Drone demo guide

In this tutorial, we will describe the deployment and execution of a drone demo in the Nalej Platform (v0.5.1).

You will create an architecture with development and production backend instances and deploy a drone in each of them. Then, when everything is going smoothly, you will move the drone in the development backend to the production one. The steps to do so are:

1. Create 2 Device Groups.
2. Deploy 2 DroneBackendES instances (DevelopmentBackend and ProductionBackend).
3. Configure the Kibana Dashboard for each DroneBackendES, so we can display the information sent by the drones.
4. Deploy 2 DroneBackendLocal instances (each associated to a DroneBackendES instance and to a Device Group).
5. Deploy 2 Virtual-Drone instances (each associated to a Device Group).
6. Change the connection from the DevelopmentBackend to the ProductionBackend to have both drones there.

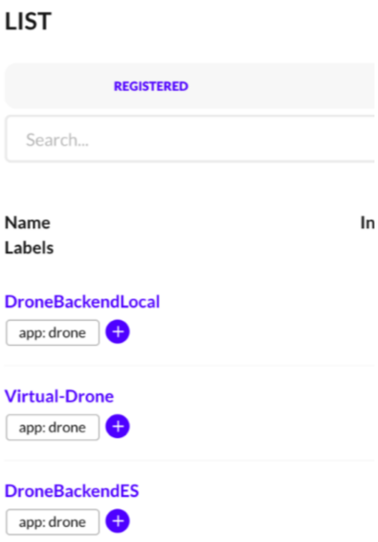
Prerequisites

We assume that no other applications are deployed in the demo environment.

This demo requires three descriptors:

- DroneBackendES
- DroneBackendLocal
- Virtual-Drone

At this point of the Descriptor Creation course, you should have downloaded these files and worked on them. If not, please download them from the course documentation and complete the proposed exercises. After that, upload them to the Nalej Platform but do not deploy any instances yet (you will be prompted to do so during the tutorial).



LIST

REGISTERED

Search...

Name In
Labels

DroneBackendLocal
app: drone

Virtual-Drone
app: drone

DroneBackendES
app: drone

There are two variables that are also needed from your demo environment:

- **Platform**: the URI of the environment without the prefix (e.g. release051.nalej.io)
- **Organization ID**: the UUID for the demo organization (e.g. b9114109-13fd-41c3-9189-f96066ecb0c6).

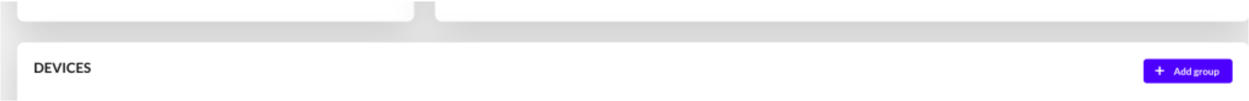
The engineering team will provide you with this information in case you don’t know it already.

You will also need the configuration for the default dashboard in Kibana, which is defined in a JSON file that is also among the material you need to download from the Descriptor Creation course documentation.

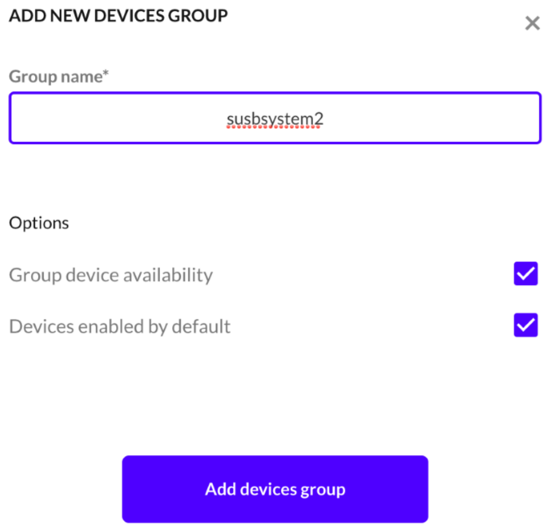
1. Creating the Device Groups

This demo requires two Device Groups, named “subsystem1” and “subsystem2”.

To create a new device group, you must go to the Devices section, and the click the “Add group” button.



A dialog like this one will appear, where you can introduce the name of the Device Group and establish the availability of the devices in this group and if they are enabled by default (please check both options).



