
Blaise

Blaise Deployment Fallback / Cloud

October 23th, 2023

Topics Presentation (1)

- Blaise Server roles
  - Changes applied to Blaise Server roles 5.13 –> 5.14
  - Fallback Server roles
- Blaise in the Cloud
  - Technical Architecture
    - Docker
    - Kubernetes
    - Kafka
    - Monitoring : Prometheus & Grafana
    - Blaise cloud applications / containers

Topics Presentation (2)

- Blaise in the Cloud
  - Deployment
    - Deployment of Blaise in the cloud
    - Deployment of a (web)survey
    - Monitoring and checking running state
    - Scaling : horizontally and vertically
    - Upgrading / downgrading Blaise versions
    - Installing multiple versions of Blaise side-by-side
    - Security
    - Troubleshooting

Blaise Server Roles 5.13

Version 5.13	Server role description
Cati	The *CATI* server is responsible for making day batches, appropriately handling CATI requests and keeping track of their status. A server park can hold only one *CATI* server.
Data	A *Data* server stores the survey data collected with Blaise instruments. A server park can hold only one *Data* server
Data Entry	A *Data Entry* server handles pages and executes the rules for Blaise surveys. A Server Park can hold multiple publicly hosted *Data Entry* servers.
Event	The *Event* server stores all events while working with the Blaise system. A server park can hold only one *Event* server.
Resource	A *Resource* server hosts the intangible resources like corporate templates, images, brands, embedded fonts etc. for Blaise surveys. A server park can hold multiple *Resource* servers.
Session	A *Session* server manages and stores all data of active interview sessions. A server park can hold only one *Session* server.
Web	*A Web* server hosts files for Blaise internet surveys. A server park can hold multiple Web servers.
Audit Trail	An Audit Trail server is used to collect Audit Trail para data. The role is also used in this and previous versions for Publishing events and Cari functionality. A server park can hold only one *Audit Trail* server.

Blaise Server Roles - Changes

Version 5.14	Changes to Blaise server roles
Data	A *Data* server stores the survey data collected with Blaise instruments. A server park can hold multiple *Data* servers
Session	A *Session* server manages and stores all data of active interview sessions. A server park can hold multiple *Session* servers.
Audit Trail	An Audit Trail server is used to collect Audit Trail para data. The Event Publishing service and Cari service are not part of the Audit Role anymore, because for these two services a dedicated role has been introduced (see below). A server park can hold multiple *Audit Trail* servers.
Cari	A *CARI* server is used to collect CARI para data such as audio recordings and screenshots. A server park can hold only one *CARI* server
Publish	A *Publish* server hosts the Server Events Publish/Subscribe services. With these services you can subscribe to events and/or signal certain events. A server park can hold only one *Publish* server.
Case Management	A *Case Management* server is responsible for handling Case Management requests and processing of case events. A server park can hold only one *Case Management* server.

Fallback Server Roles

- Blaise Server Park
  - Single Server Roles: Data, Audit Trail, Session
  - Single Point of Failure
- Goal: Want to increase availability
- Solution: Allow to define fallback Server(s):
  - Multiple Servers can have ‘single’ server role when:
  - Data is stored in a database (not file-based) for each Survey
- Fallback mechanism:
  - When Request to (single role) Server fails, the Request is sent to a fallback Server.

Fallback for Data Server Role

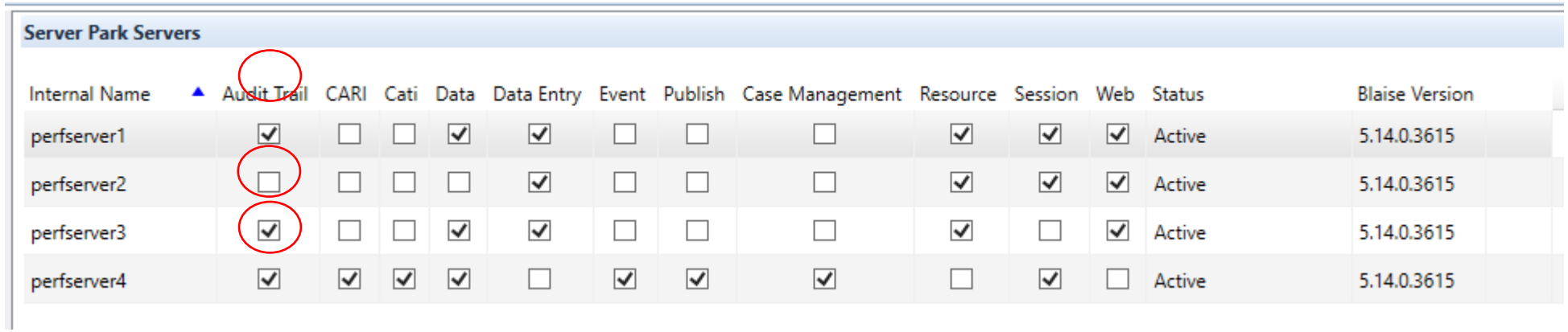
- Record Locking implementation changes:
  - Record Locks are currently stored in Memory in Data Service
  - Not suited for Park with Fallback Data Server(s)
  - Solution: Store Record Locks in a (not file-based) Database.
  - Define Record Locking Data Interface.
- Sample deployment:

Server Park Servers

Internal Name	Audit Trail	CARI	Cati	Data	Data Entry	Event	Publish	Case Management	Resource	Session	Web	Status	Blaise Version
perfserver4	☑	☑	☑	☑	☐	☑	☑	☑	☐	☑	☐	Active	5.14.0.3617
perfserver3	☐	☐	☐	☐	☑	☐	☐	☐	☑	☐	☑	Active	5.14.0.3617
perfserver2	☐	☐	☐	☐	☑	☐	☐	☐	☑	☐	☑	Active	5.14.0.3617
perfserver1	☐	☐	☐	☑	☐	☐	☐	☐	☐	☐	☐	Active	5.14.0.3617

Fallback for Audit Trail Server Role

- Audit Trail implementation changes
  - No changes required: Audit Trail service only writes to its database
  - Audit Trail Data must be stored in a (not file-based) Database

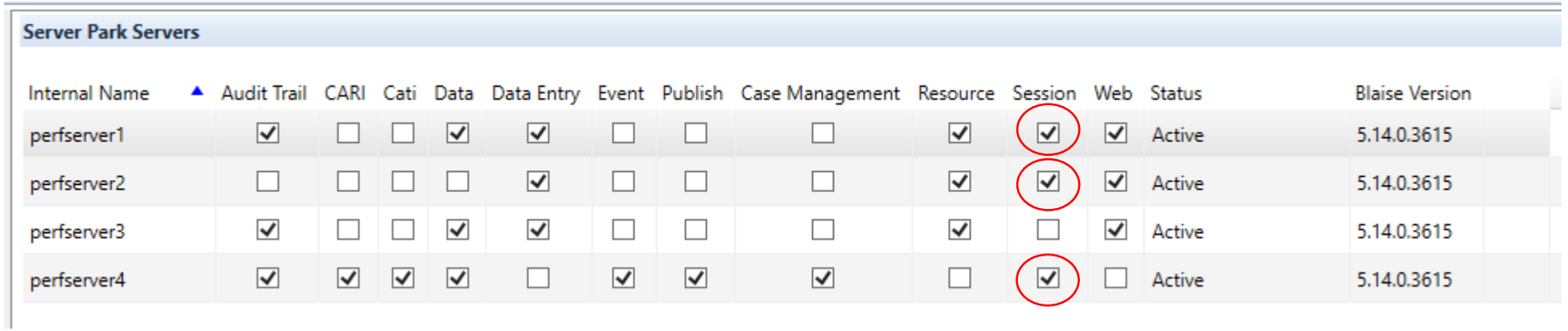


Server Park Servers

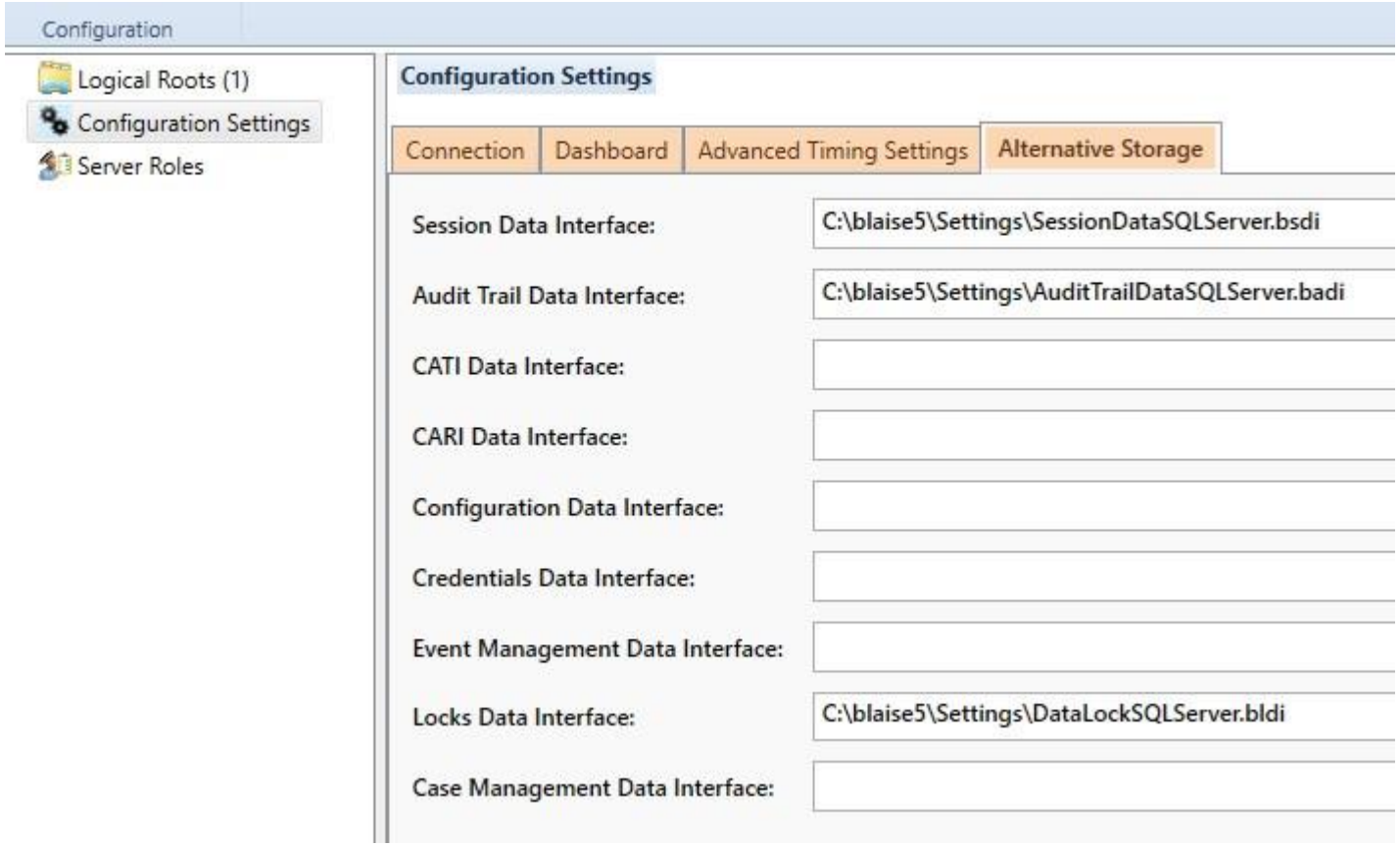
Internal Name	Audit Trail	CARI	Cati	Data	Data Entry	Event	Publish	Case Management	Resource	Session	Web	Status	Blaise Version
perfserver1	☑	☐	☐	☑	☑	☐	☐	☐	☑	☑	☑	Active	5.14.0.3615
perfserver2	☐	☐	☐	☐	☑	☐	☐	☐	☑	☑	☑	Active	5.14.0.3615
perfserver3	☑	☐	☐	☑	☑	☐	☐	☐	☑	☐	☑	Active	5.14.0.3615
perfserver4	☑	☑	☑	☑	☐	☑	☑	☑	☐	☑	☐	Active	5.14.0.3615

