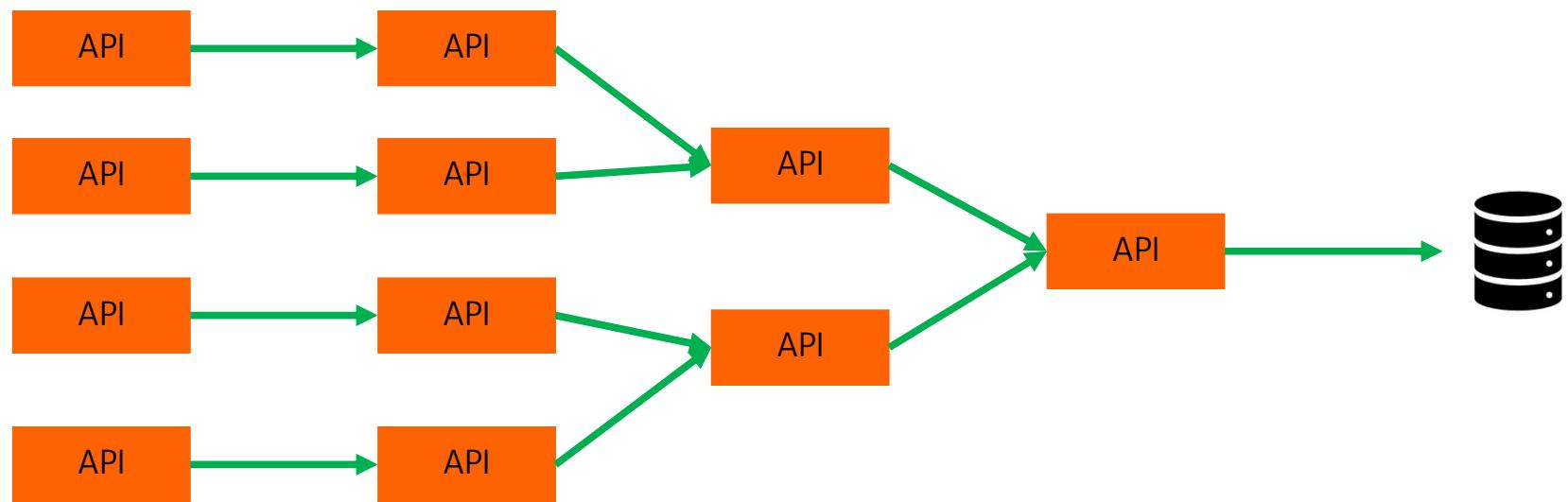
**Timeout in chain
set to 10 seconds**

Timeouts usually setup incorrectly, using global defaults

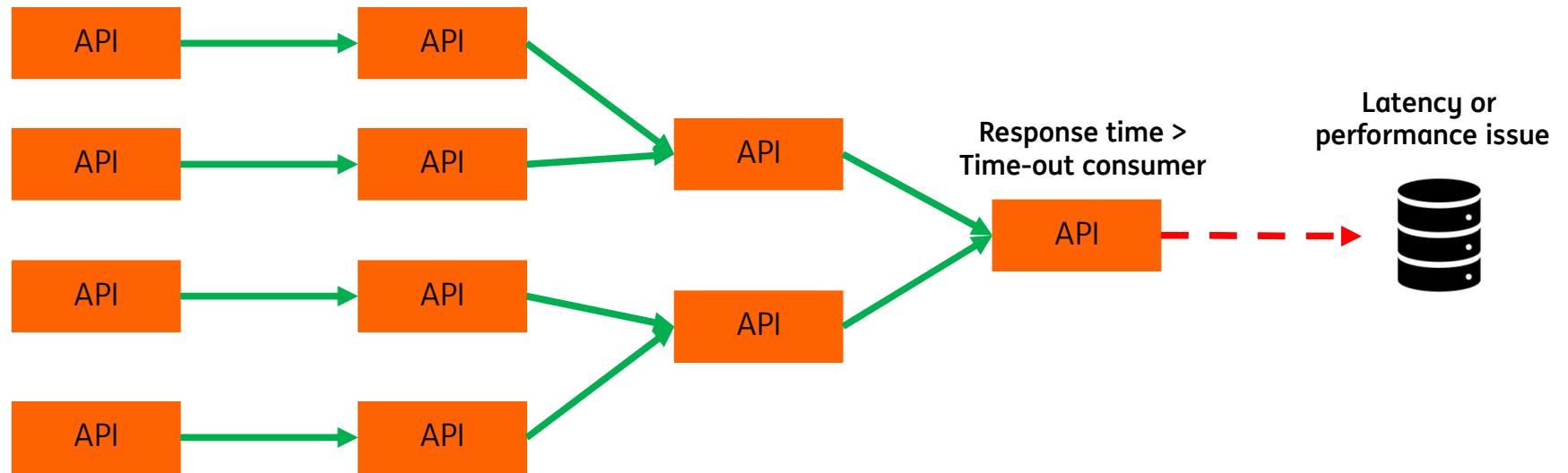
Error message
To end-user ???