Group name*

susbsystem2

Options

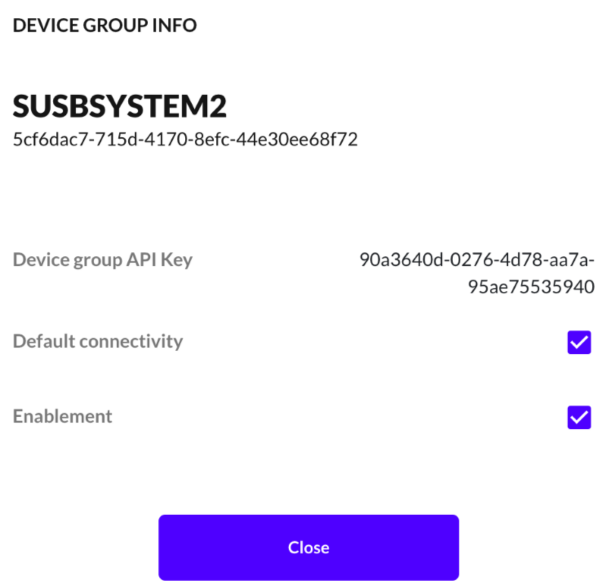
Group device availability ☑

Devices enabled by default ☑

Once the device groups have been created, they have several parameters that define them in the system. These parameters are:

- Device Group name (this is the name you chose).
- Device Group ID.
- Device Group API key.

You can access this information by clicking on the three dots that appear beside each Device Group in the list. As you can see in the image below, the Device Group ID is displayed under the Device Group name.



DEVICE GROUP INFO

SUSBSYSTEM2

5cf6dac7-715d-4170-8efc-44e30ee68f72

Device group API Key	90a3640d-0276-4d78-aa7a-95ae75535940
Default connectivity	☑
Enablement	☑

Once you know the parameters of both Device Groups, please create a file that contains all this information, with the following structure:

platform:
release051.nalej.io
OrganizationID:
b9114109-13fd-41c3-9189-f96066ecb0c6
deviceGroupName:
subsystem1
deviceGroupID:
98a389b4-f651-4aa8-9696-ad15c64bbcd3
Device group API Key
7f727c31-730a-4524-b296-fc537ee1493a

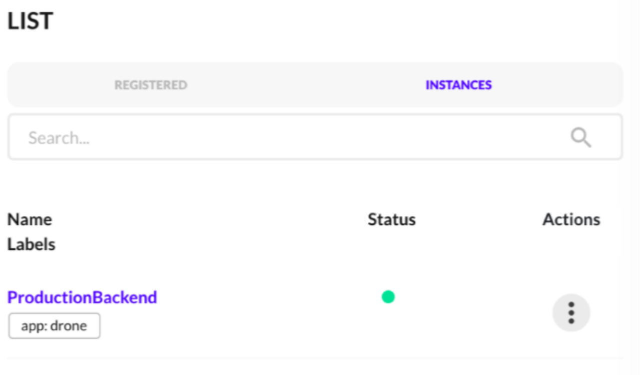
platform:
release051.nalej.io
OrganizationID:
b9114109-13fd-41c3-9189-f96066ecb0c6
deviceGroupName:
subsystem2
deviceGroupID:
a7d76c0c-0505-4b2d-ae8e-3605074cb332
Device group API Key
0dc41ad6-9ea4-4d4b-9171-298af7a9da3d

2. Deploying backend applications

Now you need to deploy two instances of the application DroneBackendES. One of them will be called ProductionBackend and the other will be DevelopmentBackend.

When both applications are deployed, you must go to the Kibana dashboard and configure them. So, for each instance, you must follow these steps:

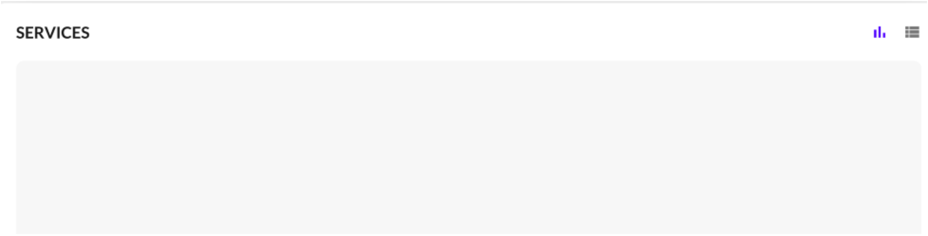
1. Click on the application instance in the list.