Fallback for Session Server Role

- Session Data must be stored in a (not file-based) Database.
- Implementation of Session Service has hardly changed:
  - In Memory updates to a Buffer
  - Periodic update of Session Database
  - Main reason: Preserve Performance



Fallback Server Roles - Database configuration



Architecture (1) – Blaise Cloud Solution

Architecture of the Blaise Cloud technology stack

- Docker
  - Containerization
- Kubernetes
  - Container workload management
- Apache Kafka
  - Messaging between containers and other applications
- Prometheus & Grafana
  - Collecting, monitoring, analyzing and visualizing metrics

Architecture (2) - Docker

Docker – Containerization

- Run anywhere, decouple application logic from underlying system
- Footprint is very small & deployment only takes seconds
- Using containers fits well in scalable solutions
- Blaise container images/apps are versioned and stored in a container registry (DockerHub)
- During deployment images are pulled from a container registry

Architecture (3) - Kubernetes

Kubernetes – Workload Management

- Kubernetes cluster consist of one or more nodes/worker machines where containers/apps are deployed in so called pods
- All Blaise apps (data entry, web,…) are deployed within pods
- Kubernetes continuously monitors the workload/metrics of the deployed application/services in the cluster
- Supports (auto-)scaling; both horizontal and vertical
- Supports file shares/persistent volumes and load-balancing
- Kafka brokers are also deployed in the cluster

Architecture (4) - Apache Kafka

Apache Kafka – Distributed Messaging

- Provides a unified, high-throughput, low-latency platform for handling real-time data feeds
- Apps sends messages on a topic to Kafka (producer/publish)
- Apps consume messages from a topic from Kafka (consumer/subscribe)
- Offers scalability out-of-the-box; create multiple consumers per topic
- Offers high availability out-of-the-box; deploy multiple brokers in a cluster. If one fails, the others will take over

Architecture (5) - Prometheus/Grafana

Prometheus & Grafana - Monitoring

- Prometheus
  - Metrics data collection
  - From Kubernetes/Kafka (CPU, Memory, IO and Network traffic)
  - From Blaise Cloud Apps (Messages, Requests/Responses and Duration)

- Grafana
  - Metrics data visualization

Deployment – Blaise (1)

- Blaise Apps / Containers
  - The following Blaise Services / Apps can be deployed in the cloud:
    - Data Entry, Audit Trail, Session, Resource, Data, Web and Cari
  - For now: no Manipula, no Data Entry Apps, no CMA/CMA Admin
    - It is possible to run CATI (web) surveys in the cloud, but then the CATI, Data, Session, Audit Trail roles have to be on-premise. Also the Dashboard runs on-premise.

Deployment – Blaise (2)

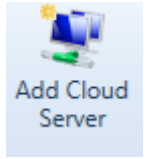
- First create an on-premise server and do not select server roles that you want to run in the cloud, e.g.

Server Park Servers

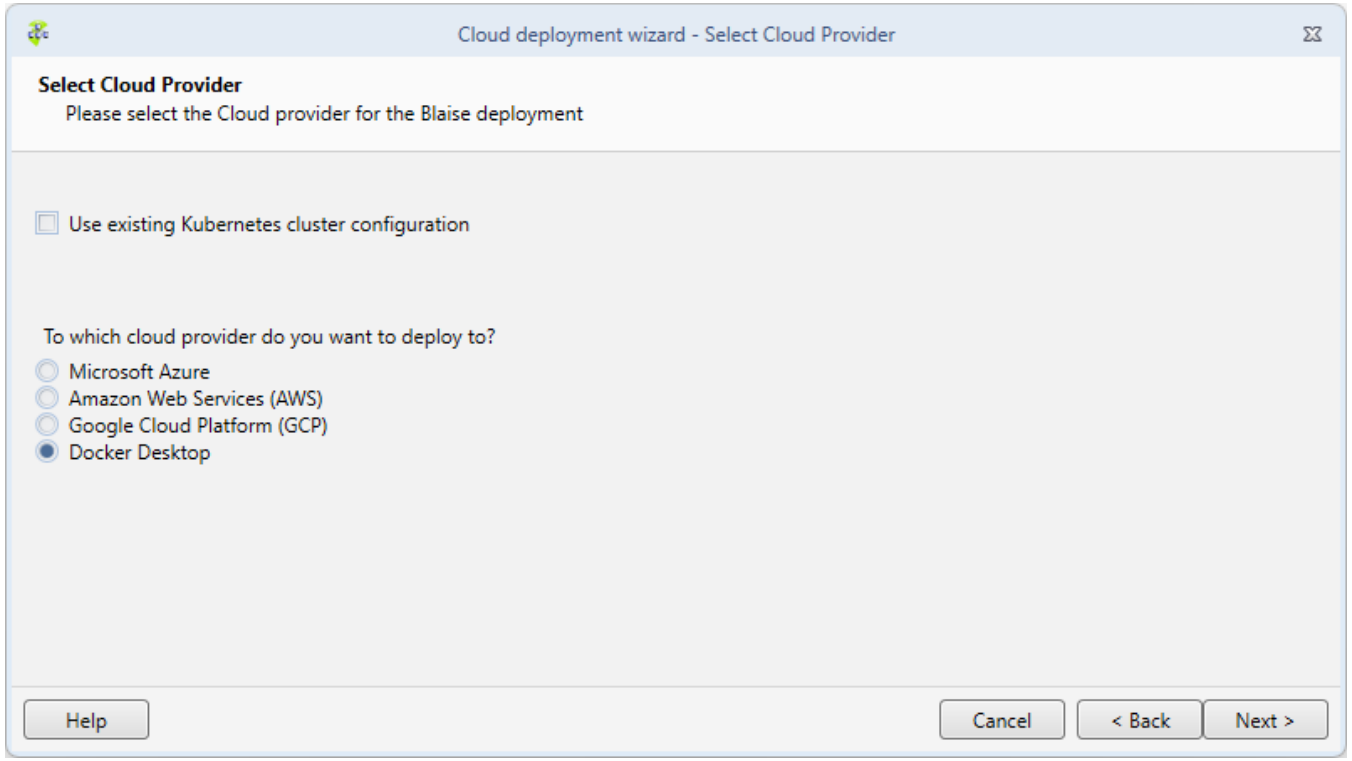
Internal Name	Audit Trail	CARI	Cati	Data	Data Entry	Event	Publish	Case Management	Resource	Session	Web	Status	Blaise Version
DESKTOP-0J1J8LU	☐	☐	☑	☐	☐	☑	☑	☑	☐	☐	☑	Active	5.14.0.0

- Remark: Select also the web role otherwise a deployed web survey in the cloud fails to start

Deployment – Blaise (3)