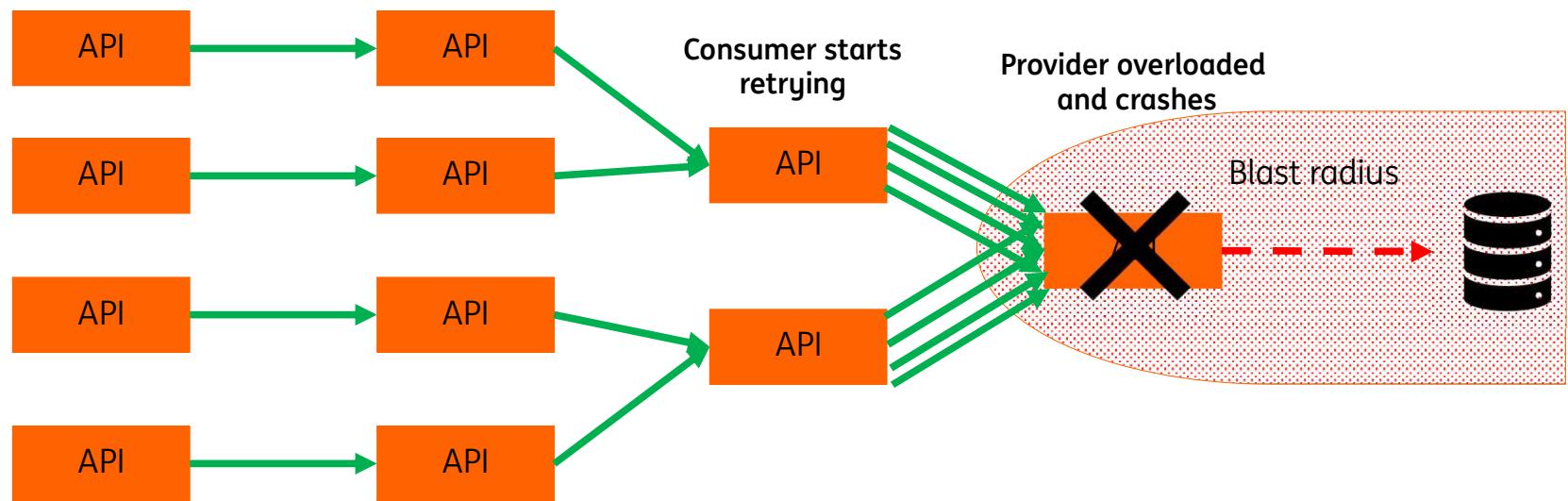
Retry storm



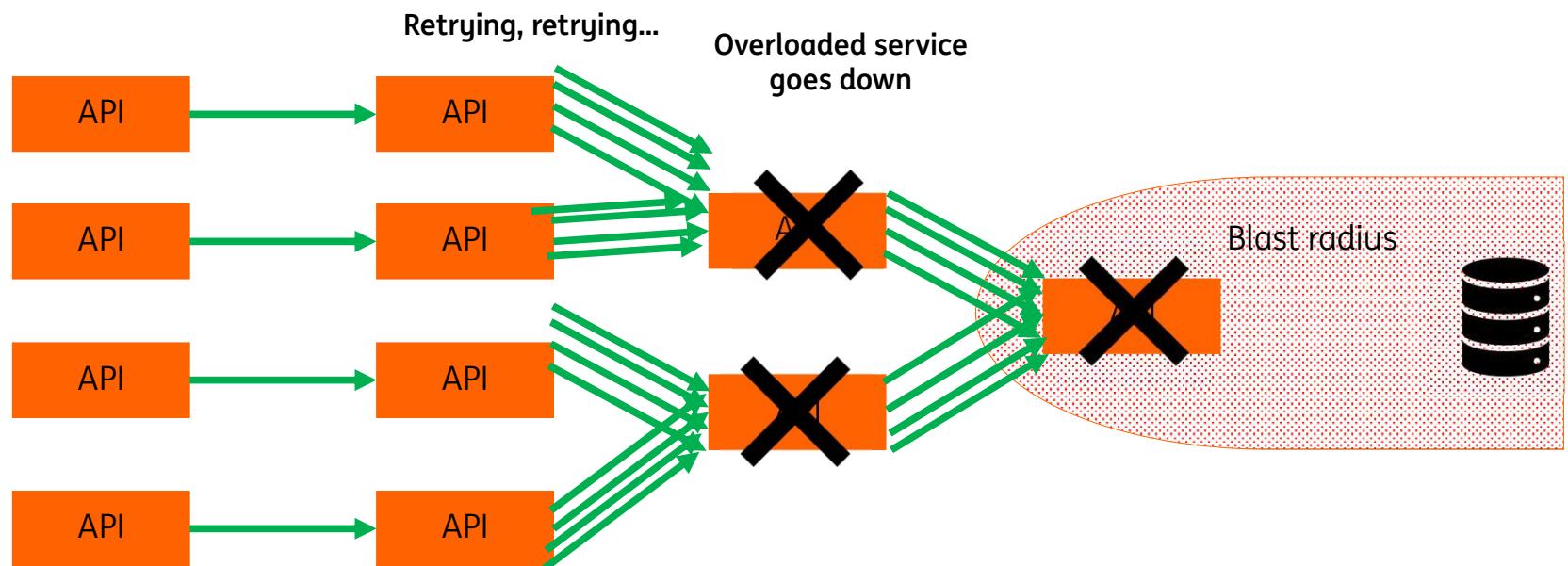
Retry storm



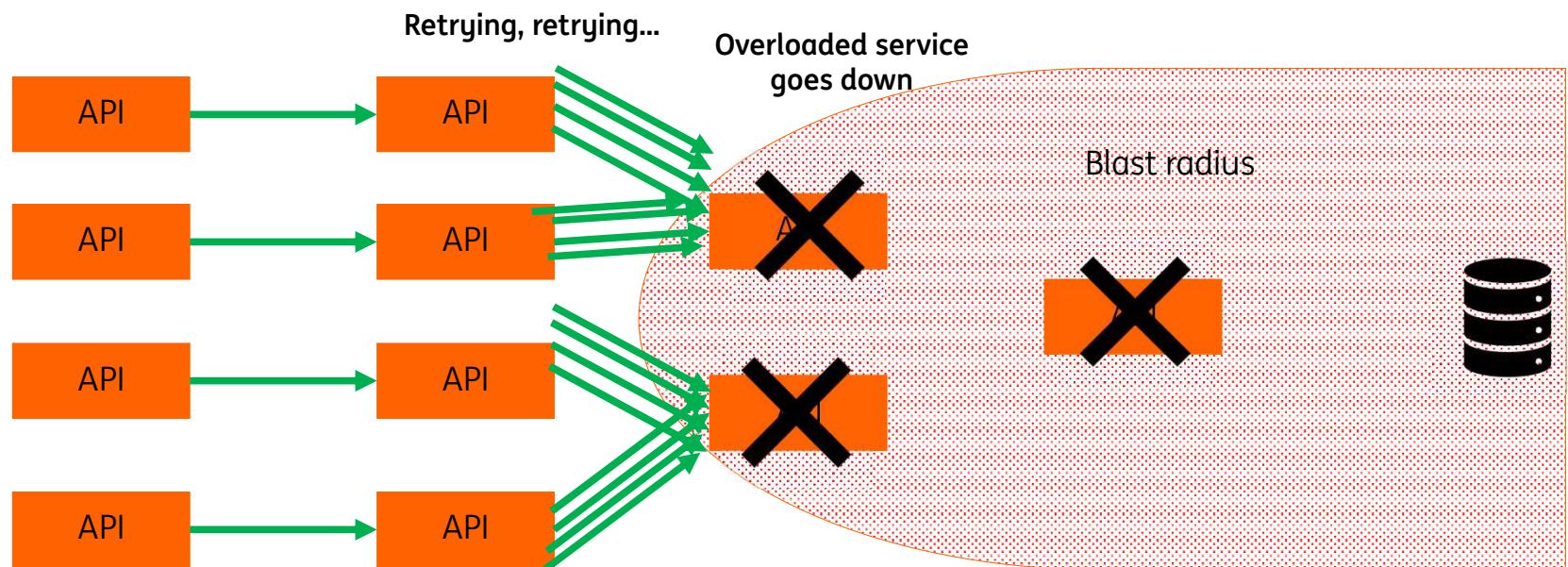
Retry storm



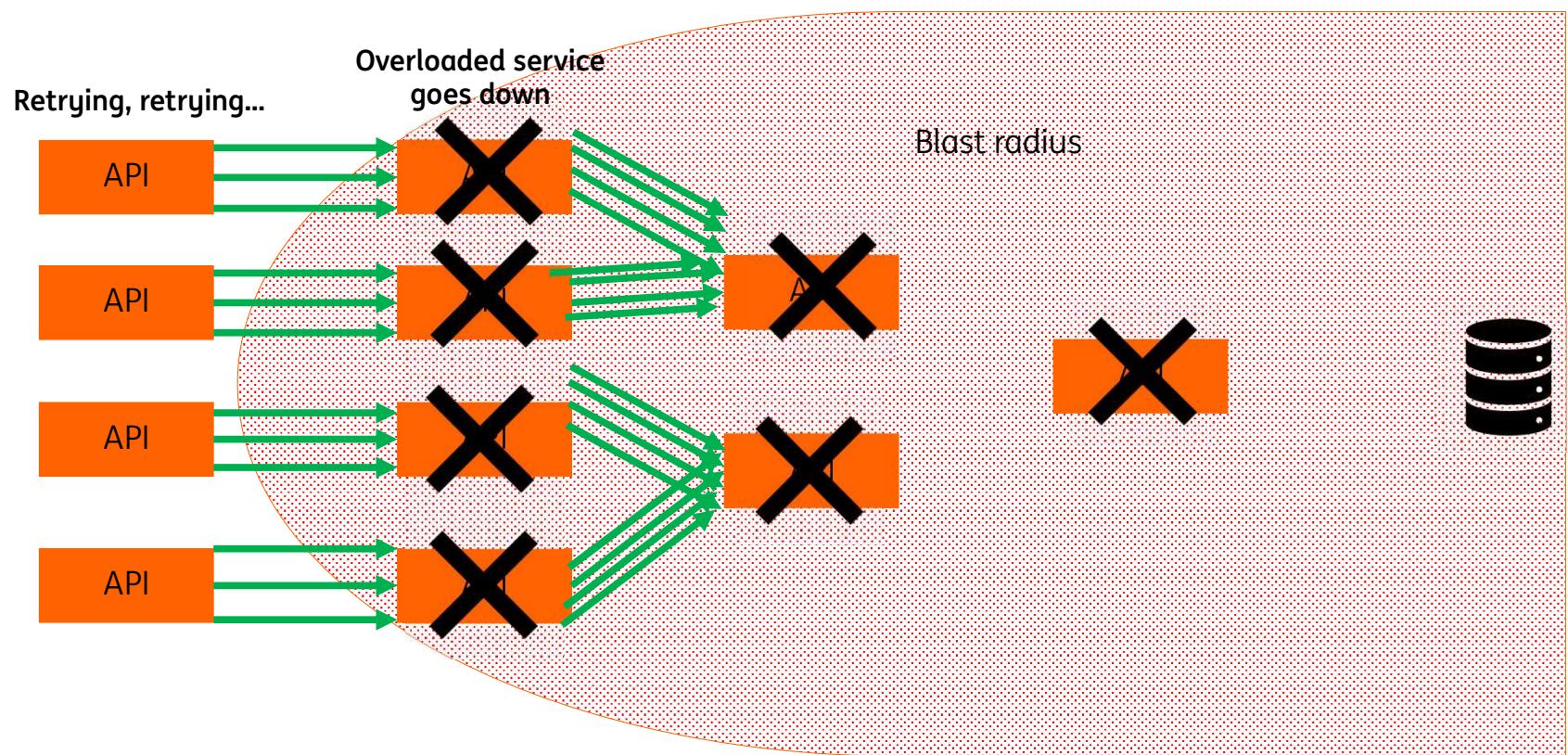
Retry storm



Retry storm



Retry storm

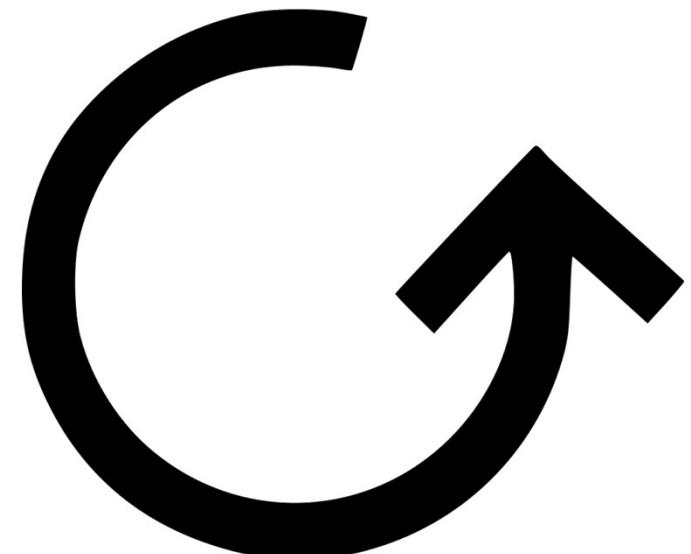


Overload prevention: setup timeouts & retries correctly

- Static timeout settings that decrease from the edge
- Align time-outs and retries in the chain
- Differentiate between a connection time-out and a request time-out

How often do retries actually succeed?

- Don't ask the same instance the same thing
- Only retry on a different connection
- Use an exponential back-off



Overload prevention: setup timeouts & retries correctly

Setting timeout settings that decrease from the edge and aligning time-outs and retries in the chain is difficult:

Do you know the chain, which chain, decrease how much?

Mechanisms for Time Outs and Retries come out of the box in Touchpoint. These are described on the extensive API SDK documentation on The Forge. The retry mechanism has an **exponential back-off** setting.



Overload prevention: Throttling

Control the consumption of resources used by an instance of an application or an entire service.

This can allow the system to continue to function and meet service level agreements, even when an increase in demand places an **extreme load** on resources.

Have an estimation on how much load your service can actually handle. **Stress tests / ceiling tests** and production metrics can help.



Examples throttling strategies:

- Rejecting requests from a consumer who's already accessed your API more than n times per second over a given period of time.
- Disabling or degrading the functionality of selected nonessential services

To improve resilience and mitigate retry storms two new features are introduced in the SDK:

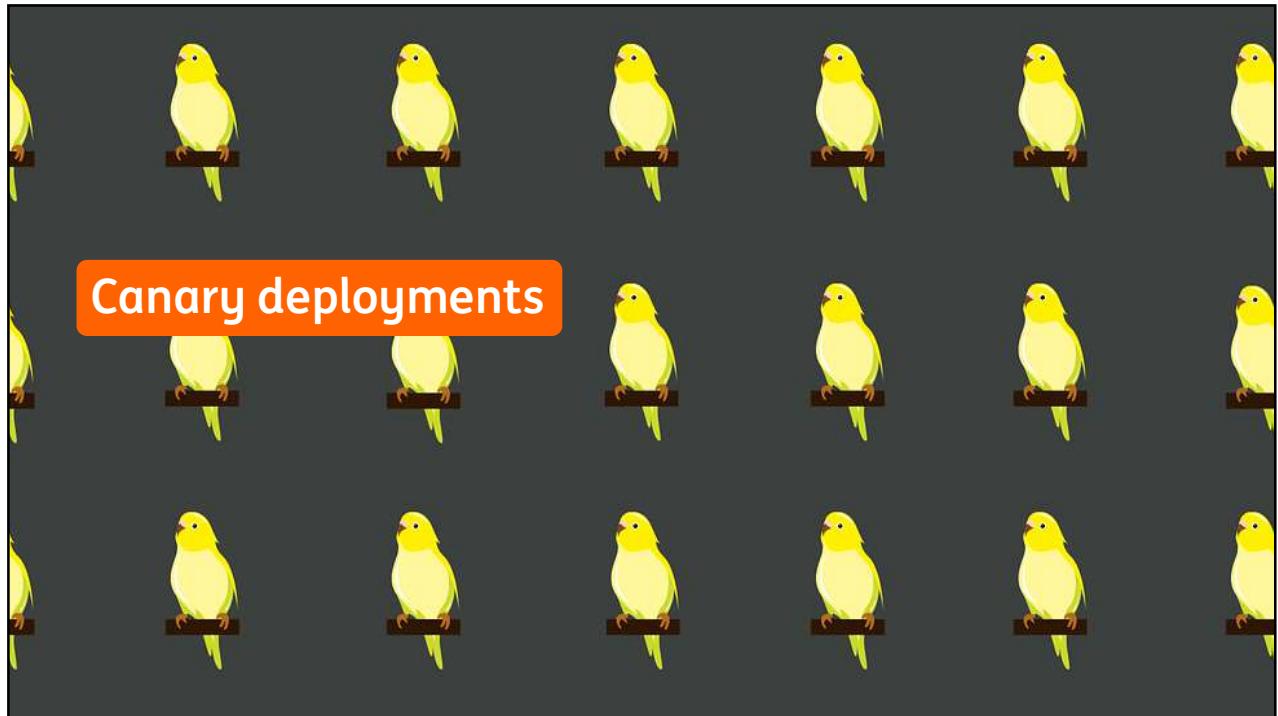
1. Deadlines (Experimental, 07-2023)

When acting on deadlines is enabled, Merak will compare the timestamp of the deadline to the current server time. If the time is passed it will reject the request to prevent the request from entering business logic and wasting resources. The request is rejected, and the client receives a response (even though they are no longer interested in the response) with the status code set to 408.

2. Backpressure (Experimental, 07-2023)

When backpressure is enabled, Merak will keep a count of the number of requests it can handle at the same time. This limit is calculated on each request based on the ability of the server to serve without errors.

When an incoming request is received then it will be served unless the number of in-flight requests is already at the edge of the server's ability. In that case the request is rejected, and the client receives a response with the status code set to 503



What is a canary release?

A canary release is a software testing technique used to reduce the risk related to deployment of a new software version into production by rolling out the changes to only small group of users (typically no more than 10 - 30% during change adoption phase) before actual production deployment.



2

<https://theforge.ing.net/product/45120/documentation/latest/how-to/canary-deploy>

Canary release using Kingsroad for ICHP

With Kingsroad, a canary deployment is essentially a deployment of a new version of your application restricted to only **1 instance (pod) running along with your current application deployment/version**.

The weight of **traffic load** that goes into the canary **is set automatically** and is equal to **1/(n+1)** (where n is the number of pods for your current production deployment/version)

The timeout for approving/rejecting the canary is set by default to 48hrs but you can modify it through the *canaryTimeout* setting
If the timeout is reached, then the canary is removed

NO Rollback is needed

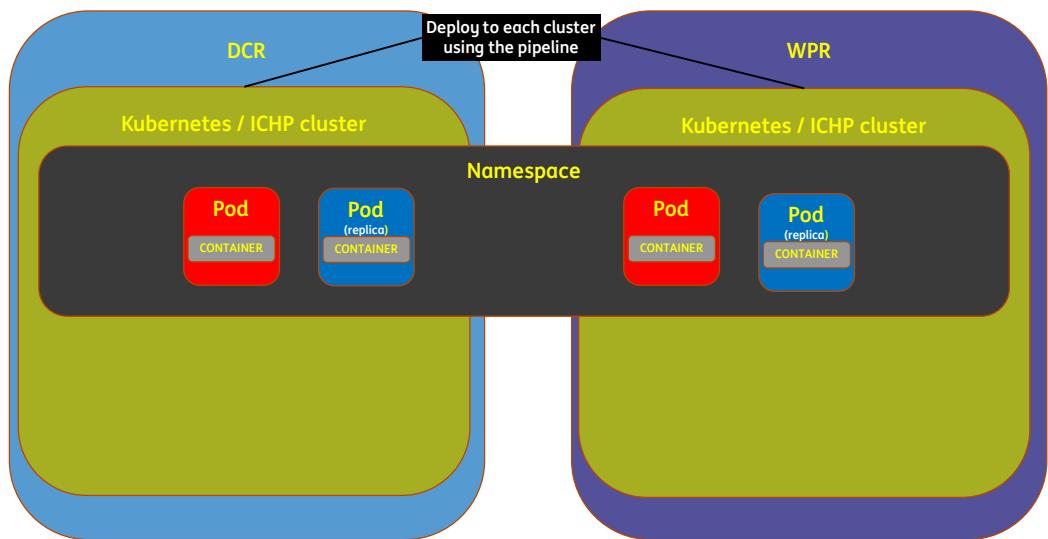
3

<https://theforge.ing.net/product/45120/documentation/8.19.0/how-to/canary-deploy>

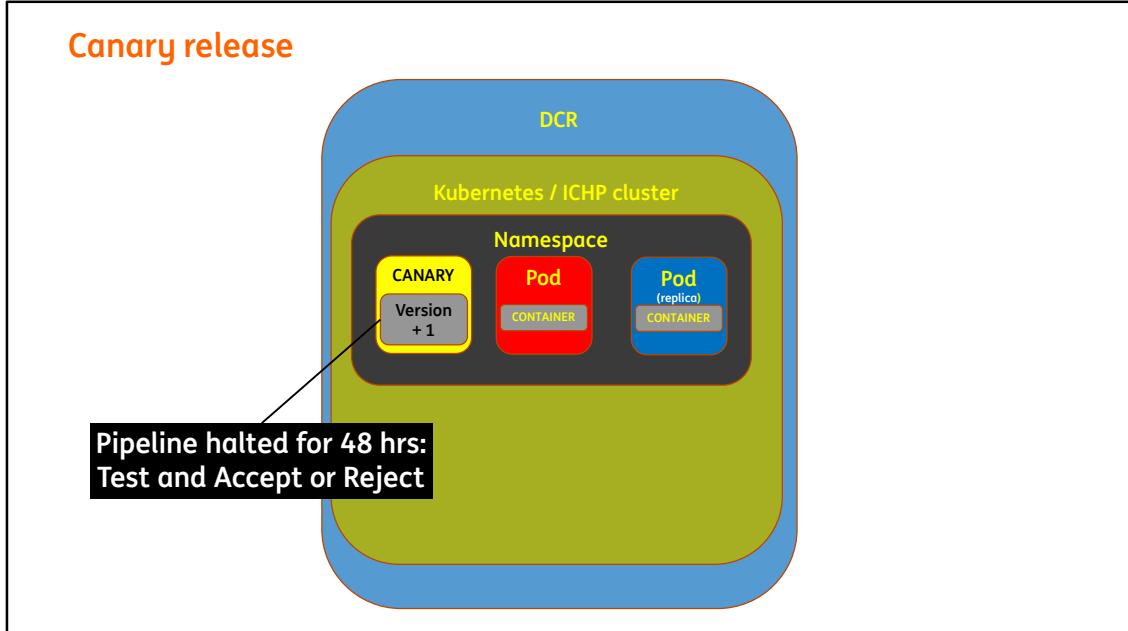
If you want to go through the canary “flow” on only one DC, you need to call the release library twice in the pipeline, once with “deploy_canary” action on one DC and once with “deploy” action on the other DC.

(how most teams do it) If you want to go through the canary flow on both DCs, you call release library only once with deploy_canary for both DCs and the canary will resume (after the user decides to move forward) on the second datacenter as well (not a bad thing in my opinion).

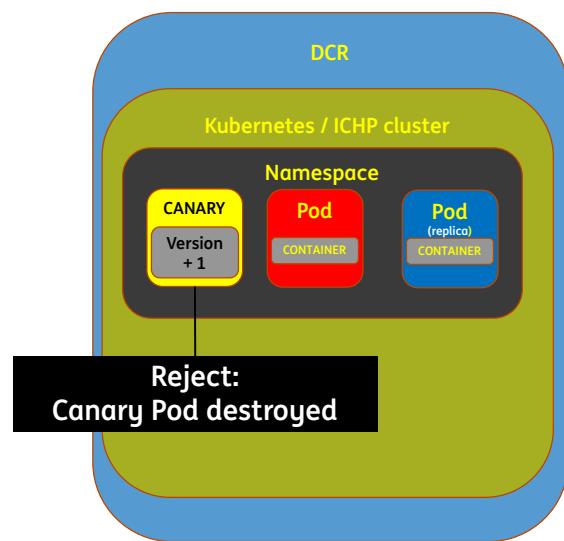
Canary release



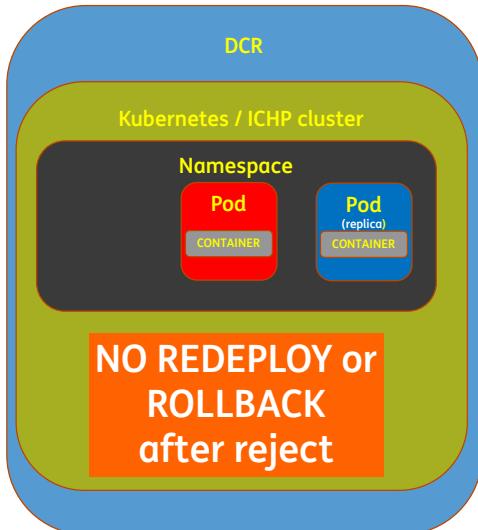
Canary release



Canary release



Canary release



Observability

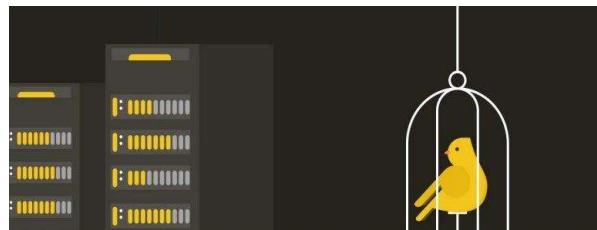
Please dont forget to create a dashboard for your canary release and monitor carefully!