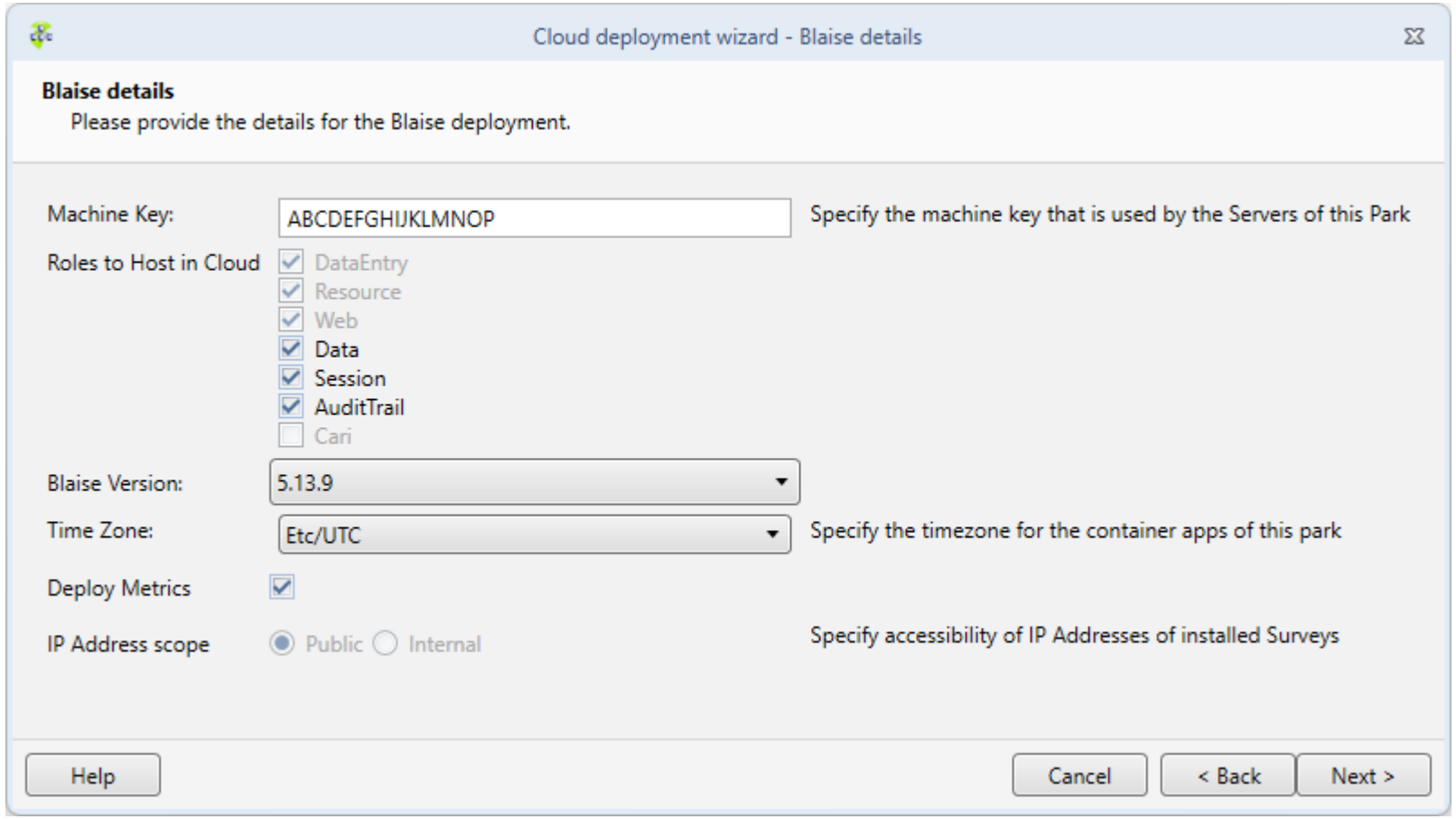
- Click the 

button to start the Cloud deployment Wizard



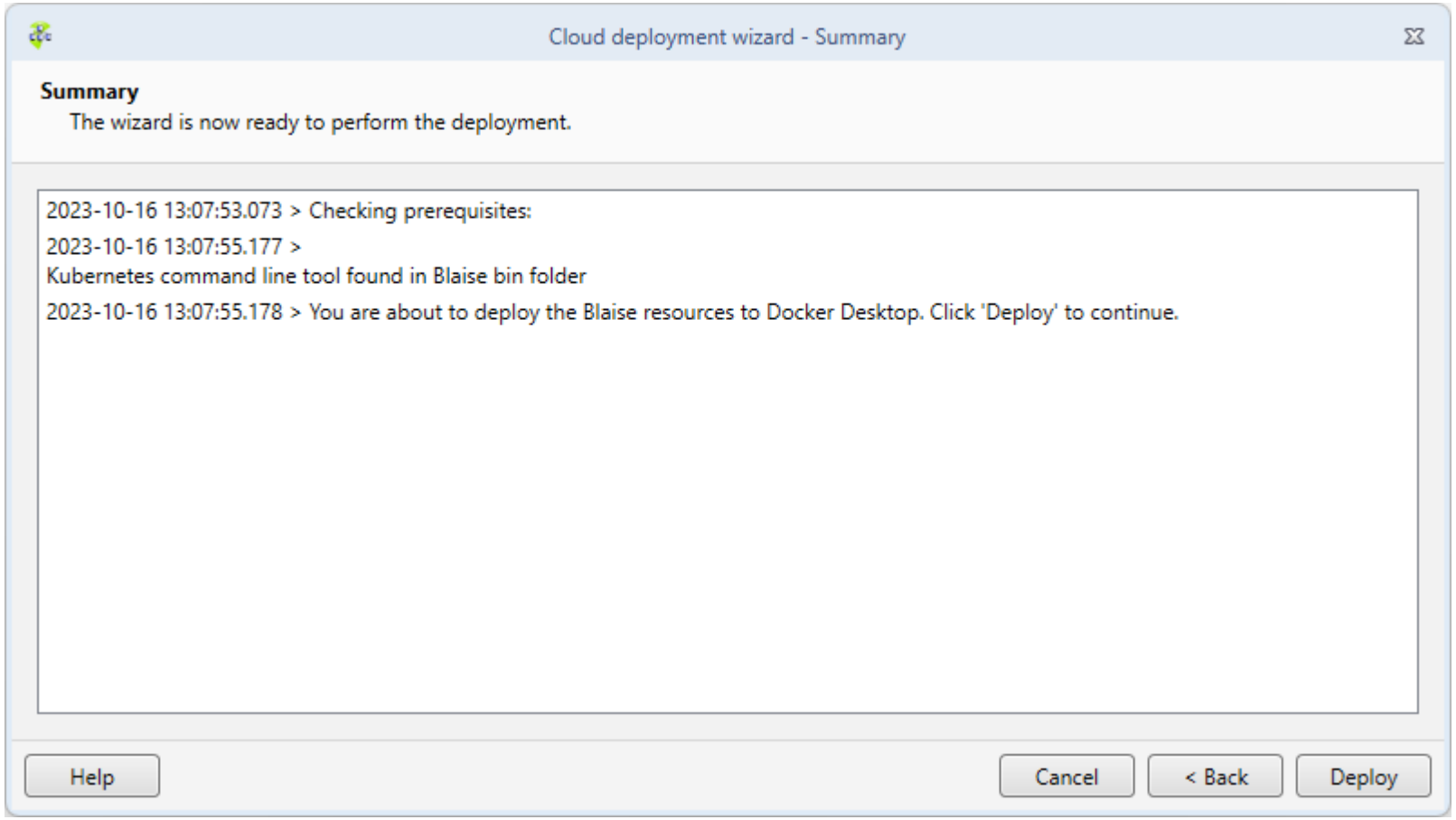
Deployment – Blaise (4)

- Select the Blaise Version for the container



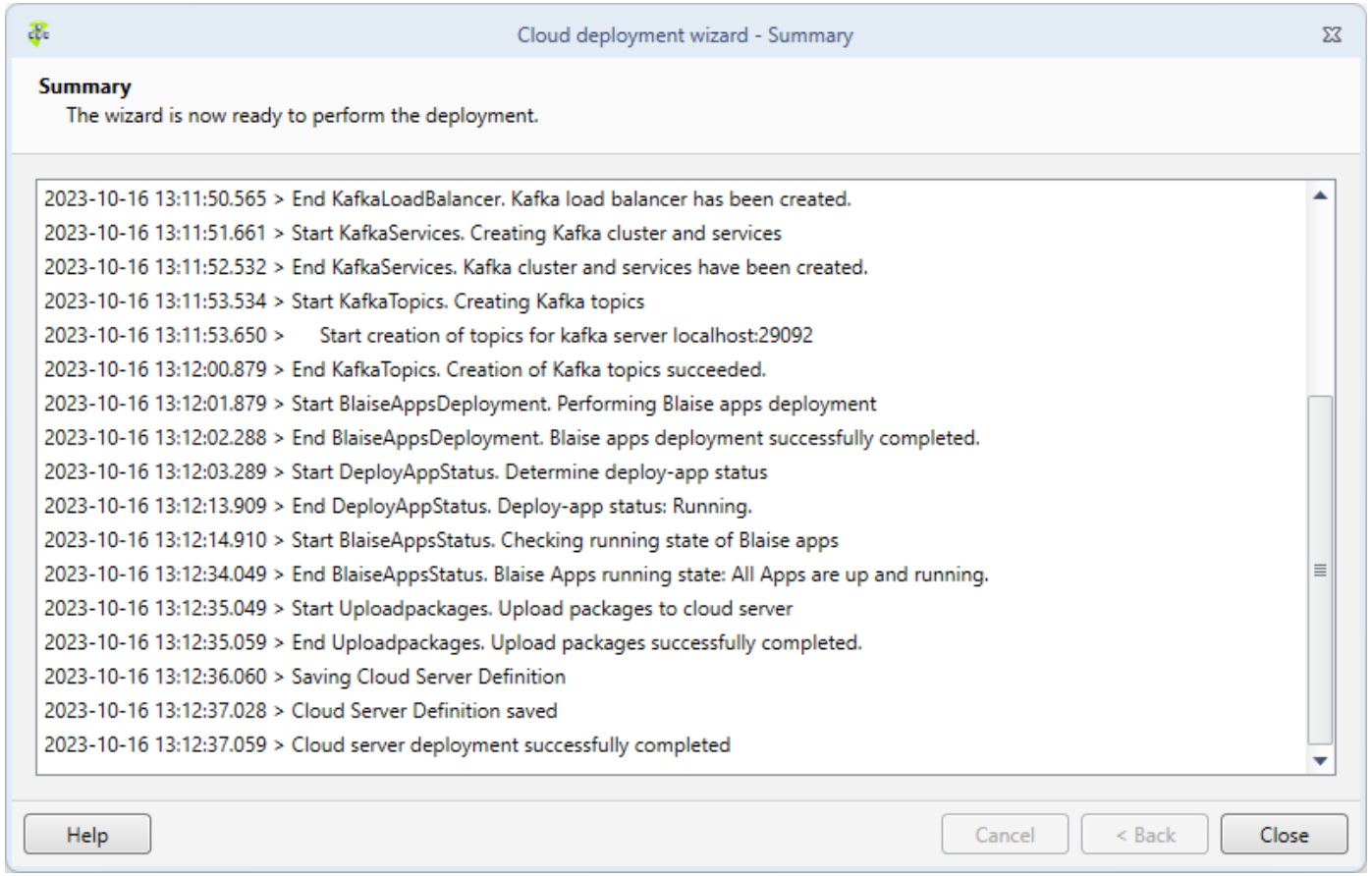
Deployment – Blaise (5)

- Click ‘Deploy’ to start the deployment



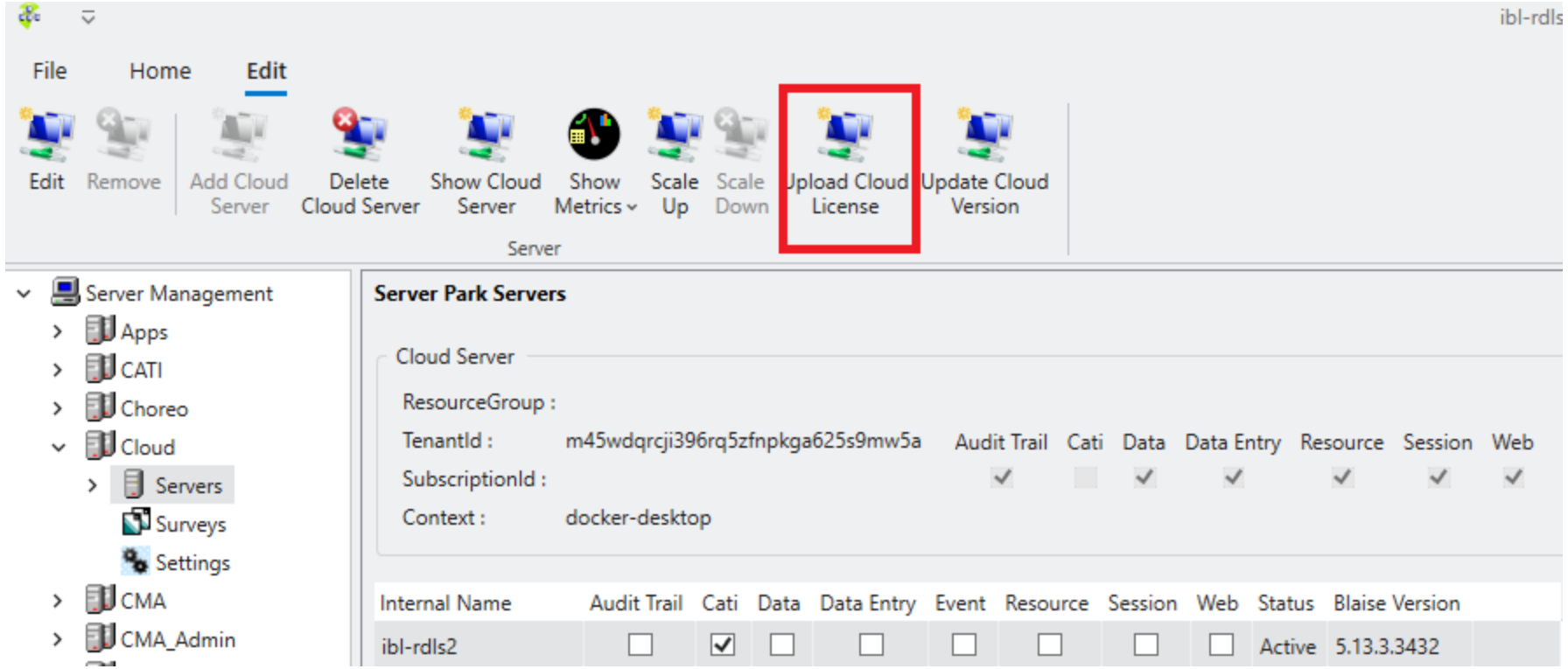
Deployment – Blaise (6)

- View the progress of the deployment and click ‘Close’ to end the wizard



Deployment – Blaise (7)

- Now the Blaise license key has to be deployed to the cloud version.
- In the ribbon, click on Upload Cloud License

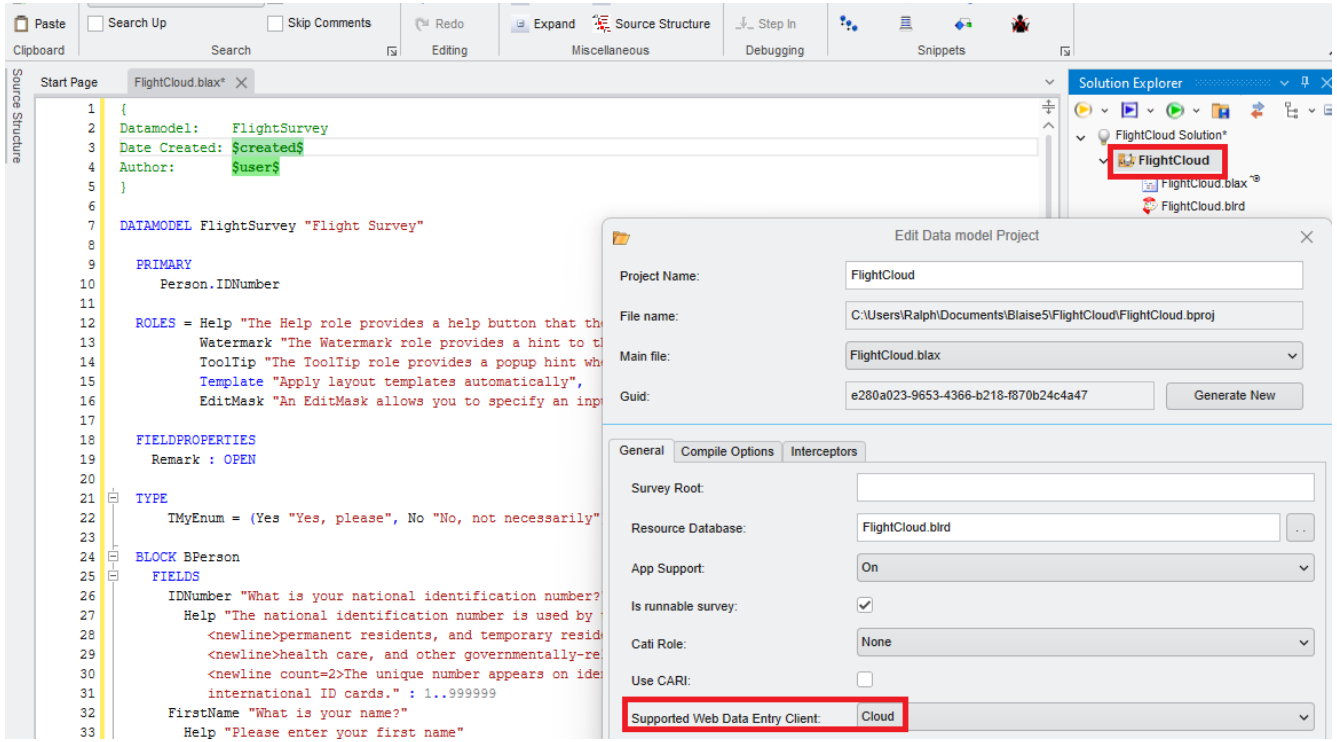


Deployment – Blaise (8)

- Demo
  - Deployment Blaise on Docker Desktop
  - Upload Blaise License

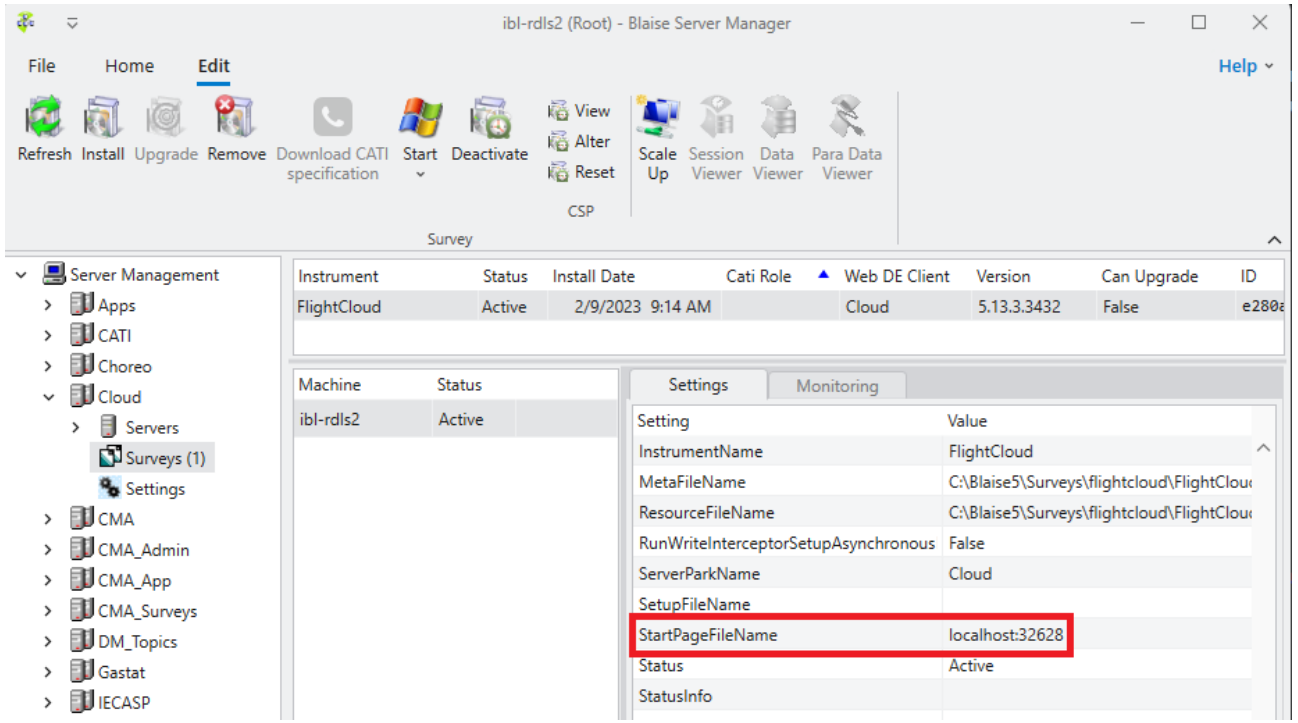
Deploy a (web) survey to the cloud (1)

- Create a (web) survey in the Control Centre
- Right click the project, select Edit Project and set the ‘Supported Web Data Entry Client’ to: Cloud



Deploy a (web) survey to the cloud (2)