Canary deployment on VMs

A canary release on **VMs** works differently:

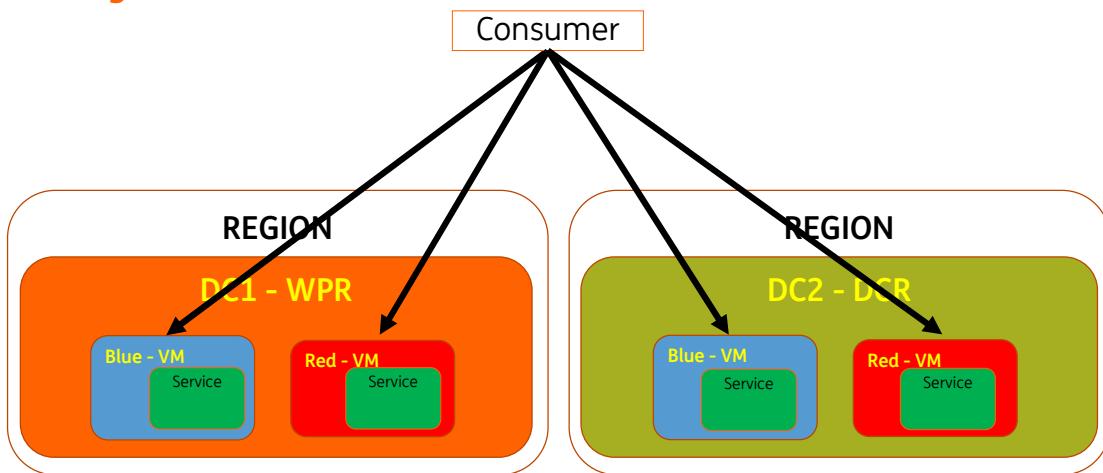
1. NO extra runtime is created (as is possible in ICHP/Kubernetes with an extra pod)
2. The rollback or rollout is **managed by the squad**. There is no procedure built in the pipeline for Canary releases.

However, **you can still use a Canary deployment!**



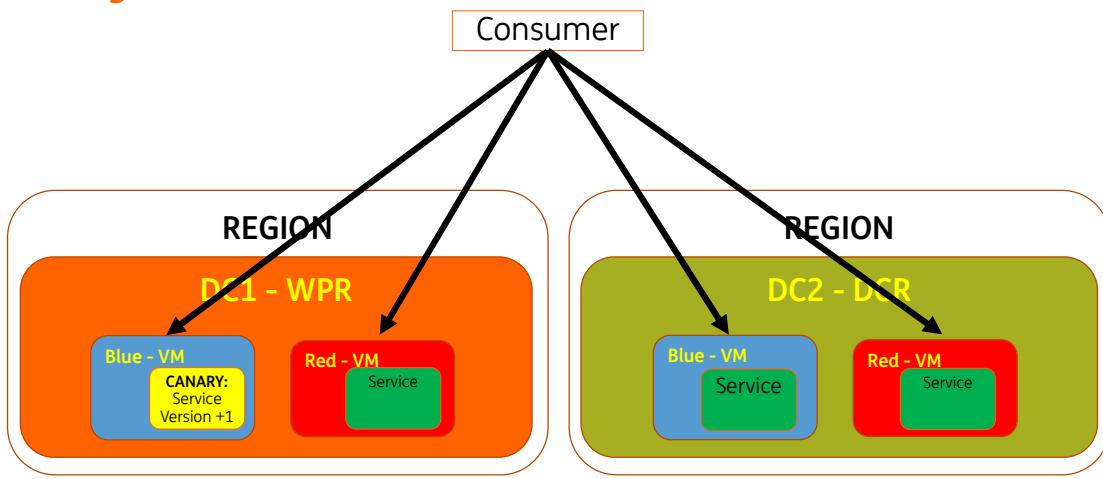
9

Canary on VMs

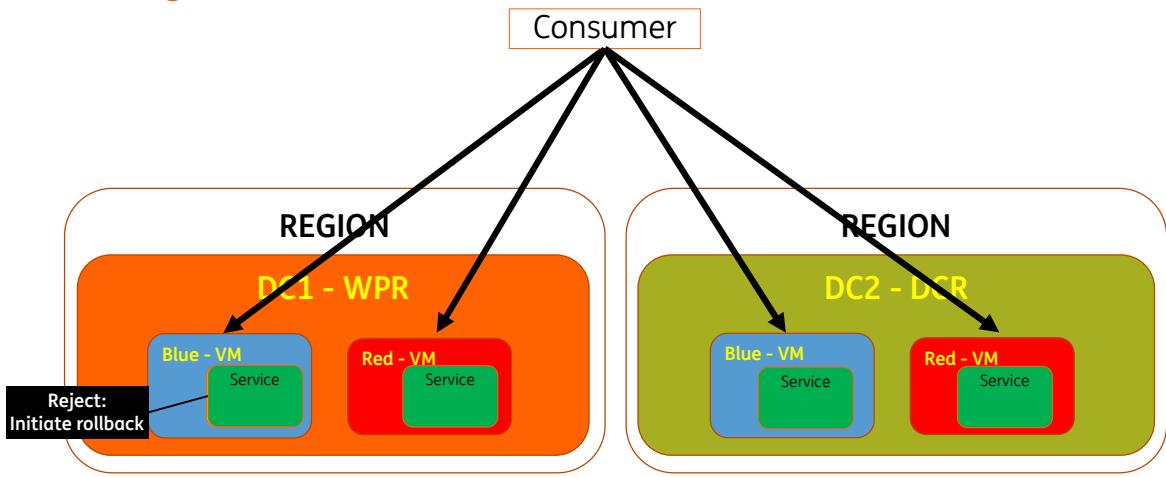


**Remember: each availability zone
must be able to handle the peak load!**

Canary on VMs

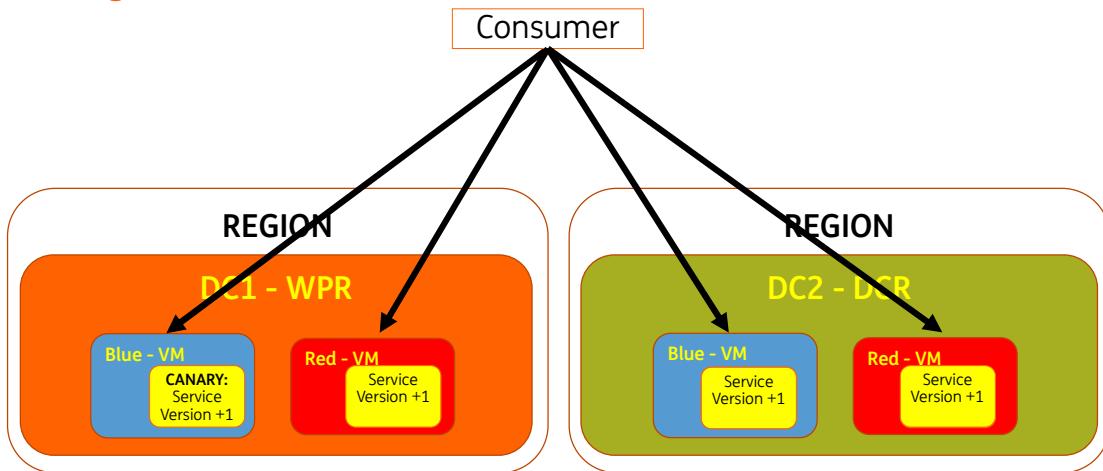


Canary on VMs



If NOT correct, initiate a rollback with the pipeline. A canary release is NOT integrated in the pipeline (like for ICHP)

Canary on VMs



If correct, rollout the new version to all runtimes

Canary deployments and data change

With a Canary deployment two versions of your service are running in parallel:

this should not be a problem, as a service must always be **backwards compatible**.

However, if your service is using a datastore or data service like DBaaS, SQLServer or KaaS things are **more complex** when a data change is in scope.

Still, scenarios and techniques exist to use a canary release (implicitly, do a **rollback**), for example using a tool like **Liquibase***.



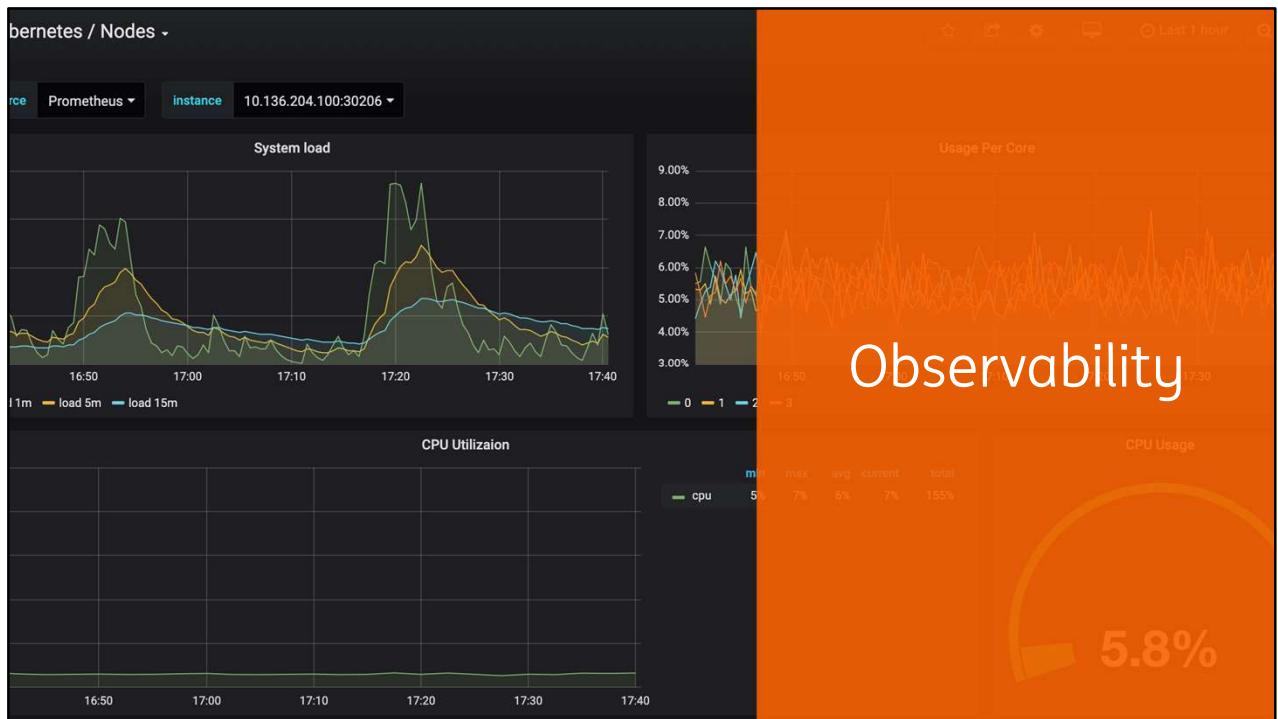
Even with a data change it is possible to use a Canary deployment. [Liquibase.org](https://liquibase.org)

*Please check the links in the notes of this slide for more information.

14

<https://academy.ing.net/learn/technology-foundation/episode-1/academy/evolutionary-database-design/README>

<https://theforge.ing.net/product/45120/documentation/8.28.0/templates/manage-db/manage-db-liquibase>



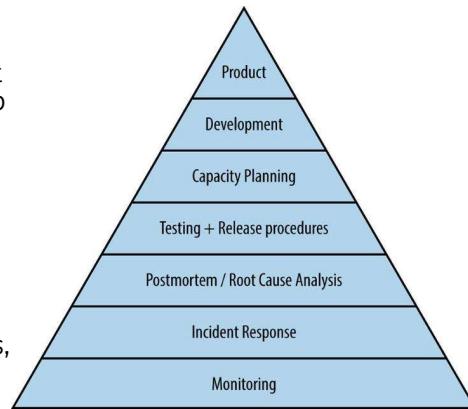
Service Reliability Hierarchy

Successfully operating a service entails a wide range of activities.

A **Site Reliability Engineering (SRE) pyramid**, also known as the 'Dickerson pyramid,' provides a set of principles that an organization can use to define and improve reliability to promote engineering excellence.

Site reliability engineering (SRE) is a software engineering approach to IT operations.

It builds upon the principles of DevOps to bring an engineering-led approach to IT operations. SRE uses software to automate system operation, identify problems, and implement resolutions.



The concept of SRE is developed at Google.

2

Because the SRE role is responsible for the reliability of systems in production, SREs are often required to be intimately familiar with a service's monitoring system and its features. Without this knowledge, SREs might not know where to look, how to identify abnormal behavior, or how to find the information they need during an emergency.

SLI's – SLO's – SLA's

Service Level Indicators (SLI's) are measurements of the characteristics of a service. The most common SLIs are the Four Golden signals (discussed later).

Service Level Objectives (SLO) specify a target level for the reliability of your service. Because SLOs are key to making data-driven decisions about reliability, they're at the **core of SRE practices**

Service Level Agreements (SLA) is an agreement between the service provider and customer about service deliverables.

Service level indicator

METRIC

Latency - time needed to process a request by a system



Service Level Objective

GOAL

95% of requests during the last 5 minutes should have latency <= 300 ms



Service Level Agreement

CONTRACT

The SLO should be met in 99% of all cases, and that will guarantee quick service latency



3

SLI as defined in Google's SRE Handbook: “*a carefully defined quantitative measure of some aspect of the level of service that is provided.*”

Example SLI's – SLO's

Category	SLI	SLO
API		
Availability	The proportion of successful requests, as measured from the load balancer metrics. Any HTTP status other than 500–599 is considered successful.	97% success
Latency	The proportion of sufficiently fast requests, as measured from the load balancer metrics. "Sufficiently fast" is defined as < 400 ms, or < 850 ms.	90% of requests < 400 ms 99% of requests < 850 ms

4

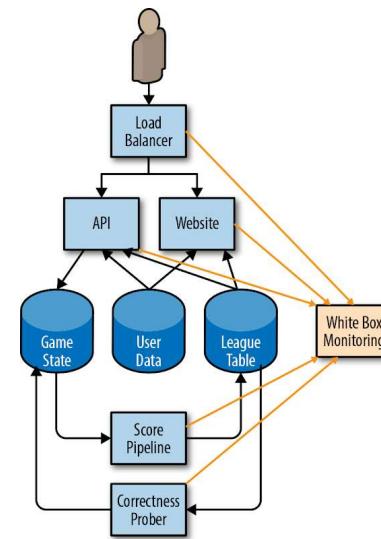
SLI as defined in Google's SRE Handbook: “*a carefully defined quantitative measure of some aspect of the level of service that is provided.*”

SLI implementation

- Find a measurable implementation of your service outcome that you think **matters to users**.
- Make sure that the specification includes an **event**, **success criteria** and **where/how it is measured**.
- Use **white-box monitoring** to collect SLI metrics from the various components of your application

The SLI is best expressed as an equation between good events that fulfil the success criteria and all valid events (e.g. you might want to exclude HTTP 500 errors as invalid events)

$$\text{SLI} = \left(\frac{\text{good events}}{\text{valid events}} \right) \times 100\%$$



5

Note that, typically SLIs are not infrastructure metrics (CPU, load, memory usage, network bandwidth), but should rather be directly connected to user happiness

Of course, you should still be collecting infrastructure metrics to be able to debug and identify root-cause of an outage, feed into AIOps to detect and correlate anomalies in order to be able to prevent and quickly understand SLI degradation.

Error Budget

An SLO sets a target level of reliability for the service's customers. Above this threshold, almost all users should be happy with your service (assuming they are otherwise happy with the utility of the service). Below this threshold, users are likely to start complaining or to stop using the service.



$$\frac{99.9\% T}{\text{month}} = \frac{0.1\% L}{\text{month}} = 0.001 \times \frac{30 \text{ day}}{\text{month}} \times \frac{24 \text{ hour}}{\text{day}} \times \frac{60 \text{ min}}{\text{hour}} = \frac{43.2 \text{ min}}{\text{month}}$$

Error Budget is the amount of error that your service can accumulate over some period of time before your users start being unhappy.

You can think of it as the pain tolerance for your users but applied to a certain dimension of your service: availability, latency, and so forth.

The **error budget** gives the number of allowed bad events, and the **error rate** is the ratio of bad events to total events.

6

<https://sre.google/workbook/alerting-on-slos/#low-traffic-services-and-error-budget-alerting>

Alerting on SLOs

You need to turn your SLOs into (actionable) **alerting rules**.

Your goal is to be notified for a significant event: an event that consumes a large fraction of the **error budget**.