- Install the survey in the cloud server park



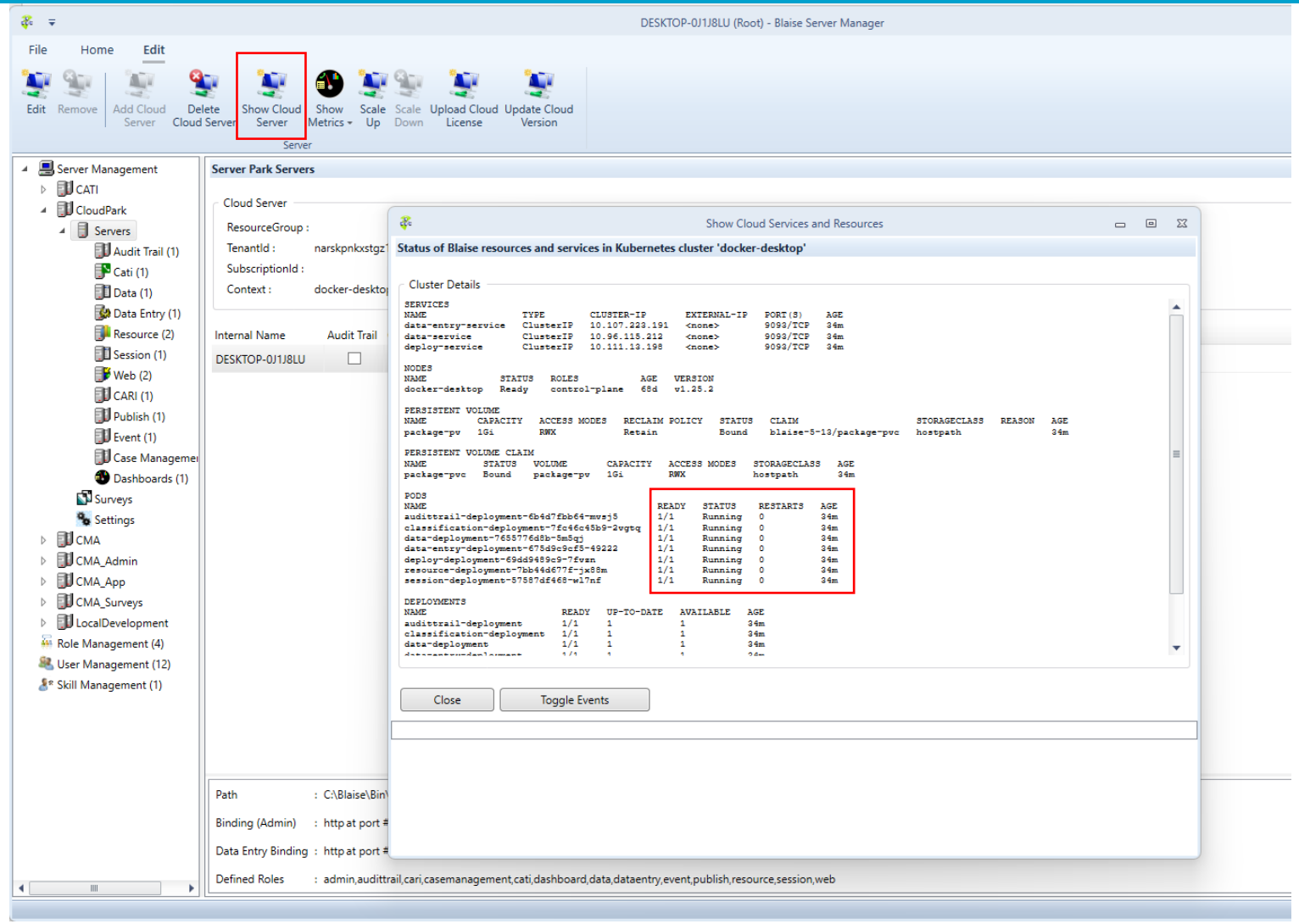
- If everything went well, you should see a StartPageFileName pointing to the cloud-address
- Now you can test it from the start-button in the Ribbon

Deploy a (web) survey to the cloud (3)

- Demo
  - Deployment of Flight Survey
    - Start survey
  - Deployment of Leisure Survey
    - Start survey

Monitoring – Check running state (1)

- Use‘Show Cloud Server’

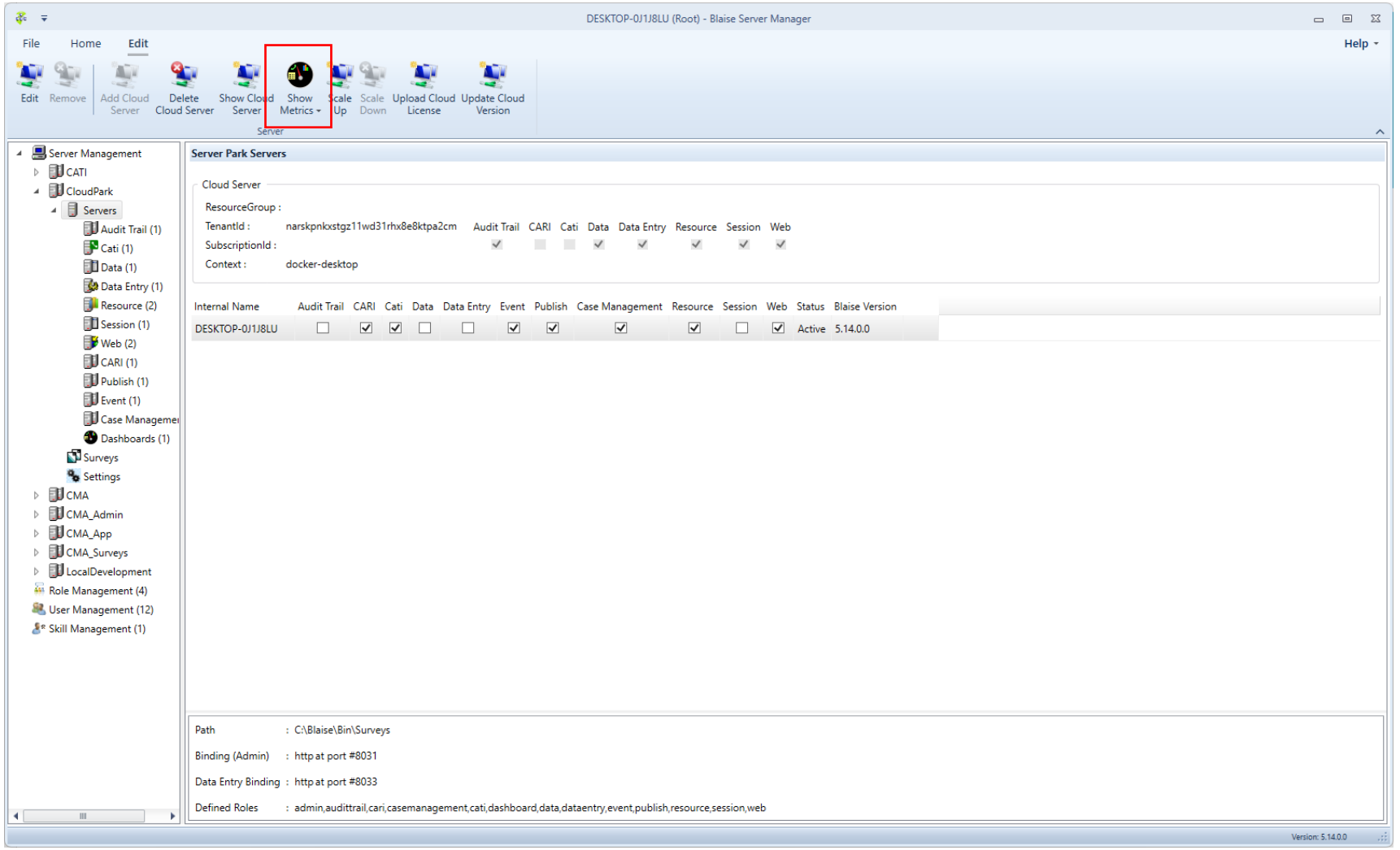


Monitoring - Check running state (2)

- Use Kubectl command line
- Some useful commands:
  - kubectl get all --all-namespaces
  - kubectl get nodes
  - kubectl get pv
  - kubectl get pvc -n <name space name> or --all-namespaces
  - kubectl describe pod <pod name> -n <name space name>
  - kubectl logs <pod name> -n <name space>

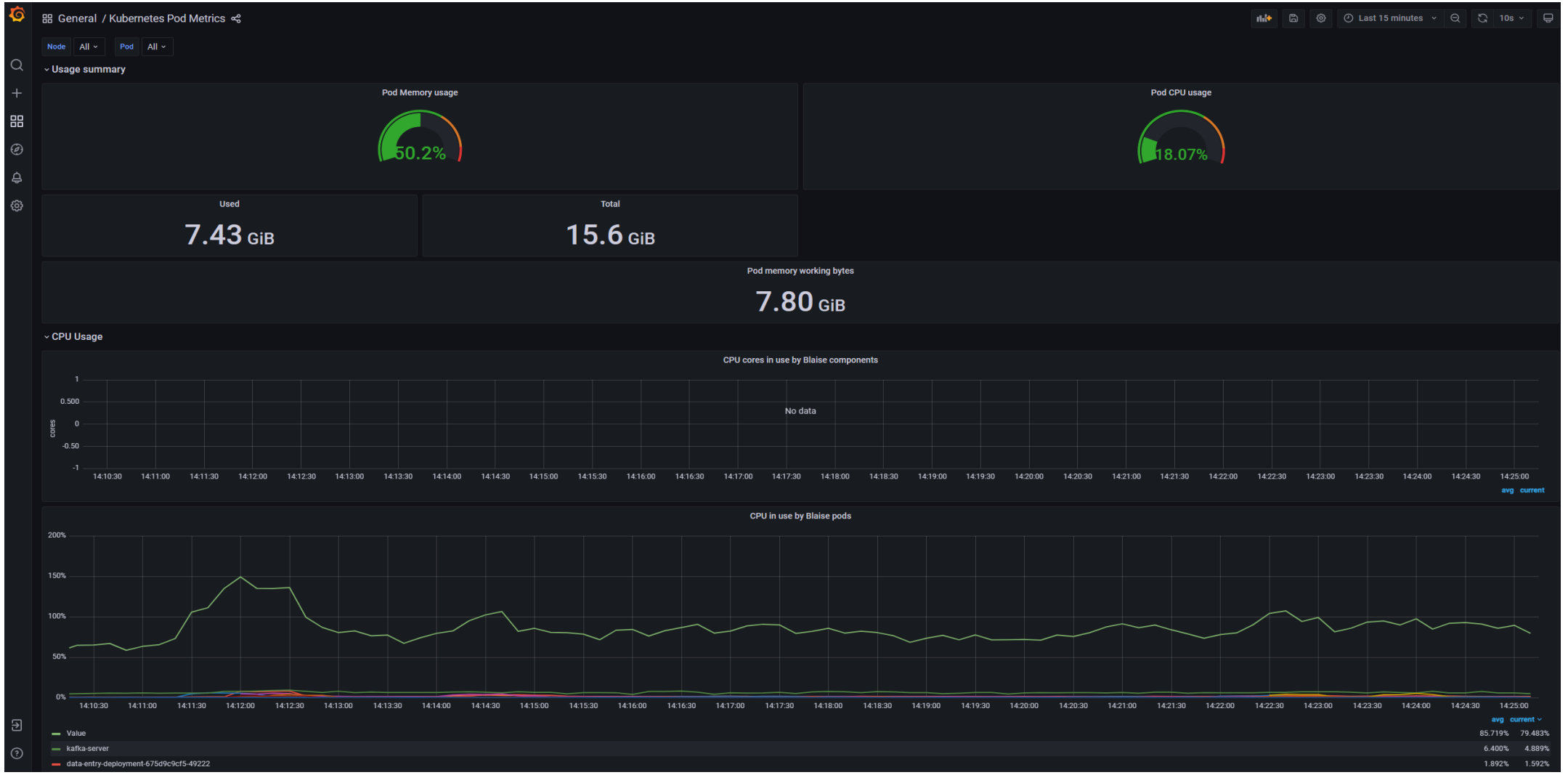
Monitoring - Check running state (3)