Consider the following attributes when evaluating an alerting strategy:

- **Precision:** the proportion of events detected that were significant. Precision is 100% if every alert corresponds to a significant event.
- **Recall:** the proportion of significant events detected. Recall is 100% if every significant event results in an alert.
- **Detection time:** how long it takes to send notifications in various conditions. Long detection times can negatively impact the error budget.
- **Reset time:** how long alerts fire after an issue is resolved. Long reset times can lead to confusion or to issues being ignored.



7

<https://sre.google/workbook/alerting-on-slos/#low-traffic-services-and-error-budget-alerting>

What is observability?

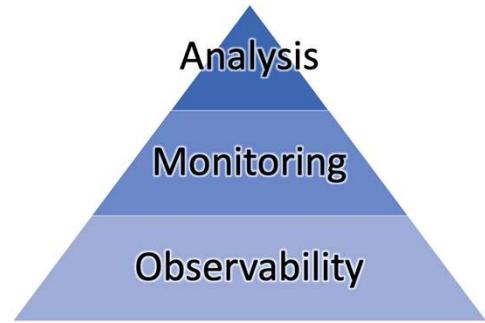
Observability is the ability to measure a system's current state based on the data it generates, such as **logs, metrics, and traces**.

It allows teams to detect problems proactively and resolve issues faster.

Observability tools collect and analyze data, user experience, infrastructure, and network telemetry to resolve issues before they impact business key performance indicators (KPIs).

The more observable a system, the more quickly and accurately you can navigate from an identified performance problem to its root cause, without additional testing or coding.

The architects and developers who create the software must **design it to be observed**.



8

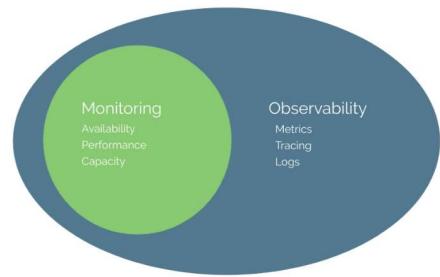
In software, observability refers to telemetry produced by services. Observability is divided into three major verticals—metrics, logs, and distributed traces—the so-called three pillars of observability. Thanks to projects such as OpenTelemetry, which promotes the standardization of data collection, and W3C trace-context, built-in telemetry will soon become a must-have feature of cloud-native software..

Because the SRE role is responsible for the reliability of systems in production, SREs are often required to be intimately familiar with a service's monitoring system and its features. Without this knowledge, SREs might not know where to look, how to identify abnormal behavior, or how to find the information they need during an emergency.

Observability & monitoring

Monitoring tells you when something is wrong, while observability can tell you what's happening, why it's happening and how to fix it.

- Observability provides **context** around what is wrong with a system. Monitoring, for the most part, offers surface-level information about the existence of a problem.
- Monitoring helps engineers/SREs determine that a problem exists. Observability goes further by allowing them to understand **why** it exists.



After you've made the system observable, and after you've collected the data using a monitoring tool, you must perform **analysis** either manually or automatically.

The better your analysis capabilities are, the more valuable your investments in observability and monitoring become.

9

In a monitoring scenario, you typically preconfigure dashboards that are meant to alert you to performance issues you expect to see later. However, these dashboards rely on the key assumption that you're able to predict what kinds of problems you'll encounter before they occur.

In an observability scenario, where an environment has been fully instrumented to provide complete observability data, you can flexibly explore what's going on and quickly figure out the root cause of issues you may not have been able to anticipate.

Why we need observability



10

Challenges with observability:

- **Data silos:** Multiple agents, disparate data sources, and siloed monitoring tools make it hard to understand interdependencies across applications, multiple clouds, and digital channels, such as web, mobile, and IoT.
- **Volume, velocity, variety, and complexity:** It's nearly impossible to get answers from the sheer amount of raw data collected from every component in ever-changing modern cloud environments, such as AWS, Azure, and Google Cloud Platform (GCP). This is also true for Kubernetes and containers that can spin up and down in seconds.
- **Manual instrumentation and configuration:** When IT resources are forced to manually instrument and change code for every new type of component or agent, they spend most of their time trying to set up observability rather than innovating based on insights from observability data.
- **Lack of pre-production:** Even with load testing in pre-production, developers still don't have a way to observe or understand how real users will impact applications and infrastructure before they push code into production.
- **Wasting time troubleshooting:** Application, operations,

infrastructure, development, and digital experience teams are pulled in to troubleshoot and try to identify the root cause of problems, wasting valuable time guessing and trying to make sense of telemetry and come up with answers.

Monitoring

At the most basic level, monitoring allows you to **gain visibility** into a system, which is a core requirement for judging service health and diagnosing your service when things go wrong.



Your monitoring system should address two questions: **what's broken, and why?**

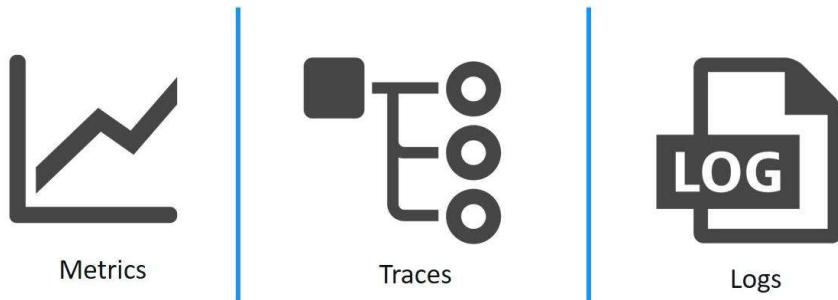
The “what’s broken” indicates the symptom; the “why” indicates a (possibly intermediate) cause. In other words, monitoring is finding the symptoms and causes.

The objectives of monitoring:

- **Alert** on conditions that require attention.
- **Investigate** and **diagnose** those issues.
- **Display information** about the system visually.
- **Gain insight into trends** in resource usage or service health for long-term planning.
- **Compare** the behavior of the system **before** and **after** a change, or between two groups in an experiment.

11

Three pillars of observability



To support a **data driven** resolution we need:

- **Logs:** A record of what's happening within your software.
- **Metrics:** A numerical assessment of application performance and resource utilization.
- **Traces:** How operations move throughout a system, from one node to another.

12

Logs are granular, timestamped, complete and immutable records of application events. Among other things, logs can be used to create a high-fidelity, millisecond-by-millisecond record of every event, complete with surrounding context, that developers can 'play back' for troubleshooting and debugging purposes.

Metrics(sometimes called time series metrics) are fundamental measures of application and system health over a given period of time, such as how much memory or CPU capacity an application uses over a five-minute span, or how much latency an application experiences during a spike in usage.

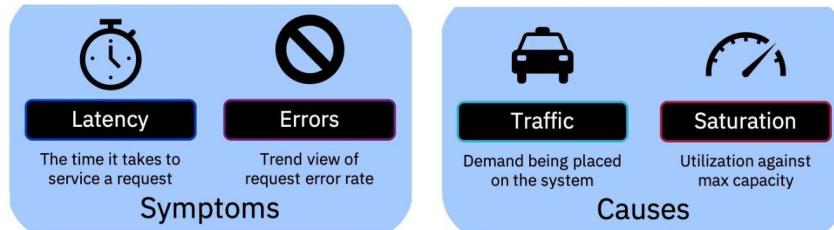
Traces record the end-to-end 'journey' of every user request, from the UI or mobile app through the entire distributed architecture and back to the user.

Four golden signals of SRE

SRE's golden signals define what it means for the system to be "healthy." The four golden signals are the basic, essential **building blocks for any effective monitoring strategy**.

- **Latency:** the time taken to serve a request
- **Traffic:** the stress from demand on the system. The more traffic, the more "stress"
- **Errors or error rate:** the rate of requests that are failing
- **Saturation:** overall capacity of the service. The saturation is a high-level overview of the utilization of the system

Establish benchmarks for each metric showing when the system is healthy – ensuring positive customer experiences and uptime.



13

Latency

Latency refers to the time taken to serve a request.

Define a benchmark for "good" latency rates. Then, monitor the latency of successful requests against failed requests to keep track of health. Tracking latency across the entire system can help identify which services are not performing well and allows teams to detect incidents faster. The latency of errors can help improve the speed at which teams identify an incident – meaning they can dive into incident response faster.

Traffic

Traffic refers to the stress from demand on the system. The more traffic, the more "stress".

How much stress is the system taking at a given time from users or transactions running through the service? Depending on the business, what you define as traffic could be vastly different from another organization. Is traffic measured as the number of people coming to the site or as the number of requests happening at a given time? By monitoring real-user interactions and traffic in the application or service, SRE teams can see exactly how customers experience the product while also seeing how the system holds up to changes in demand.

Errors

Errors, or the error rate, defines the rate of requests that are failing. SRE teams need to monitor the rate of errors happening across the entire system but also at the individual service level. Whether those errors are based on manually defined logic or they're explicit errors such as failed HTTP requests, SRE teams need to monitor them. It's also important to define which errors are critical and which ones are less dangerous. This can help teams identify the true health of a service in the eyes of a customer and take rapid action to fix frequent errors.

Saturation

Saturation is defined as the overall capacity of the service. The saturation is a high-level overview of the utilization of the system:

- How much more capacity does the service have?
- When is the service maxed out?
- Because most systems begin to degrade before utilization hits 100%, SRE teams also need to determine a benchmark for a "healthy" percentage of utilization.
- What level of saturation ensures service performance and availability for customers?

“Good observability is actionable.”

The four golden signals serve as an excellent jumping-off point for **actionable** monitoring.

That is, the engineer alerted was able to immediately take some action.

Observability requires that actionable data from multiple sources is appropriately connected, optimized and enhanced for **context**.

- **Understand the context and the topology:** instrument in a way that creates an understanding of relationships and dependencies (**dependency maps**) between services in distributed environments of potentially millions of interconnected components.
- Create a single pane of glass view into the health of all services

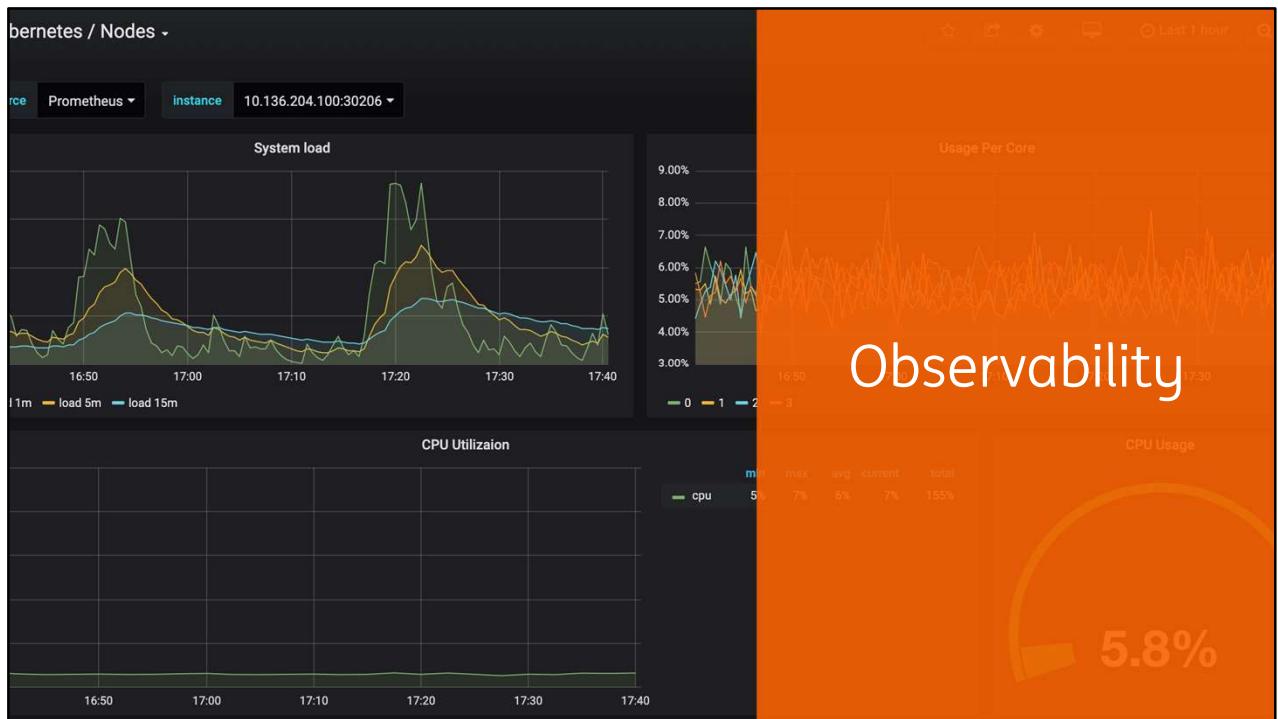
Building “observable” systems requires being able to understand the failure domain of the critical components of the system proactively, and that’s a tall order.



14

The responsibility of monitoring and observability solutions is to extract and apply the context needed to transform metrics into actionable insights.

Rich context metadata enables real-time topology maps, providing an understanding of causal dependencies both vertically throughout the stack and horizontally across services, processes, and hosts.



Types of monitoring

Infrastructure monitoring

Infrastructure monitoring involves real-time tracking of the computer systems, servers, processes and equipment that make up the computing network in an enterprise (middleware and hardware units, virtual machines, data centers, networks, disk storage etc.)

Network monitoring

Within the concept of IT infrastructure monitoring is network monitoring, which refers to tracking the health and performance of a network and its related components. Network monitoring tools help to identify network performance issues by evaluating servers, switches, routers, firewalls and virtual machines.

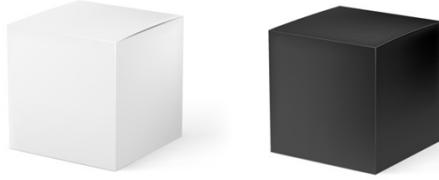
Application performance monitoring (APM)

As its name suggests, this type of continuous monitoring involves measuring the performance and availability of specific applications. Application performance monitoring tools track metrics such as: API responses and transactions, System responses, Real-user monitoring (RUM) and more.

2

Observability is a natural evolution of APM and NPM data collection methods that better addresses the increasingly rapid, distributed and dynamic nature of cloud-native application deployments. Observability doesn't replace monitoring — it enables better monitoring, and better APM and NPM.

Blackbox and whitebox monitoring



Black box monitoring

Testing externally visible behavior **as a user would see it**. It includes error responses and latency from the users' perspective. It is more focused on the symptoms.

Blackbox monitoring — that is, monitoring a system from the outside by treating it as a blackbox — is very good at the **what is broken and alerting** about a problem that's already occurring (and ideally end user-impacting)

White box monitoring

Monitoring based on metrics exposed by the **internals of the system**, including logs, interfaces like the HTTP handler that emits internal statistics. Using white-box engineers can find the **causes** and **symptoms** of the issues.

Whitebox monitoring can be used to catch signals we can **anticipate** in advance and be on the lookout for. White-box monitoring therefore allows detection of **imminent** problems.

Black box monitoring: user-experience

Extending the three-pillars approach, IT teams must augment telemetry collection with user-experience data to eliminate blind spots (or **unknown unknowns**):

Black box monitoring extends traditional observability telemetry by adding the outside-in user perspective of a specific digital experience on an application, even in pre-production environments.



4

Full stack observability

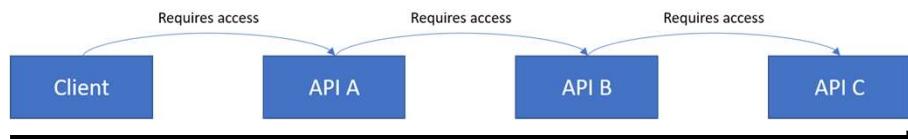
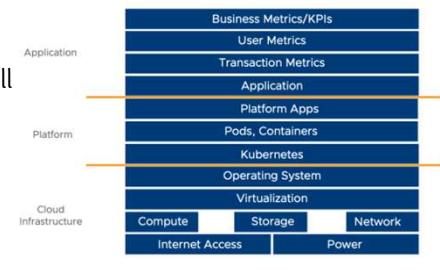
Traditional monitoring solutions offer limited, siloed visibility across the distributed applications. For example, there can be limited visibility for application services, networks, infrastructure, clouds, databases, and logs.

A full-stack observability solution gives IT teams insight into application, infrastructure, and UX performance. It provides cross-domain correlations and **dependency mapping**.

This crucial **context** also enables teams to understand how all entities are **connected**.

Dependencies (also called **dependency maps**) reveal how each application component is dependent on other components, applications and IT resources

It shows the relationships and dependencies for all entities, both **vertically up and down the stack and horizontally between services, processes, and hosts**.



5

Full stack observability enables teams to correlate application performance to the entire application technology stack—connecting performance back to their business metrics.

Full stack observability monitors the inputs (application and infrastructure stacks) and outputs (business transactions, user experiences, application performance) and provides cross-domain correlations and dependency mapping.

It includes cross-domain correlation and dependency to inform teams exactly which areas are causing performance issues and the reasons.

Measuring a distributed system means having observability in many places and being able to view them all together. This might mean both a frontend and its database, or it might mean a mobile application running on a customer's device, a cloud load balancer, and a set of microservices. Being able to connect data from all of these sources in one place is a fundamental requirement in modern observability tools.

Some more observability benefits...

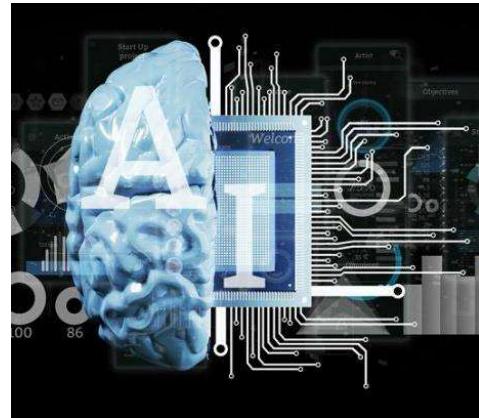
Discover and address 'unknown unknowns' - issues you don't know exist.

A chief limitation of monitoring tools is that they only watch for '**known unknowns**' - exceptional conditions you already know to watch for.

Observability discovers conditions you might never know or think to look for, then tracks their relationship to specific performance issues and provides the context for identifying root causes to speed resolution.

Enable automated remediation and self-healing application infrastructure.

Combine observability with AIOps machine learning and automation capabilities to **predict** issues based on system outputs and resolve them **without manual intervention**.



Instrument your code to maximize observability

A well-designed system aims to have the right amount of observability that starts in its **development phase**.

Don't wait until an application is in production before you start to observe it. Instrument your code and consider the following guidance:



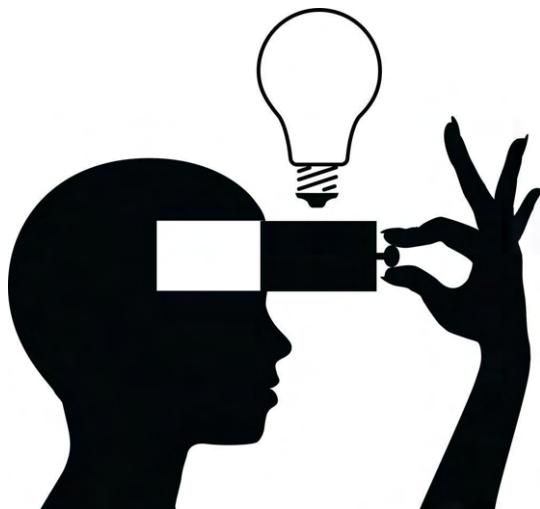
- To debug and troubleshoot efficiently, think about what log and trace entries to write out, and what metrics to monitor and export. Prioritize by the most likely or frequent failure modes of the system.
- Periodically audit and prune your monitoring. Delete unused or useless dashboards, graphs, alerts, tracing, and logging to eliminate clutter

Engineering intuition and instincts

Observations can lead a developer to the answers, it can't make them necessarily find it.

The process of examining the evidence (observations) at hand and being able to deduce still requires a good understanding of the system, the domain as well as a good sense of intuition.

No amount of "observability" or "monitoring" tooling can ever be a substitute to **good engineering intuition and instincts**.



Full stack observability



9

**Working together, solving complex
problems, that's what IT is all about!**

Blame game





MDPL



Log4all

RTK2 - Reliability Toolkit 2



Observability at ING

Monitoring at ING: SRE guidelines

Monitoring is mandatory in ING as per risk control: *Operational resilience B2 - Ability to monitor*. What does Risk require you to do?

You simply need to show you have thought about and have implemented a monitoring solution that is operating effectively. That does not help us at all, so SRE took it upon them to write out a more complete minimum standard.

1. Access application logs
2. Insight in system metrics
3. Configurable dashboarding
4. Filterable alerting to engineers



Access application logs and Insight in system metrics

1. Access application logs

Every application should on some level log what it is doing and what is going wrong. Traditionally this is done in log files on servers.

[How to achieve this](#)

Sending your logs to an **Elastic Stack** is a very good way of doing this.



Logs

2. Insight in system metrics

Metrics are simply the measurements of a specific variable over time.

The Four Golden Signals are:

- Latency
- Traffic
- Errors
- Saturation

[How to achieve this](#)

Prometheus can scrape these metrics. If you have the appropriate collectors.



Metrics

Configurable dashboarding

3. Configurable dashboarding

Next to data collection you need to be able to access and organize it. A good monitoring solution has some options to put data in graphs and the capability to do so is imperative.

How to achieve this

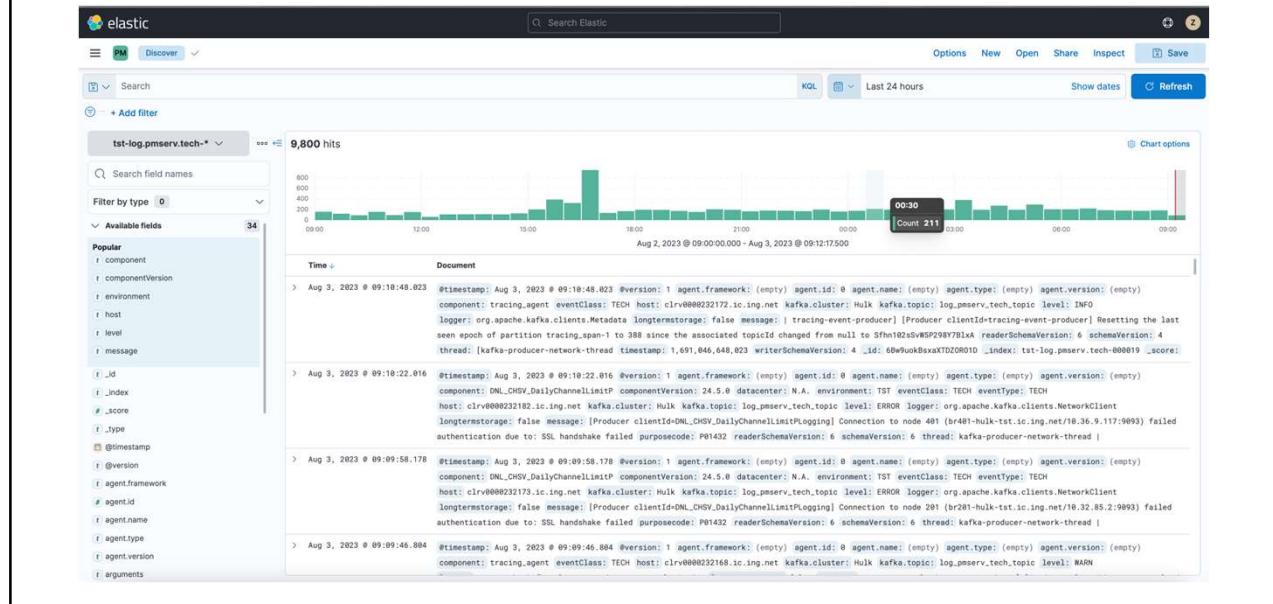
All_Elastic stacks come with dashboarding capabilities, as **Kibana** is part of the ELK stack. For metrics the most obvious choice is **Grafana** - which is part of the Reliability Toolkit (RTK2) solution.



4

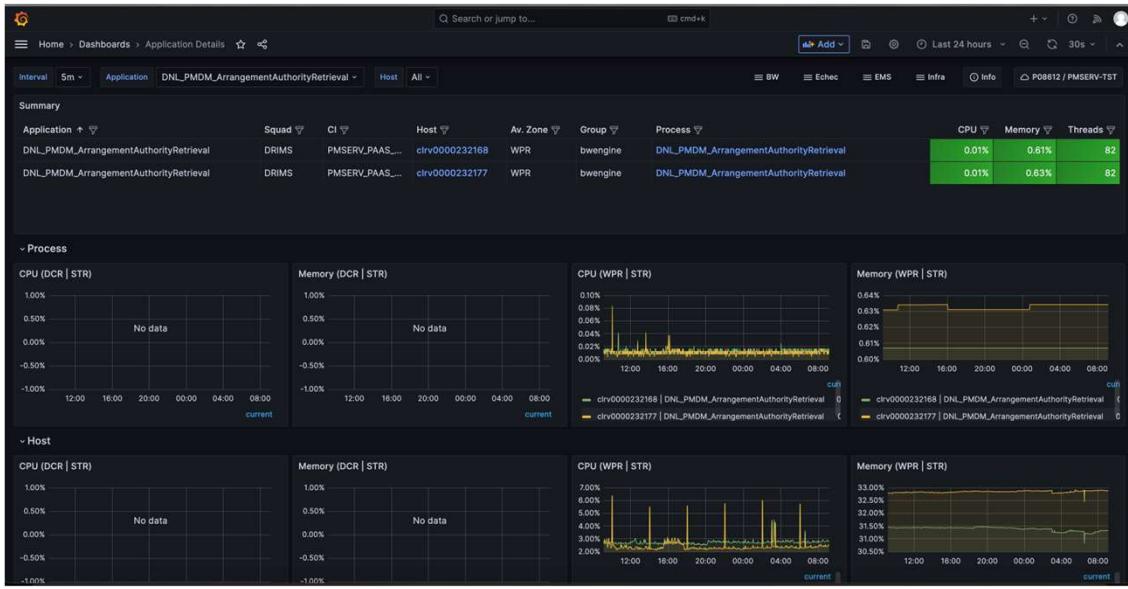
Not setting up alerting correctly risks either not sending critical alerts or in the worst cases engineers might start ignoring alerts because they are getting so many false alerts they stop taking it seriously. The latter is called **alarm fatigue** and is quite common in ING.

Example Kibana



Not setting up alerting correctly risks either not sending critical alerts or in the worst cases engineers might start ignoring alerts because they are getting so many false alerts they stop taking it seriously. The latter is called **alarm fatigue** and is quite common in ING.

Example Grafana



6

Not setting up alerting correctly risks either not sending critical alerts or in the worst cases engineers might start ignoring alerts because they are getting so many false alerts they stop taking it seriously. The latter is called **alarm fatigue** and is quite common in ING.

Filterable alerting to engineers

4. Filterable alerting to engineers

Your system needs to be able tell if something is wrong with it and able to reach the correct person. Your alerting system must be able to filter less important issues or route them to less intrusive channels.

How to achieve this

A combination of **Prometheus** and **IAT** is a very good way of achieving this. Prometheus can detect issues in your system and filter if these are worth alerting about. IAT supports a large amount of means to contact engineers (mattermost, incident in snow, email, sms, robo-call).



7

Not setting up alerting correctly risks either not sending critical alerts or in the worst cases engineers might start ignoring alerts because they are getting so many false alerts they stop taking it seriously. The latter is called **alarm fatigue** and is quite common in ING.

Confidentiality higher than 2

Never store data with a Confidentiality
higher than 2 in operational logs
(NO customer, payments data or
passwords...)

MDPL (Monitoring data pipeline)

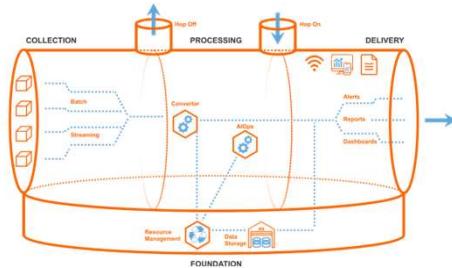
MDPL (Monitoring data pipeline) is a set of tools, applications and processes to help users get control over their application data. MDPL delivers IT Monitoring as a Service (IT-MaaS) at global level in ING.

The concept of the Monitoring Data Pipeline (MDPL):

- comprised of three functional areas [**Collection**, **Processing** and **Delivery**]
- supporting data ingestion of four different streams of Monitoring data [**Logs, Metrics, Events and Traces**]
- coming from **different source domains** [Non-Cloud, ING Private Cloud and different Public Clouds]
- able to **process** (part of) the collected data
- and **deliver the resulting information** to its consumers

All the above is to support the goal of decreasing **MTTD (Mean Time to Detect)** and **MTTR (Mean Time to Repair)**.

Ultimately, by using Machine Learning and Big Data Analysis, we aim to satisfy our ultimate goal of becoming predictive rather than reactive.



9

ING's IT landscape is changing very fast. To support this ongoing change we constantly provide new monitoring services and features that help you to have insights in reliability and availability of your IT product. The shift from monolithic infrastructures and applications, through distributed, to microservices is the main driver behind the changes in the way we operate our IT environment within ING.

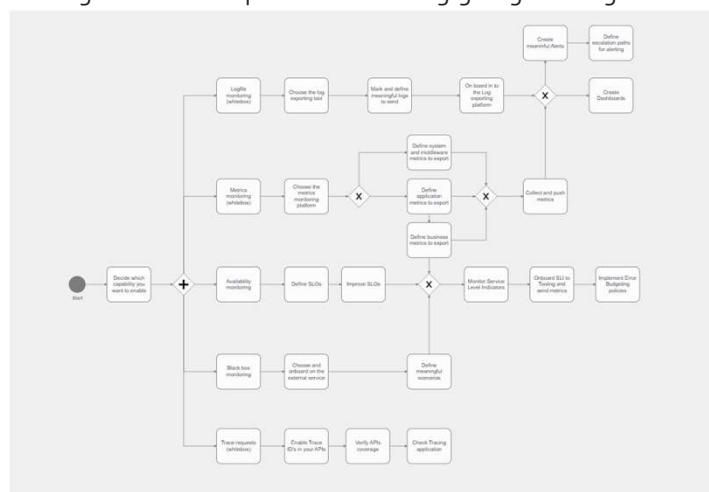
Similarly, monitoring is shifting from generating structured data from a few known sources, to unstructured data coming from many – not all ING-hosted or owned – sources. In short: monitoring is becoming Big Data

<https://globalmonitoring.ing.net/getting-started>

Monitoring Journeys

Scope for this journey is to help you understand the different types of monitoring available in ING and which ones are best suited for your needs.

You can begin by clicking on the Start point below or by going directly to the topic you are interested in



10

<https://theforge.ing.net/journey/Monitoring1/details>

<https://theforge-test.ing.net/searchview/monitoring/journeys>

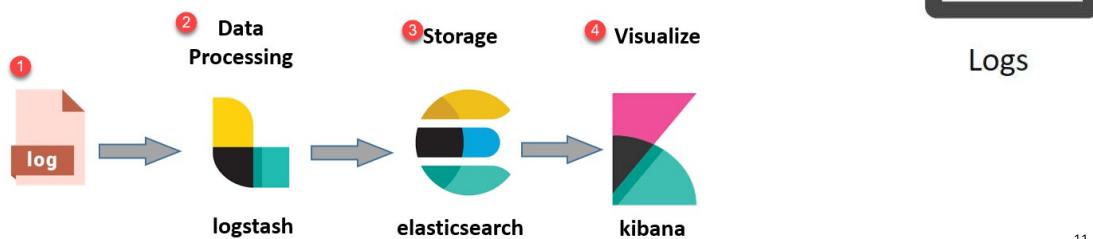
Metrics:

<https://theforge.ing.net/journey/CwOnOtZKTiMT5VK9/details>

ELK

ELK is a well-known standard to parse arbitrary log files and send them to an analytics engine (**ElasticSearch**) for processing and visualization (**Kibana**).