- Use ‘Show Metrics’
  - This option is only available if you
  - chose to ‘Deploy Metrics’ in the cloud
  - Deployment wizard
  - During the deployment 3 Grafana
  - dashboards will be installed



Grafana (1) – Blaise Cloud dashboard

Grafana (2) – Kubernetes Pod Metrics



Grafana (3) – Kafka Metrics

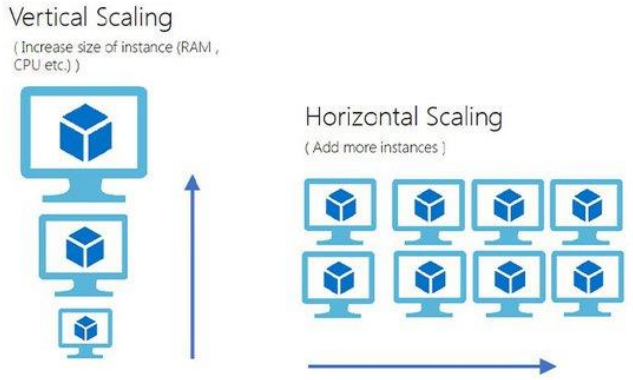
Monitoring

- Demo
  - ServerManager: Show Cloud Server
  - Usage Kubectl
  - ServerManager: Show Metrics
    - Blaise Cloud Dashboard
    - Kubernetes Pod Metrics
    - Kafka Metrics

Scaling (1)

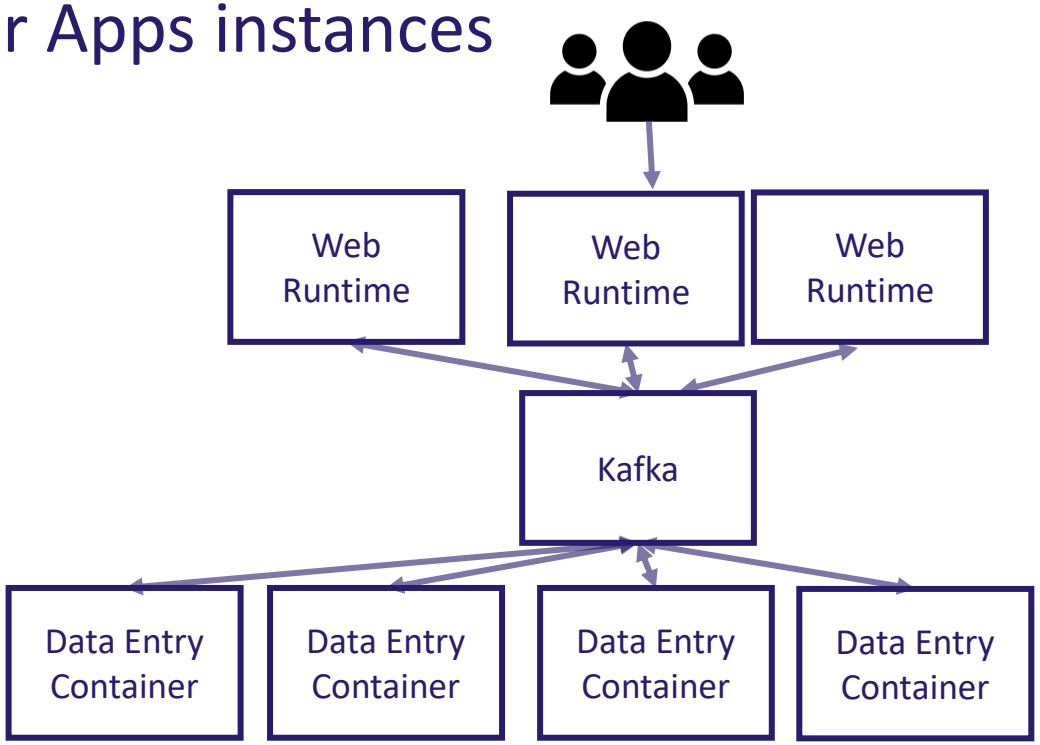
- Cloud: Dynamic allocation of resources
- Vertical vs Horizontal
  - Vertical scaling: add more compute power to your existing nodes
  - Horizontal scaling: add more compute power by adding instances

- Blaise system supports horizontal
- The cloud provider supports vertical



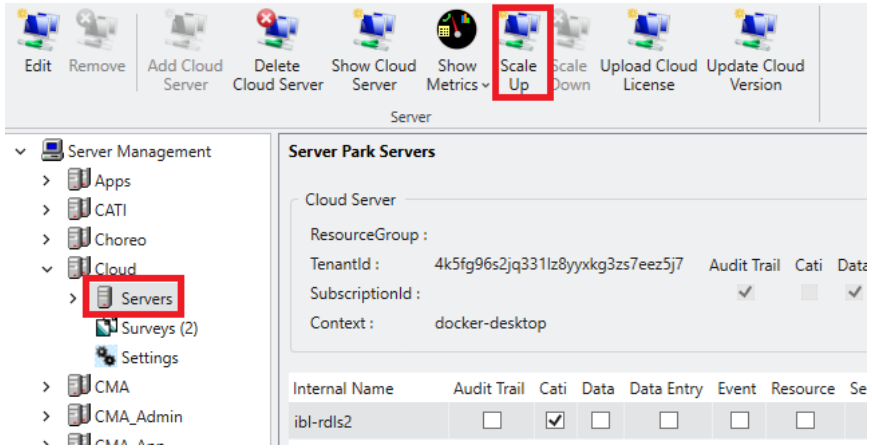
Scaling (2)

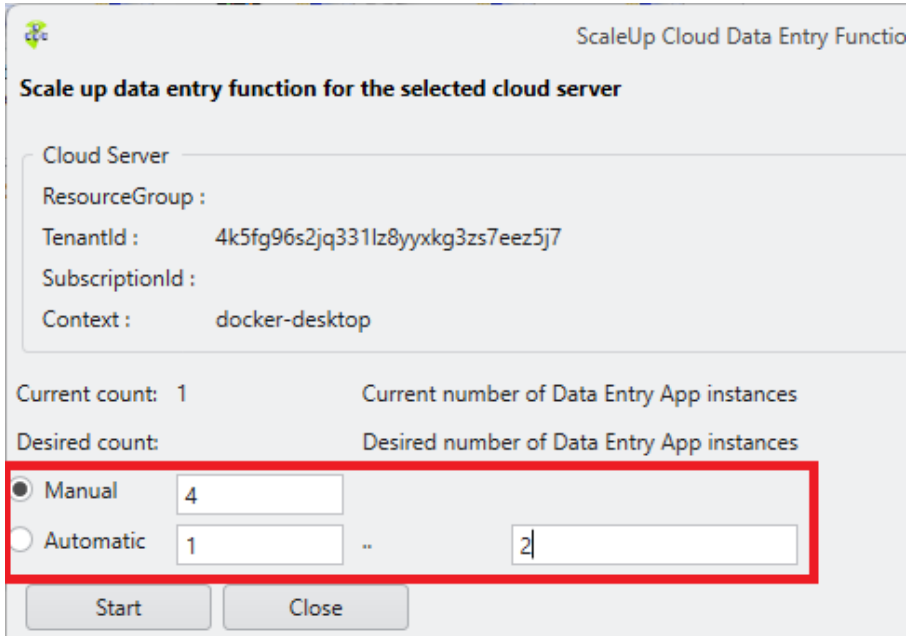
- Blaise supports scaling in Server Manager
- Data Entry Container apps (automatic or manually)
- Web Survey Container Apps instances



Scaling in the Server Manager

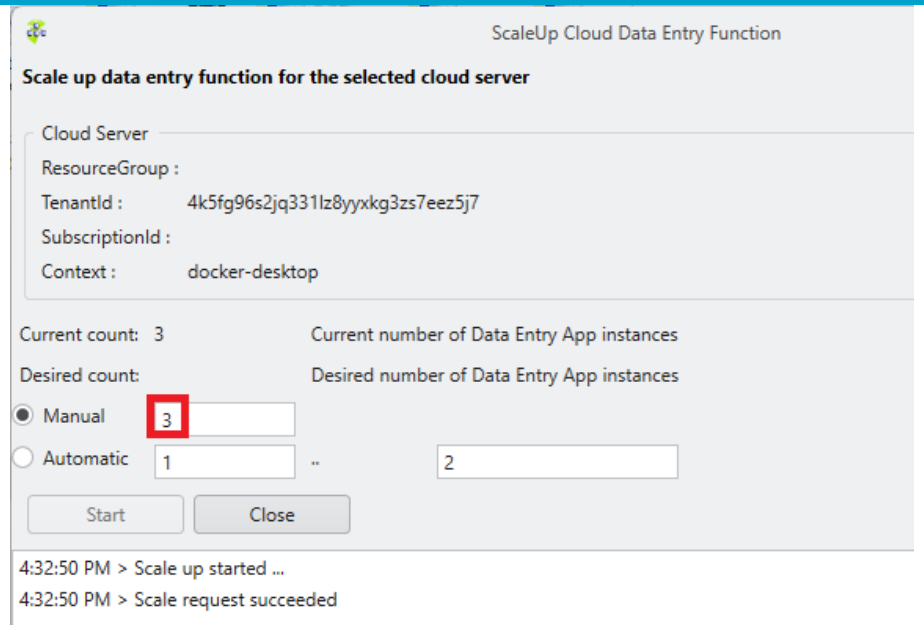
- Data Entry: choose ‘Servers’ and click on ‘Scale Up’
  - Manual : Instruct Kubernetes to Create/Remove instances of a pod
  - Automatic: Use Kubernetes ‘HorizontalPodAutoscaler’ functionality to Create/Remove instances
    - --cpu-percent=70% --minPods=1 --maxPods=2