Goal of ELK is to facilitate investigation of your application for known patterns as well as detect anomalies.



11

<https://confluence.ing.net/pages/viewpage.action?pageId=857200051>

LogFile monitoring

If you want to send logs to ELK you can choose between different flavors:

- **ELKaaS**: self serving offering to build your own elasticsearch cluster. This solution is suggested if you have a good knowledge and understanding of ELK and are able to automate in your pipeline all of your setup. Due to the initial cost it makes sense to opt for this solution only in some specific cases
- **Log4all**: ELK offering based on ELKaaS, supports natively logs from linux servers and therefore could allow you to perform queries that include system and application information
- **Elastic**: ELK offering based on ELKaaS, it provides a business ELK that can provide you alerting and other interesting offering such as APM. Provided by the global monitoring squad from Belgium does not yet integrate with system logs.
- .



Logs

12

<https://confluence.ing.net/pages/viewpage.action?pageId=857200051>

Logs - Log4All

Need to have insight in your Log data? Want to look for anomalies, or trends over the last weeks? Fancy your own dashboard to visualize your data? Want to query your collected Log data in an easy way without accessing your server? Want to analyze huge data without compromising your application performance? Want to easily browse your Linux syslog?

If you're looking for a solution for this then **Log4All** might be something for you. Log4All stores the content of your logfiles in a centralized logging environment and makes these easy searchable on key fields of your choice. Any update of your log will be sent to Log4All where you can access these directly, or, even better, display them on a dashboard.

The main components used to implement Log4All are Elastic Stack (**Elastic**, **Logstash**, **Kibana**) , **Kafka** and **ACE**.



Logs

13

<https://confluence.ing.net/pages/viewpage.action?pageId=857200051>

Metrics

How utilized is my IT infrastructure and what is the trend of the utilization? How many users are connected to my application over time and is the number of concurrent connections increasing or decreasing? How many transactions per second are handled by my application and how long does it take to complete a transaction?

These types of questions can be answered by collecting metrics and visualizing them.



Metrics

RTK2

RTK2 is based upon open-source tools Prometheus and Grafana. Prometheus brings together metrics for all your applications and infrastructure in one place and with Grafana you are able to visualize the health of these applications and infrastructure components

Long Term Metric Store (LTMS)

Where RTK2 is able to store all metrics for at most 32 days, the LTMS will keep metrics much longer. That enables you to detect trends in your IT resource utilization as early as possible.

14

<https://theforge.ing.net/product/45412/documentation/latest/onboarding.html#onboarding-index>

<https://theforge.ing.net/product/240493/documentation/latest/monitoring/ing-tooling.html>

SingleView will ultimately allow you to tie several MDPL products into one single view. The first implementation is aimed at Metrics and will give you the option to create one dashboard showing Metrics of multiple applications you own or have interest in.

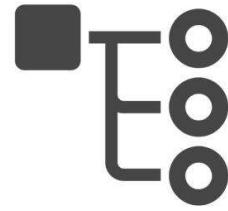
<https://theforge.ing.net/product/239673/documentation/latest/index.html>

Traces - TracING

Wondering what goes on behind the scenes once your API has been called? Having trouble pinpointing where in the chain things go wrong? You need trace information.

TracING is the way to go. Not only does TracING show you what your chain looks like, it also reports whether success was achieved or whether errors have occurred (and more importantly, where they've occurred).

Use the interactive GUI to find out what happens when and where and reduce your time-to-market. TracING is an implementation of the **OpenTracing.io** standard for track and trace.



Traces

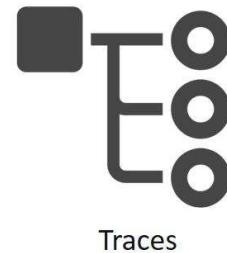
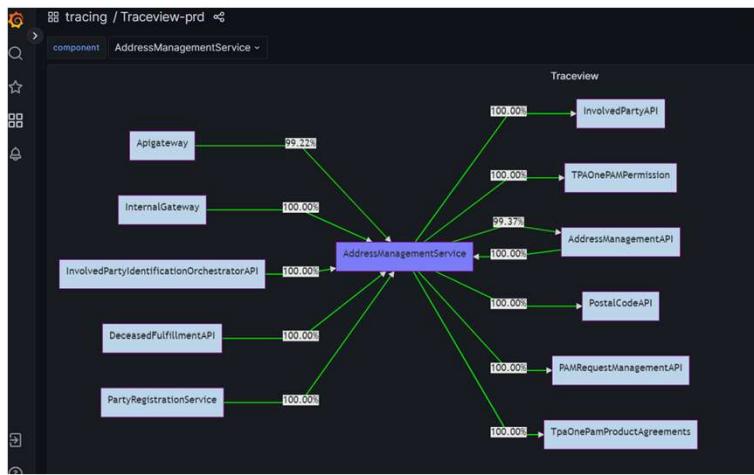
15

<https://theforge.ing.net/product/17234/documentation/latest/index>

TraceView

TraceView is an extention of TracING. With TracING you can see traces in great detail but sometimes you want to have an higher overview. TraceView provides exactly that.

You can see relations between components on a higher level.



Traces

16

<https://theforge.ing.net/product/45118/documentation/latest/index>

<https://theforge.ing.net/product/238113/details>

OneDashboard - Chain monitoring

OneDashboard is a chain monitoring tool:

Chain monitoring application **definition**:

- A real time generated visualization of a user defined chain based on dynamic chain data
- User defines a chain by providing a query

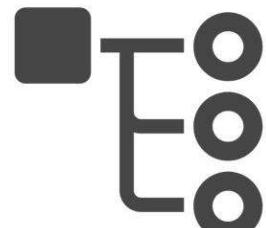
Its goals include supporting in:

- Monitoring
- Chain visualization
- Integration between API's by shortening the feedback loop

Examples are:

- iDeal flow
- Global/NL/BE Login flow
- NL/BE Landing flow

Limitations: Middleware and databases not creating trace events



Traces

17

<https://inglearn.udemy.com/course/ngt-berlin/learn/lecture/38353950#overview>

OneDashboard: Visualize in real time all interactions in a chain

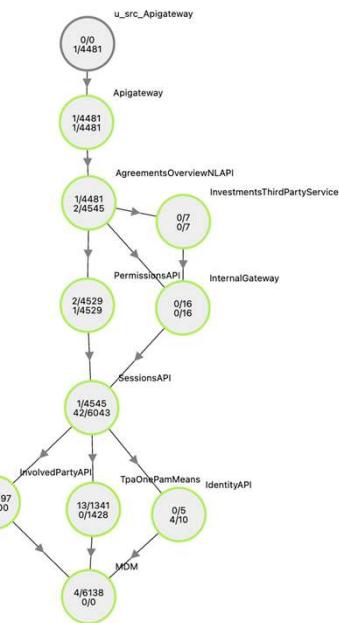
You can define your chain with a query language

- Query should represent the essence of the chain
- For the “NL Landing page interactions with MDM” chain:

```
AgreementsOverviewNLAPI and operation:GET_agreements  
FILTER MDM[*,*]
```

OneDashboard is unique because it can visualize chain data

- Show all and only the components that make up the chain:
 - InternalGateway is connected to many applications, but for this chain you can see it's only used to connect to the SessionsAPI
- Show only requests relevant to the chain:
 - Here you only see requests to MDM for customer data that are part of the Dutch AgreementOverview page



<https://theforge.ing.net/product/45118/documentation/latest/index>

<https://theforge.ing.net/product/238113/details>

OneDashboard: Consumers and dependencies

The whole chain is shown by default

- Can be an issue for very large chains

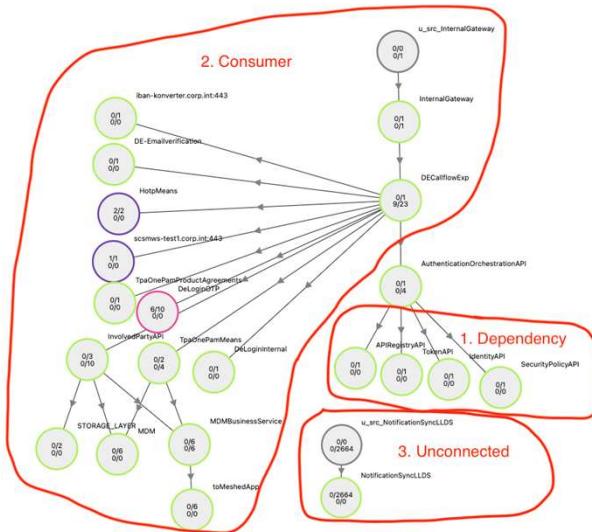
Many consumers use *AuthenticationOrchestrationAPI* to login. Each of these consumers can add a big subchain which might not be interesting (Nr 2).

Filter query allow you to specify anchors. Anchors have subchains:

- **Dependencies (Nr 1)**
- **Consumers (Nr 2)**
- Unconnected.(Nr 3)

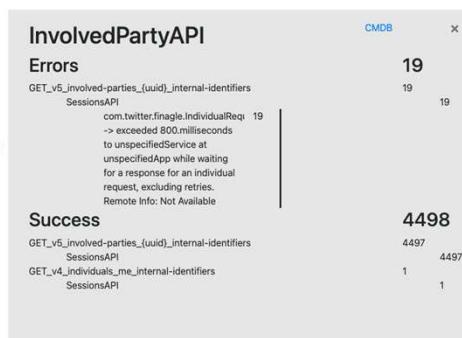
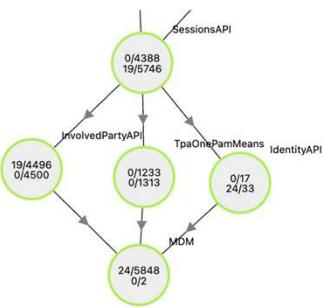
With a filter you can specify how these subchains are visualized

- Dependencies are shown up to a certain depth
- Consumers are shown, but not the consumer specific Dependencies
- Unconnected chains are not shown at all



19

OneDashboard: Detailed view



In this case you only see client timeouts from SessionsAPI and no error messages from the server side (InvolvedPartyAPI). An action could be to increase SessionsAPI client timeouts settings to prevent errors.

Click on a component for a sliding window with details.

Details are split by:

- Error/Success requests
- Endpoints
- Consumer

Error messages are shown also:

- Left: Client errors
- Right: Server errors

20

<https://theforge.ing.net/product/45118/documentation/latest/index>

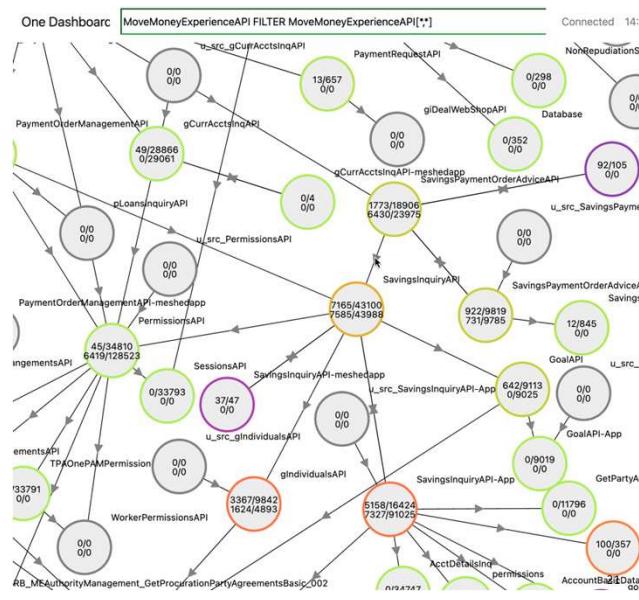
<https://theforge.ing.net/product/238113/details>

OneDashboard: Link to CMDB

Detail window also contains a link to CMDB which allows you to quickly find the team supporting a failing component in CMDB.

Example:
Major incident with SavingsInquiryAPI for
the MoveMoney NL chain

1. Find the source component of the errors
 2. Click on the source component to get detailed view: SavingsInquiryAPI
 3. Check the error messages: server error with 500 status code
 4. Find the support team by clicking on the CMDB link



<https://theforge.ing.net/product/45118/documentation/latest/index>

<https://theforge.ing.net/product/238113/details>

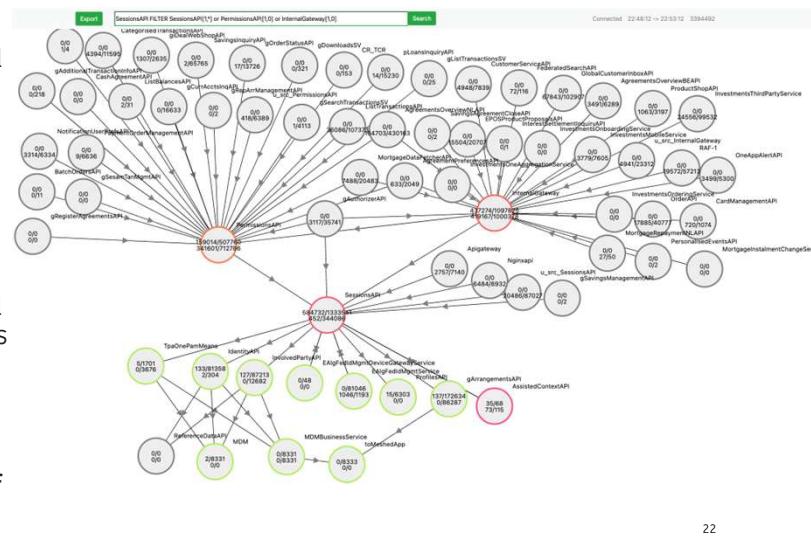
OneDashboard: Changes can be monitored in real time

OneDashboard shows current health in real time with second granularity

- Changes can be monitored in real time.
- Mitigating actions during incidents
- Monitor regular changes/deployments

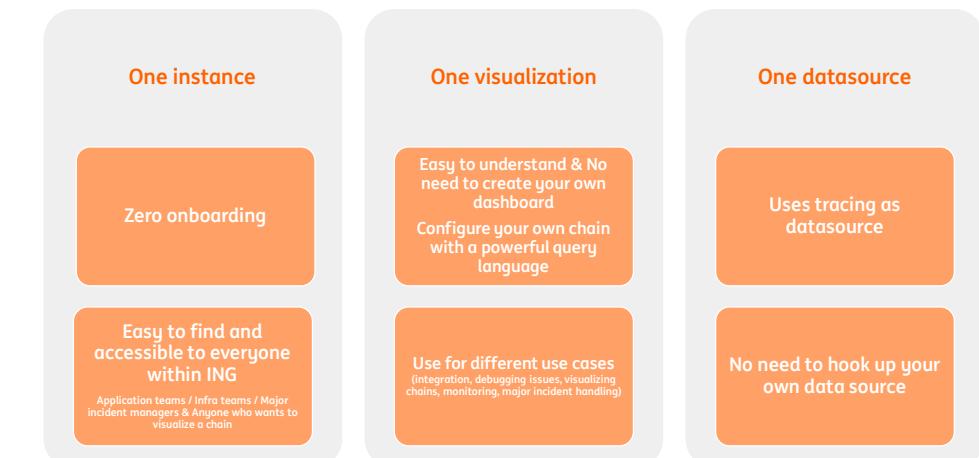
Example:

- For the Feb 1 Major we performed a lot of mitigating actions and it was essential to get quick feedback of for the actions
- **To be able to perform other mitigating actions more quickly**
- **To quickly roll back the actions if they had negative impact**



22

Why OneDashboard?



23

SingleView

SingleView will ultimately allow you to tie several MDPL products into one single view.

The first implementation is aimed at Metrics and will give you the option to create one dashboard showing Metrics of multiple applications you own or have interest in.

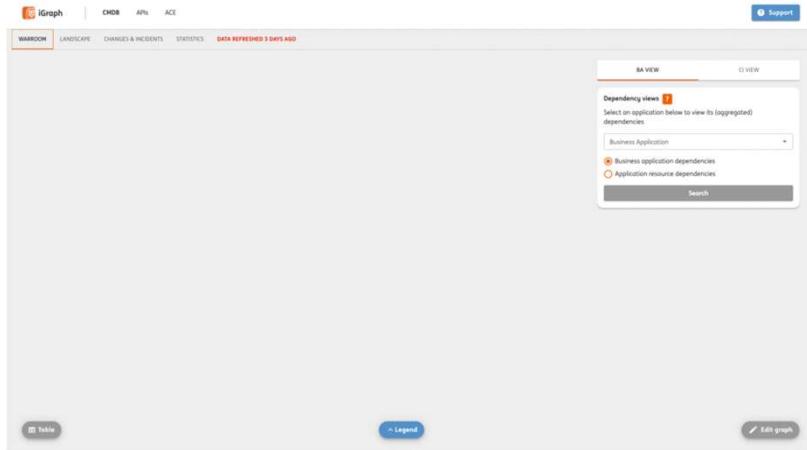


24

<https://theforge.ing.net/product/239673/details/>

Relationship cmdb tools: iGraph

iGraph tries to combine and visualize several important IT data sources using state-of-the-art graph database technologies. This enables you to quickly visualize application dependencies, find responsible teams/persons, observe an entire application landscape or see what application is subscribed to different APIs.



Visualizing this information is just the beginning. In the future, we aim to use analytics to provide striking new insights like predicting potential incidents in applications.

25

<https://confluence.ing.net/display/IGR/User+manual>



MDPL



Log4all

RTK2 - Reliability Toolkit 2



Observability at ING

Alerting - ING Alert Transporter (IAT)

ING Alert Transporter (IAT) automates the alerting process and speeds up alert notifications, thus making it painless for engineers to set up a reliable alerting solution next to their monitoring systems.

IAT's on-call management capabilities make it simple to distribute on-call responsibilities across team members. With flexible notifications, standby schedules, and escalations, IAT ensures that the right people are notified.

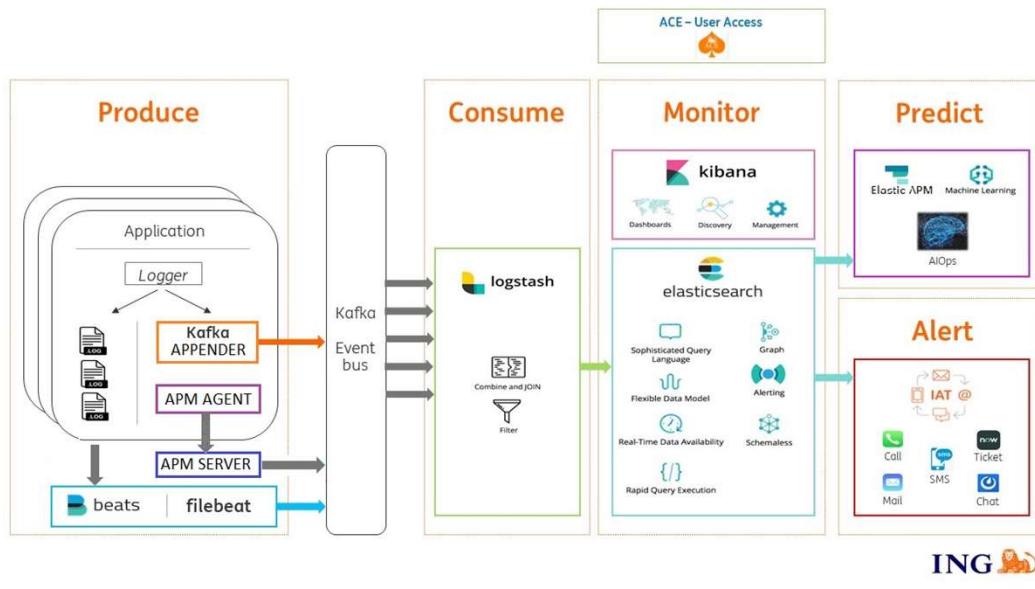
The service employs multiple outgoing channels, such as phone calls, e-mail, SMS, Mattermost, Master Control Room and Service Now.



2

<https://theforge.ing.net/product/22019/details>

Log4all



Black box monitoring

At the moment blackbox monitoring in ING is provided by two external tools: **Splunk Synthetics** and **WTSS**.

WTSS enables a synthetic monitoring of the digital channels through robots. These robots simulate multiple customer journeys on regular basis.

WTSS offers:

- Monitoring of Web and App applications.
- Provide report of journeys availability and reliability via **e-reporter**.
- Access to the WTSS raw data via Kibana



Splunk Synthetic (previously named 'Rigor') is a black box monitoring tool to test externally visible behavior of your application as a customer would see it, i.e. on business service level.

You can replicate this behavior to see what is actually happening when and how. The advantage is that you can generate load for applications that handle a small number of requests. A Sandbox environment is available for early adopters

4

<https://confluence.ing.net/display/CAP/Black+Box+Monitoring+Platform>

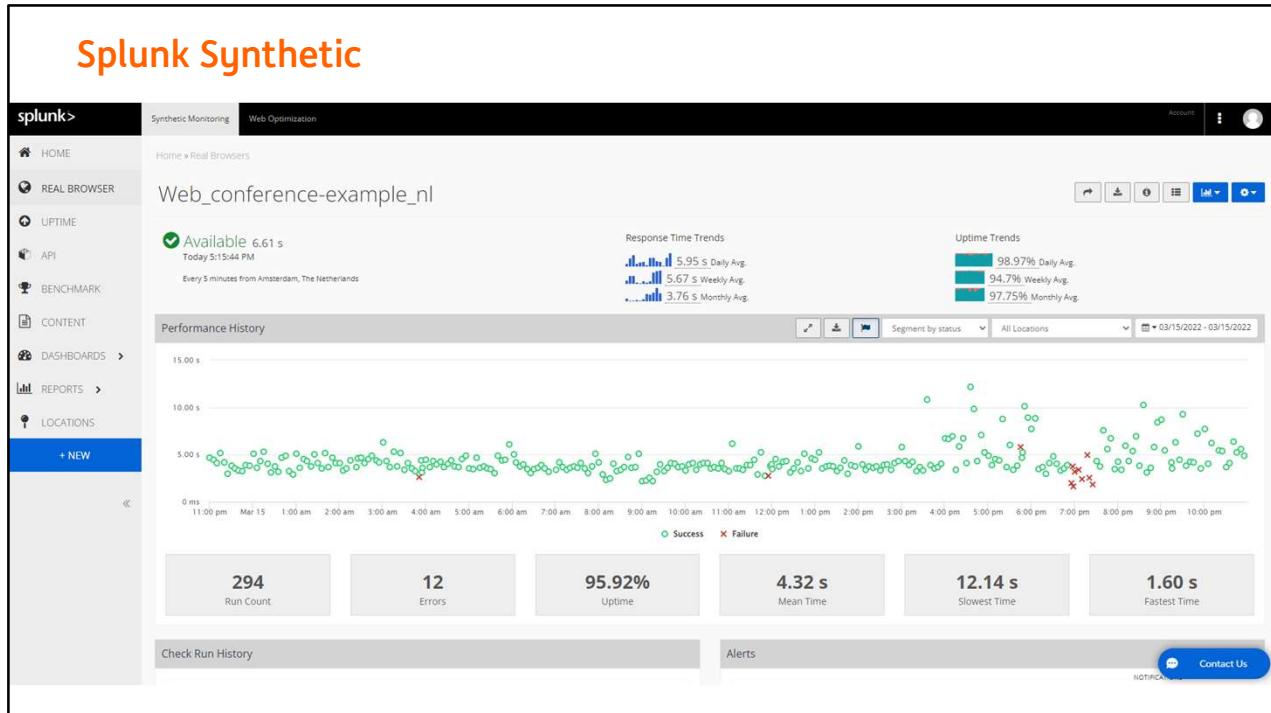
<https://theforge.ing.net/product/37125/documentation/latest/wtss.html>

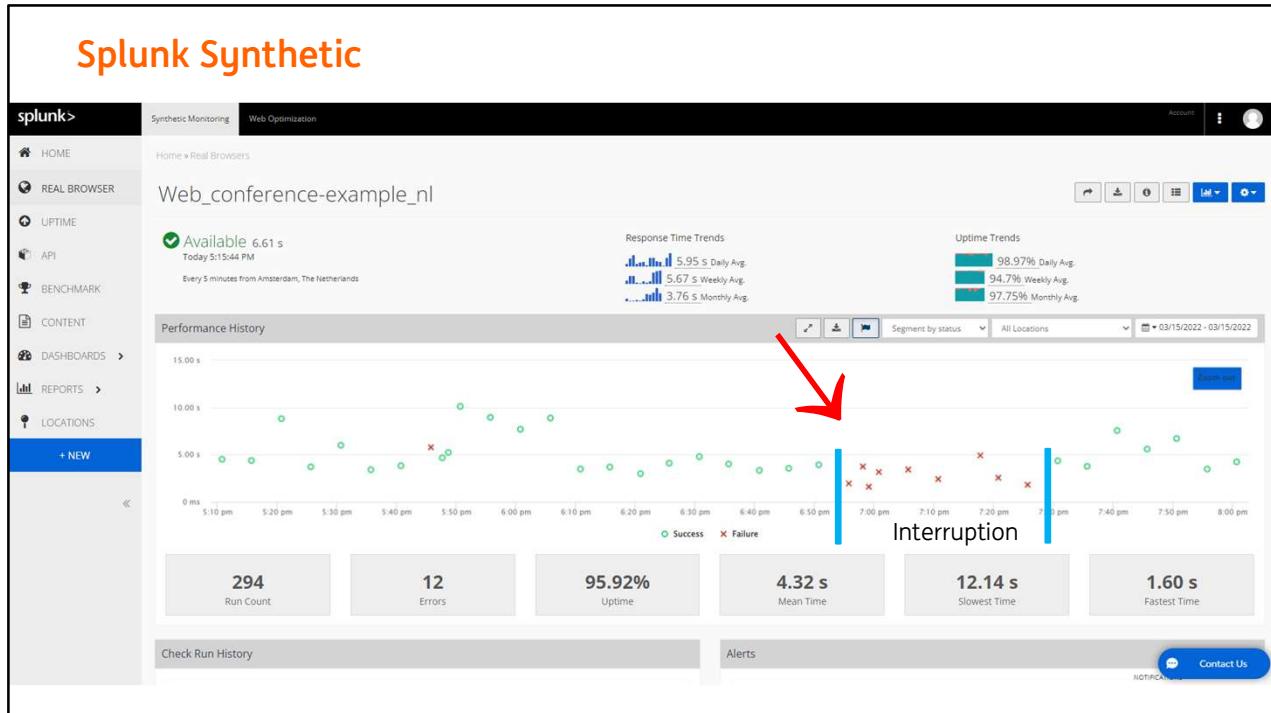
<https://theforge.ing.net/product/45010/details>

WTSS versus Splunk Synthetic

Differences **WTSS** and **Splunk Synthetic**:

Feature	Splunk	WTSS
Configuration	Teams create their own scripts	Request configuration at central team
Support for Internet Web Apps	Yes	Yes
Support for mobile apps	No	Yes
Support for intranet Web Apps	Yes	No
Support for internal APIs	Yes	No
Integrated with IAT	Yes	Yes
Integrated with eReporter	Yes	Yes





Analytics/Reporting: eReporter

The **eReporter** is an internally developed tool which reports availability of applications and business services.

Before eReporter can start collecting data a squad needs to register their asset(s) and/or business service(s) in the eReporter

Uptime:

The data for uptime availability can be retrieved from various monitoring sources like Rigor/Splunk, WTSS, Prometheus, API gateway and others.



Latency:

The data for latency availability can be retrieved from various monitoring sources like Prometheus and others.

Succesrate:

The data for succesrate availability can be retrieved from various monitoring sources like Prometheus others.

8

<https://theforge.ing.net/journey/ReliabilityReporting/details>

Analytics/Reporting



SLIM is a central location to report Service level Objective and Indicators.

While most monitoring tools in ING have a data persistence of one or two months, SLIM offers the ability to persist the data for much longer. Thus, it enables consumers to look back on their SLO history for a longer period.

Once set up in SLIM consumers are completely ready to start doing error-budgeting and become more in control of their reliability.

9

<https://theforge.ing.net/product/45150/documentation/latest/index>

<https://theforge.ing.net/journey/ReliabilityReporting/details>

https://theforge.ing.net/product/45224/documentation/latest/pages/product_description.html

Everbridge

Everbridge is the unified approach to communicate during critical events. Critical events, whether they be severe weather or a failure of IT infrastructure, can and do result in significant impacts to both business operations and to human safety. Everbridge helps ING to effectively communicate and respond to these critical events.

Everbridge service offers you access to the Everbridge SAAS solution enabling DevOps teams and departments to send out notifications in a standardized and compliant way. The available notification channels are: phone, SMS, email and Everbridge mobile app.

MVP 1 is focusing on supporting GMIM (global major incident Management) and Crisis management. These processes rely on tools like Everbridge to communicate during Majors and Crises.

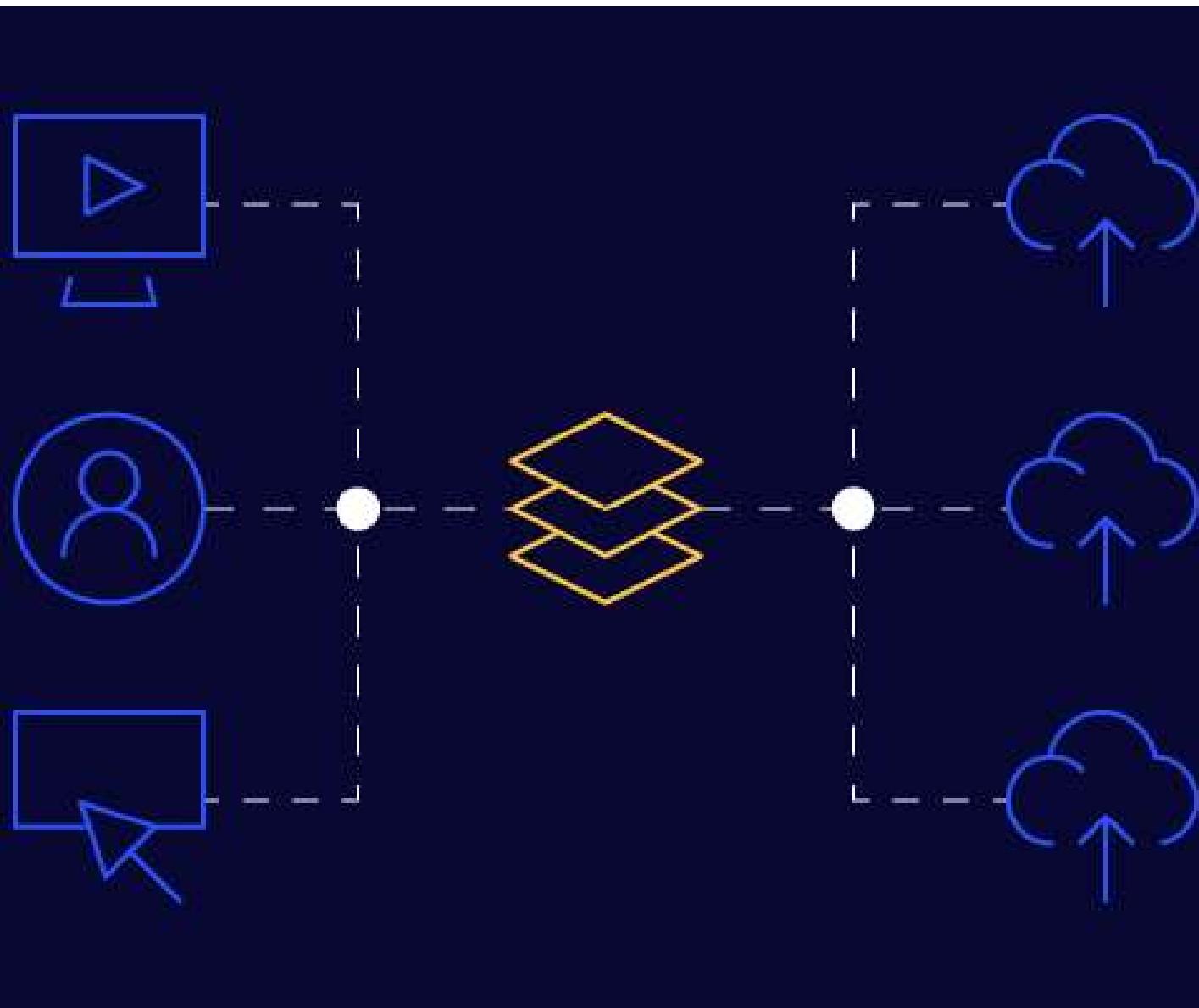


10

<https://theforge.ing.net/product/22019/details>



do your thing



API Gateways

Frontend APIs versus Backend APIs

To clarify load balancing in this training we divide the APIs in **Frontend** API's and **Backend** Apis.