Scaling Data Entry - Kubectl

- Scale up Data Entry

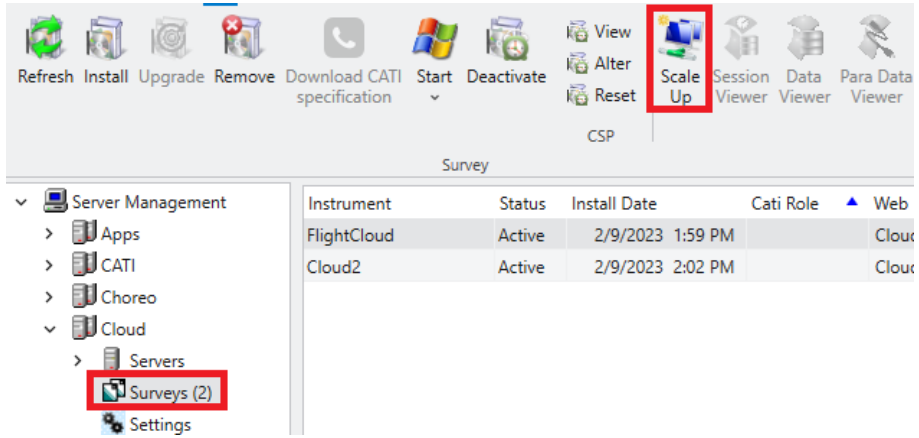


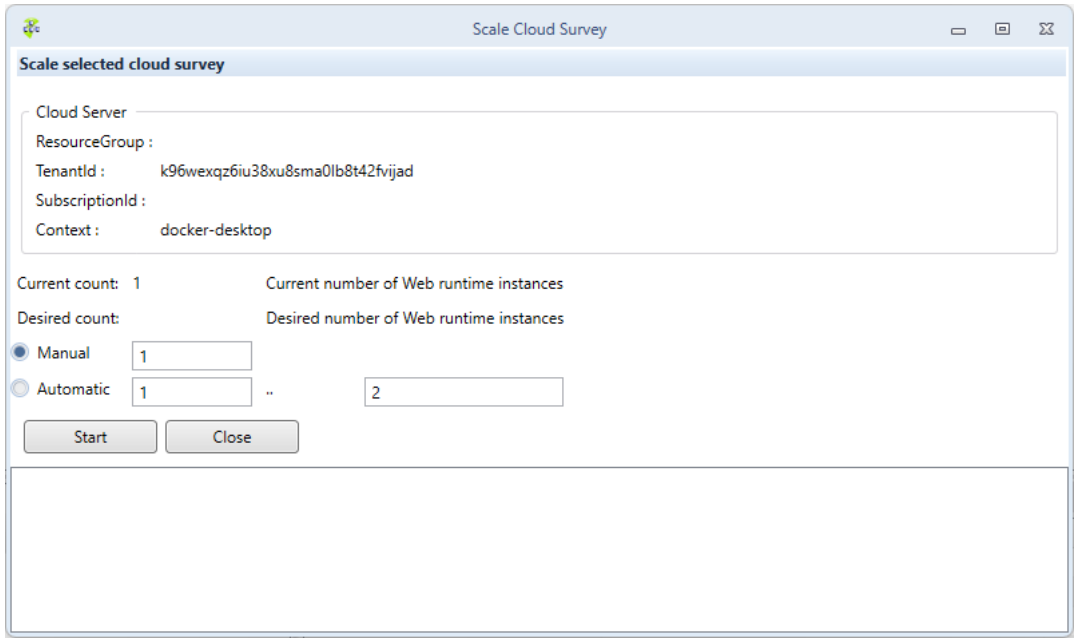
- Check with command: Kubectl get all --all-namespaces

```
blaise-5-13   pod/cari-deployment-854d55fdc7-lrv8n                1/1   Running   0   145m
blaise-5-13   pod/classification-deployment-5ff74c47b-nktfb       1/1   Running   0   145m
blaise-5-13   pod/cloud2-web-deployment-7c66dd945d-cmrlw          1/1   Running   0   141m
blaise-5-13   pod/data-deployment-857d5c495f-6s2xp                1/1   Running   0   145m
blaise-5-13   pod/data-entry-deployment-8474585cd-b5rgh           1/1   Running   0   79s
blaise-5-13   pod/data-entry-deployment-8474585cd-flvmt           1/1   Running   0   145m
blaise-5-13   pod/data-entry-deployment-8474585cd-ns9qk           1/1   Running   0   79s
blaise-5-13   pod/deploy-deployment-64fc776f94-fwp7d              1/1   Running   0   145m
blaise-5-13   pod/flightcloud-web-deployment-6784c495bc-75v9b     1/1   Running   0   144m
blaise-5-13   pod/resource-deployment-5866f6f46-sgx28             1/1   Running   0   145m
```

Scaling in the Server Manager

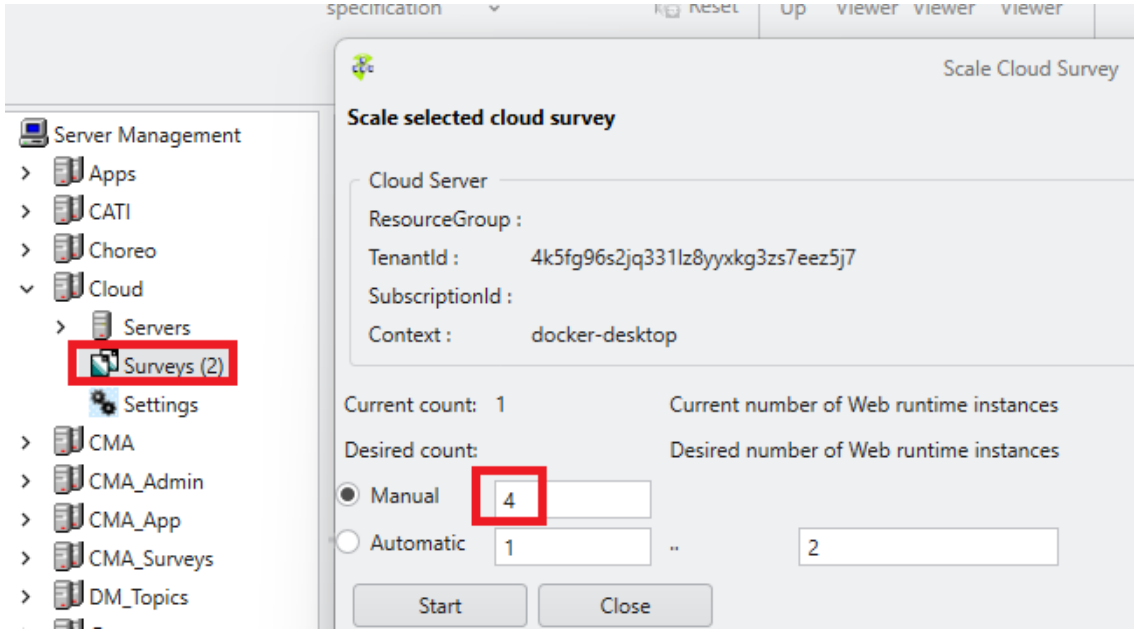
- Web: choose ‘Surveys’ and click the button ‘Scale Up’
  - Manual : Instruct Kubernetes to Create/Remove instances of a pod
  - Automatic: Use Kubernetes ‘HorizontalPodAutoscaler’ functionality to Create/Remove instances
    - --cpu-percent=70% --minPods=1 --maxPods=2





Scaling Web- Kubectl

- Scale up Web



- Check with command: Kubectl get all --all-namespaces

```
blaise-5-13   pod/flightcloud-web-deployment-6784c495bc-5m2td   0/1   Running             0   4s
blaise-5-13   pod/flightcloud-web-deployment-6784c495bc-75v9b   1/1   Running             0   18h
blaise-5-13   pod/flightcloud-web-deployment-6784c495bc-rjlqc   0/1   ContainerCreating   0   4s
blaise-5-13   pod/flightcloud-web-deployment-6784c495bc-tt24m   0/1   Running             0   4s
```

Scaling

- Demo
  - Scale Up – Data Entry instances
  - Scale Up – Web Survey instances

Update Blaise in the Cloud (1)

- Blaise updates:
  - Update a web runtime to a minor version
  - Update a complete park to a newer minor version
    - Up to 5.11 the system only updates the default Blaise apps
    - From 5.12 the system also automatically update the running instances of the web-runtime
  - Update a side by side installation

Update Blaise in the Cloud (2)

- How to see in kubectl what the minor version is?
  - E.g. take the data-entry-pod in the namespace blaise-5-13

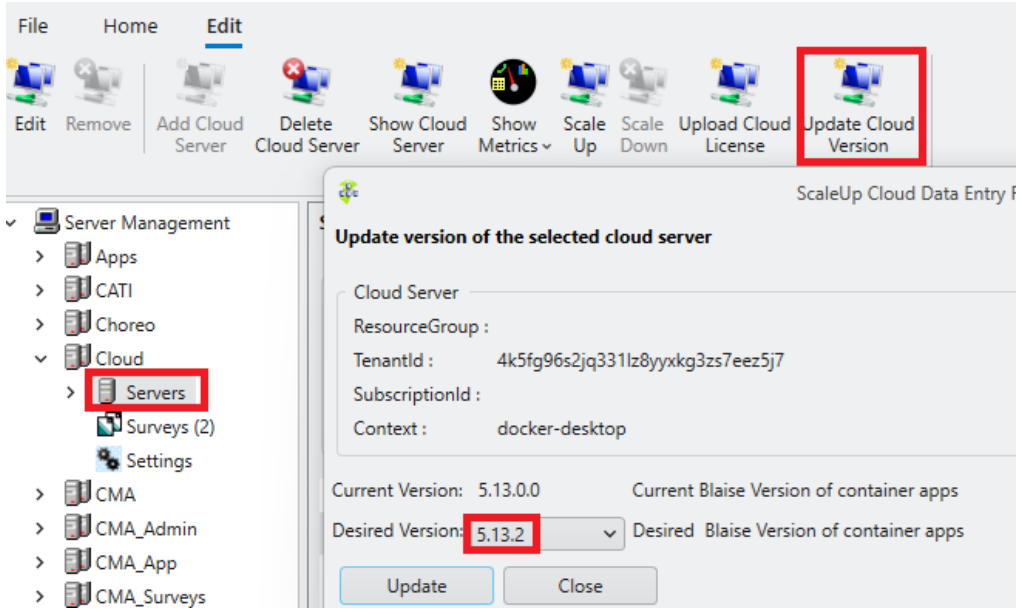
```
blaise-5-13   pod/cloud2-web-deployment-7c66dd945d-cmrlw
blaise-5-13   pod/data-deployment-857d5c495f-6s2xp
blaise-5-13   pod/data-entry-deployment-8474585cd-flvmt
blaise-5-13   pod/deploy-deployment-64fc776f94-fwp7d
```

- kubectl describe pod/data-entry-deployment-8474585cd-flvmt -n blaise-5-13

```
Containers:
  data-entry-app:
    Container ID:   docker://d4efa801313ea437886298ae6b723e8
    Image:          blaisecbs/dataentry:5.13.0
    Image ID:       docker-pullable://blaisecbs/dataentry@sh
```

Update Blaise in the Cloud (3)

- Update the Blaise version (upgrade / downgrade)




```
Containers:
  data-entry-app:
    Container ID:   docker://b6055ddaf9af98d61ea704b0cbb8a7c45
    Image:          blaisecbs/dataentry:5.13.2
    Image ID:       docker-pullable://blaisecbs/dataentry@sha2
da3090bb1
```

Update Blaise in the cloud (4)

- Demo
  - Update Blaise version

Side-by-side installation

- Allow to install multiple Blaise Versions in one Kubernetes Cluster
  - For instance:
    - Blaise 5.12
    - Blaise 5.13
  - Each Blaise version has its own:
    - Namespace for Blaise Container Apps
    - Storage (Persistent Volume Claim)
    - Message channels (Kafka Topics)

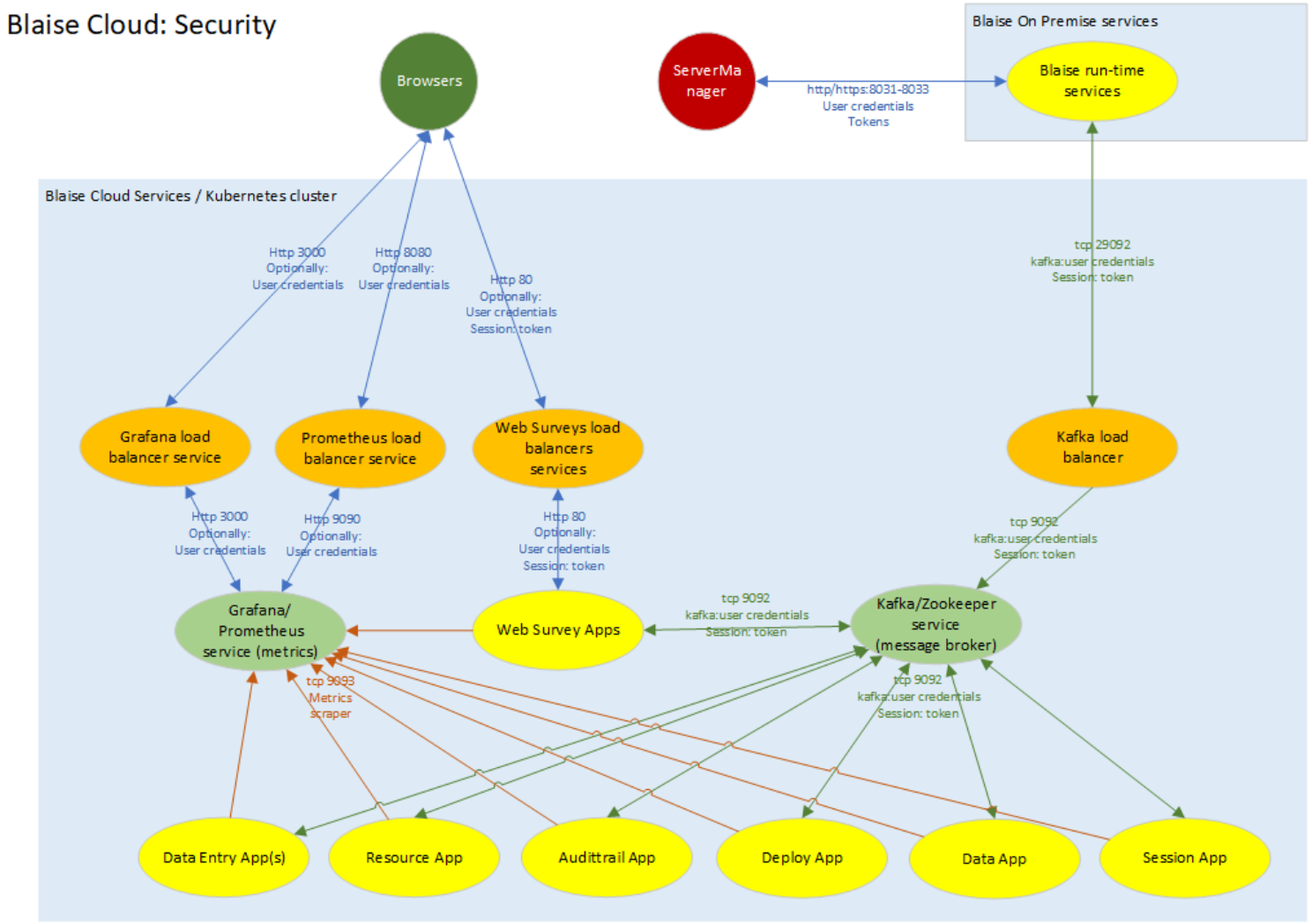
Security (1)

- Subjects that play a role in setting up security for the Blaise Cloud solution are:
  - Communication protocols being used e.g. http, https, tcp
  - Ports being used and exposed
  - User credentials required for accessing services & applications
  - Usage of tokens in order to secure user sessions

Security (2)

- Blaise cloud solution - Kubernetes
  - Blaise cloud apps expose only those ports that are required for them to run properly
  - Communication between Blaise cloud apps is handled by Kafka
  - Communication between Blaise cloud apps & services and external app & services is handled via load-balancers
  - Blaise cloud apps will only process an incoming message if a valid token is sent in the message header
  - Kafka will only process an incoming message if the sending app provides correct credentials

Security (3)


Blaise Cloud: Security
Browsers
ServerManager
http/https:8031-8033
User credentials
Tokens
Blaise On Premise services
Blaise run-time services
Blaise Cloud Services / Kubernetes cluster
Http 3000
Optionally:
User credentials
Http 8080
Optionally:
User credentials
Http 80
Optionally:
User credentials
Session: token
tcp 29092
kafka:user credentials
Session: token
Grafana load balancer service
Prometheus load balancer service
Web Surveys load balancers services
Kafka load balancer
Http 3000
Optionally:
User credentials
Http 9090
Optionally:
User credentials
Http 80
Optionally:
User credentials
Session: token
tcp 9092
kafka:user credentials
Session: token
Grafana/ Prometheus service (metrics)
Web Survey Apps
tcp 9092
kafka:user credentials
Session: token
Kafka/Zookeeper service (message broker)
tcp 9693
Metrics scraper
tcp 9092
kafka:user credentials
Session: token
Data Entry App(s)
Resource App
Audittrail App
Deploy App
Data App
Session App

Security (4)

Kubernetes cluster : Protocols, Port and security

Kind	From/to	From/to	Protocol	Internal port	External port	Secured by	Load balancer
Internal	Blaise cloud app	Blaise cloud app	Tcp Kafka	9092	-	credentials tokens	N
Internal	Blaise cloud app	Prometheus (metrics) service	http	9093	-	-	N
External	Blaise run-time services on prem	Blaise cloud app	tcp Kafka	9092	29092	credentials tokens	Y
External	Browsers	Blaise cloud web apps	http	80	80	tokens optionally: credentials	Y
External	Browsers	Grafana service	http	3000	3000	optionally: credentials	Y
External	Browsers	Prometheus service	http	9090	8080	optionally: credentials	Y

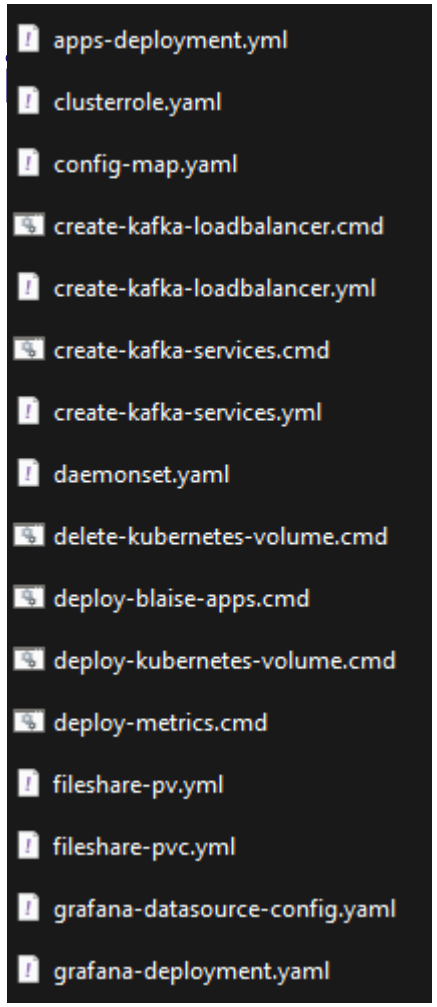
Troubleshooting Deployment (1)

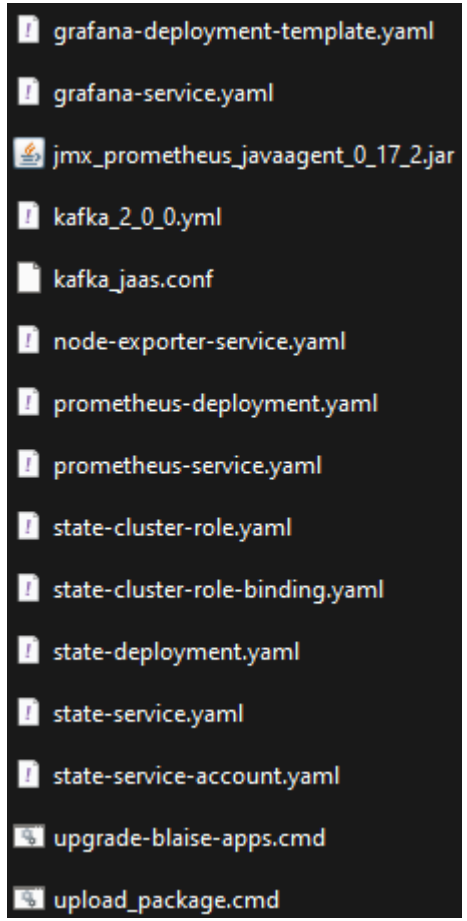
Cloud Installation

- How is Blaise cloud solution deployed by ServerManager?
  - Usage of template cmd / yaml script files
  - YAML configuration files are used for creating and configuring components in Kubernetes cluster
  - The template files contain placeholders which are actually being replaced by real values during installation/deployment
  - Located in Templates folder of Blaise installation folder
- Actual installation scripts and yaml files which have been created and applied by the ServerManager during deployment
  - Located in user's AppData\Local\Blaise folder
- Kube context
  - Context point to a certain Kubernetes cluster and contains the corresponding connection details
  - Located in config file in user's .kube folder

Troubleshooting Deployment (2)

- All the scripts that Blaise runs are located C:\Users\<name>\AppData\Local\Blaise





Thank you for your time

Gaining **deeper** understanding

www.blaise.com

blaise@cbs.nl

@blaisecbs

@Blaise5