In this context we mean with a **Frontend** API an API that is consumed by a call from a browser **OUTSIDE** the datacenter, usually javascript (Polymer etc.).

A **Backend** API is an API that is consumed by another API **INSIDE** the datacenter: an **API2API** call.

Internal or external API gateway

Frontend API's are always behind the **internal** or **external** API gateway/ security gateway/authenticating proxy. These are implemented with NGINX.

The gateway performs vital security checks.

For **external** facing API's (internet) it is mandatory traffic is routed through the **ING DMZ = EAGN**.

The external API gateway is located in the EAGN. Network zones will be discussed in detail in the network part of this training.

The NGINXs are High Available within each DC.

API gateways / Authenticating proxies

There are separate API Gateway instances for different purposes.

The (different) API gateways/authenticating proxies enforce authentication and can

1. Create the **TPA Access token**
2. Create the **RIAF channel context object**
3. Create the **BE RIAF context object**
4. Can translate the TPA Access token to NL RIAF channel context object

The **HTTP header** is manipulated by the NGINX (by several APIs integrated in the NGINX).

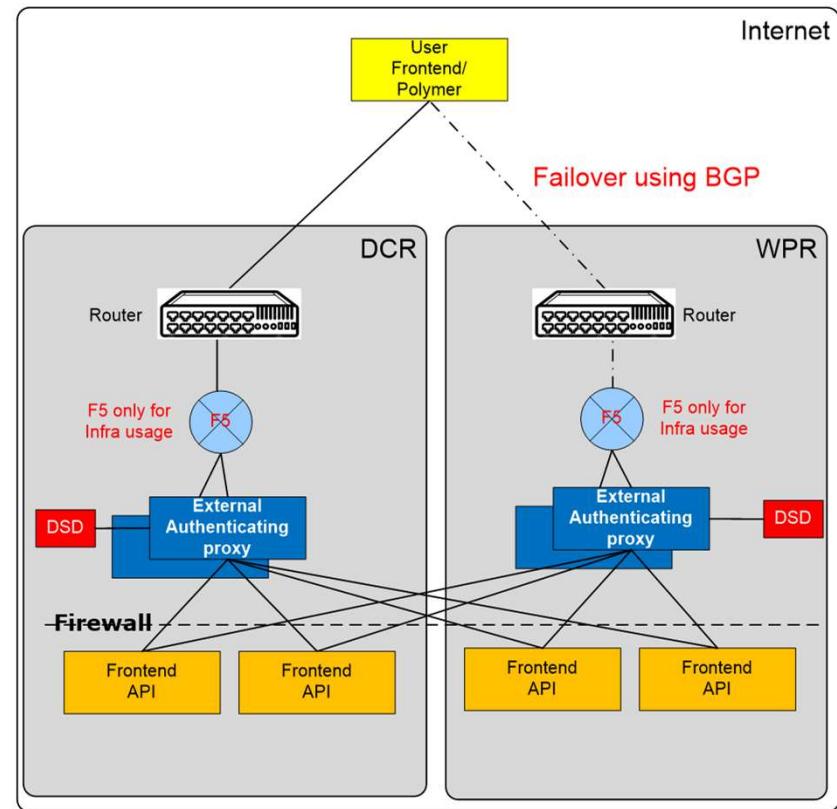
Internet Frontend APIs at ING

The Border Gateway Protocol (**BGP**) is used to manage the failover between the DC's.

BGP is a **routing protocol** used between ISP's for example to determine which path (or cable) should be used.

At the moment a **F5** is used for HA of the Authenticating proxies.

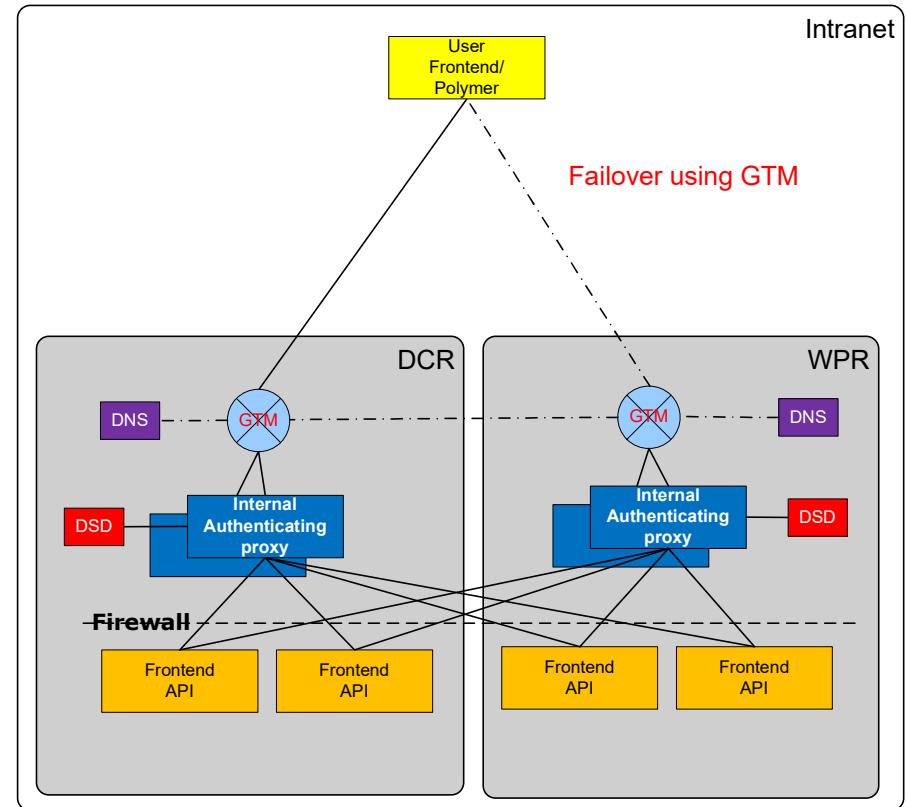
The Authenticating proxies retrieve the servernames from the **DSD** and load balances traffic over **all API instances in both DCs**



Intranet Frontend APIs at ING

The **GTM** is used for failover over the two DCs

The Internal Authenticating proxies retrieve the servernames from the **DSD** and load balance traffic over **all API instances in both DCs**



Load balancing Backend APIs

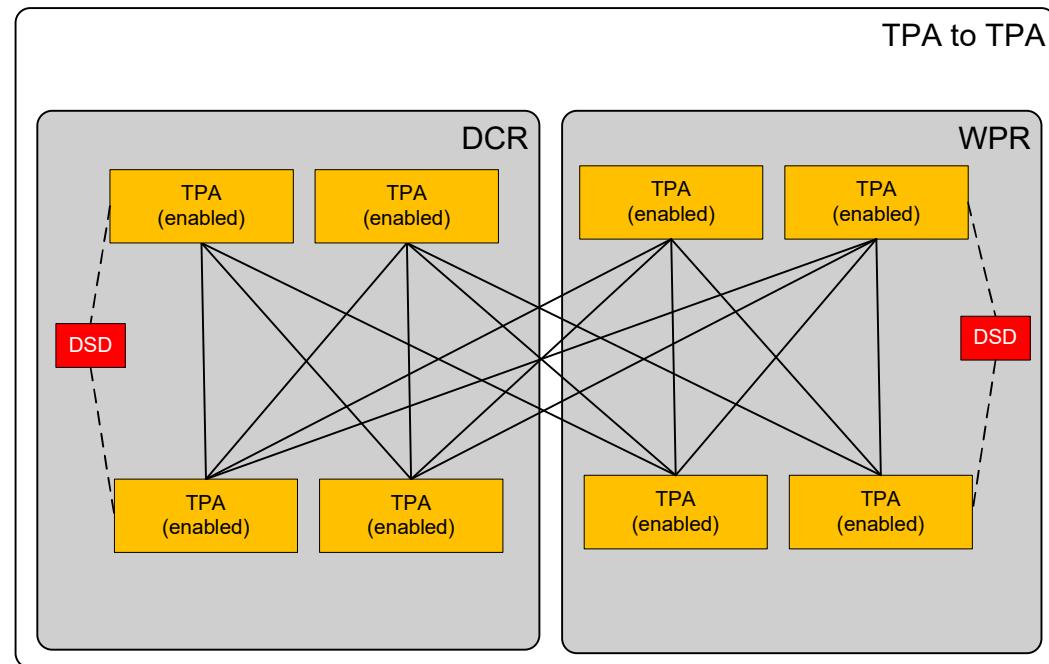
As we are in a **hybrid** situation **Backend API's, API2API** calls can use for load balancing:

1. **Clientside load balancing** (all TPA enabled APIs)
2. **(contained) F5** for existing RIAF, older APIs and COTS applications
3. **API Gateway** (for translation of TPA access token to RIAF channel context object)
4. Your own provisioned **NGINX** in the IPC

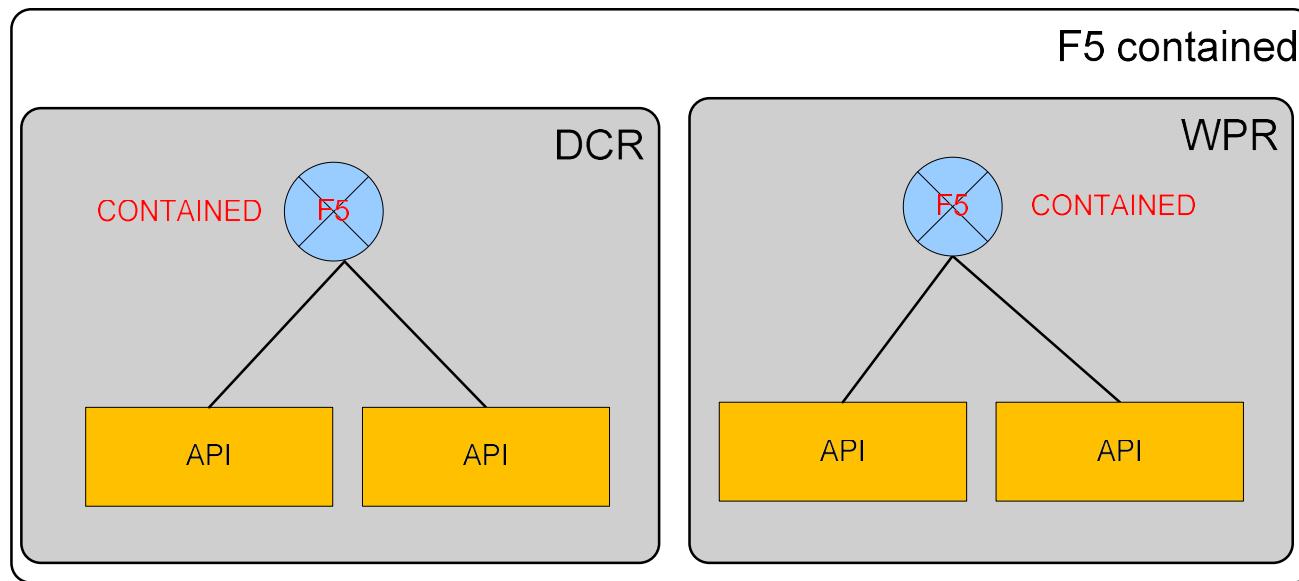
It is NOT allowed anymore to use the F5 load balancer for application load balancing.

Load balancing Backend APIs: clientside load balancing

This is the target!



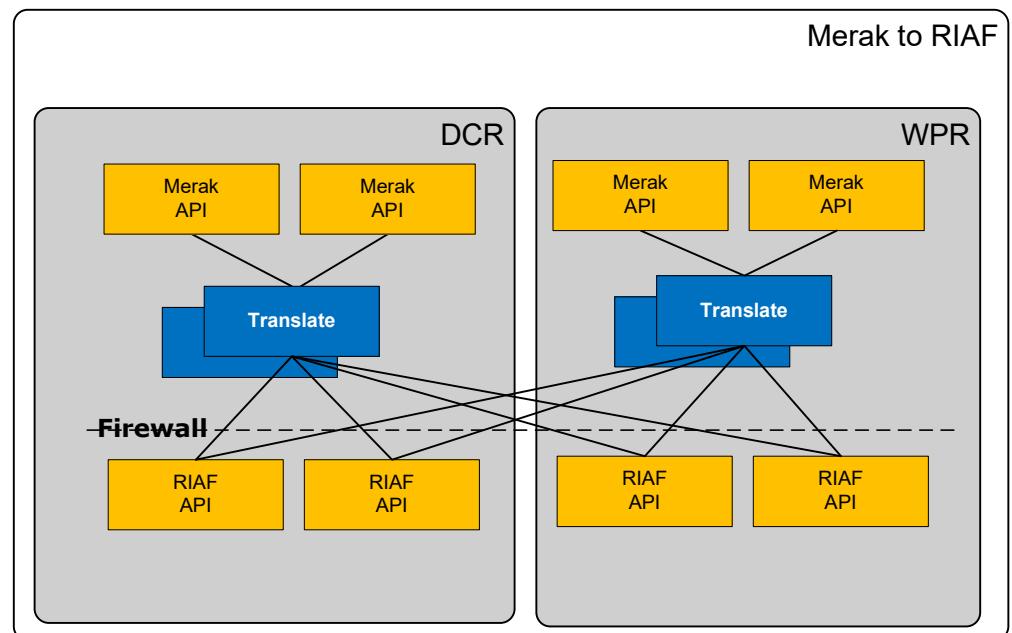
Load balancing Backend APIs: contained F5



Load balancing Backend APIs: Translate

An API Gateway is used for translation of TPA access token to RIAF channel context object.

This is a simplified view (no F5, see internet and ontranet frontend slides)



Session affinity or sticky sessions?

Stickiness or session affinity is not supported.

Some legacy applications require that all requests from the same session go to the same host. Implicitly, this is a **SPOF**: data (in memory) is lost when requests are sent to another instance...

You can solve this issue by using for example Cassandra to store your session data. Or **solve it in your application** (Ehcache etc.).

The only possibility to implement stickiness is by using and configuring your own NGINX in IPC.



Design for
Failure

Smart endpoints
Dumb pipes