LIST

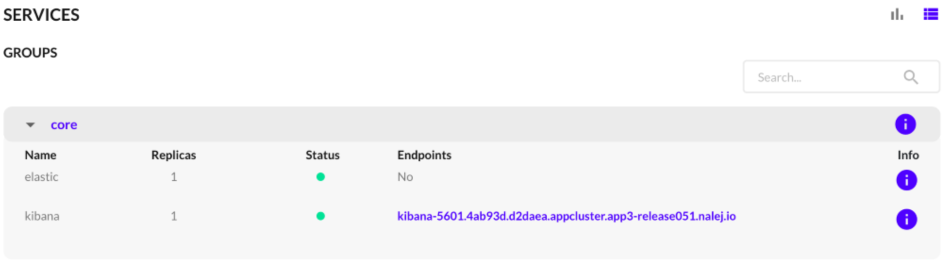
Name Labels	Status	Actions
ProductionBackend app: drone	●	⋮

2. Go to the detailed view.



SERVICES

3. Click on the URL that appears in the Kibana service section.



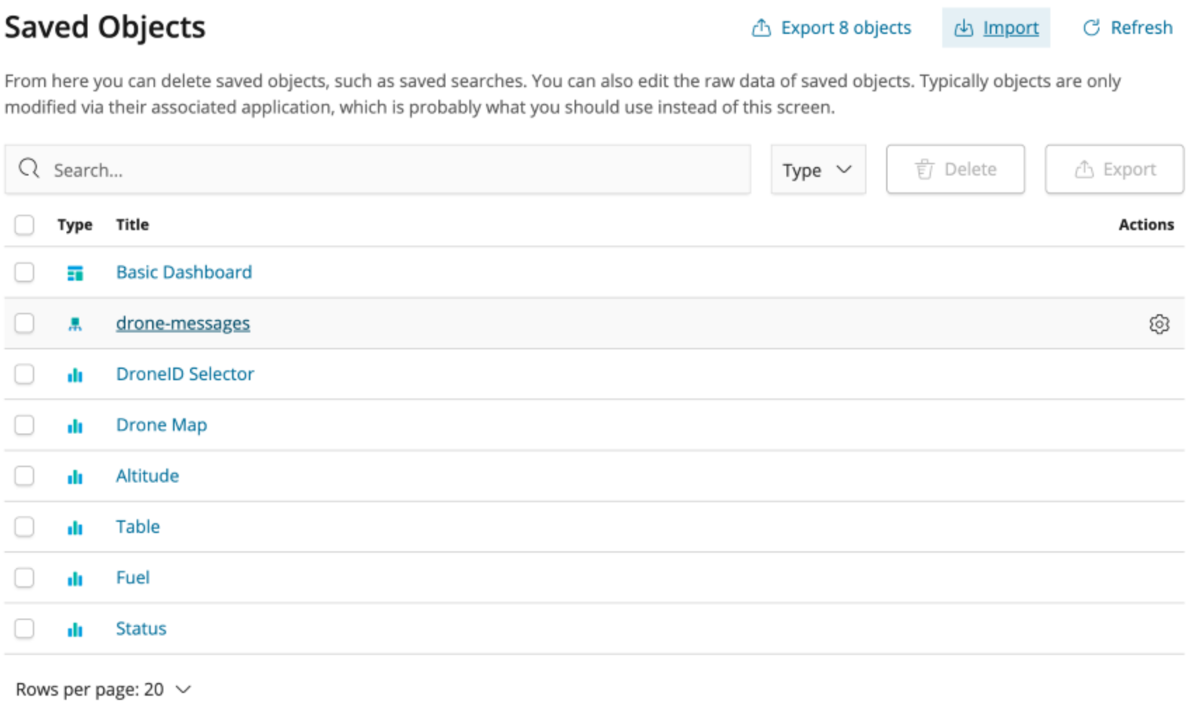
SERVICES

GROUPS

core

Name	Replicas	Status	Endpoints	Info
elastic	1	●	No	i
kibana	1	●	kibana-5601.4ab93d.d2daea.appcluster.app3-release051.nalej.io	i

4. Now, once in Kibana, you must load the default dashboard. o to the section “Management>Save Objects” and click on “Import”. Load the JSON file you downloaded at the beginning of the tutorial (named kibana_dashboard.json). This will include eight new objects:



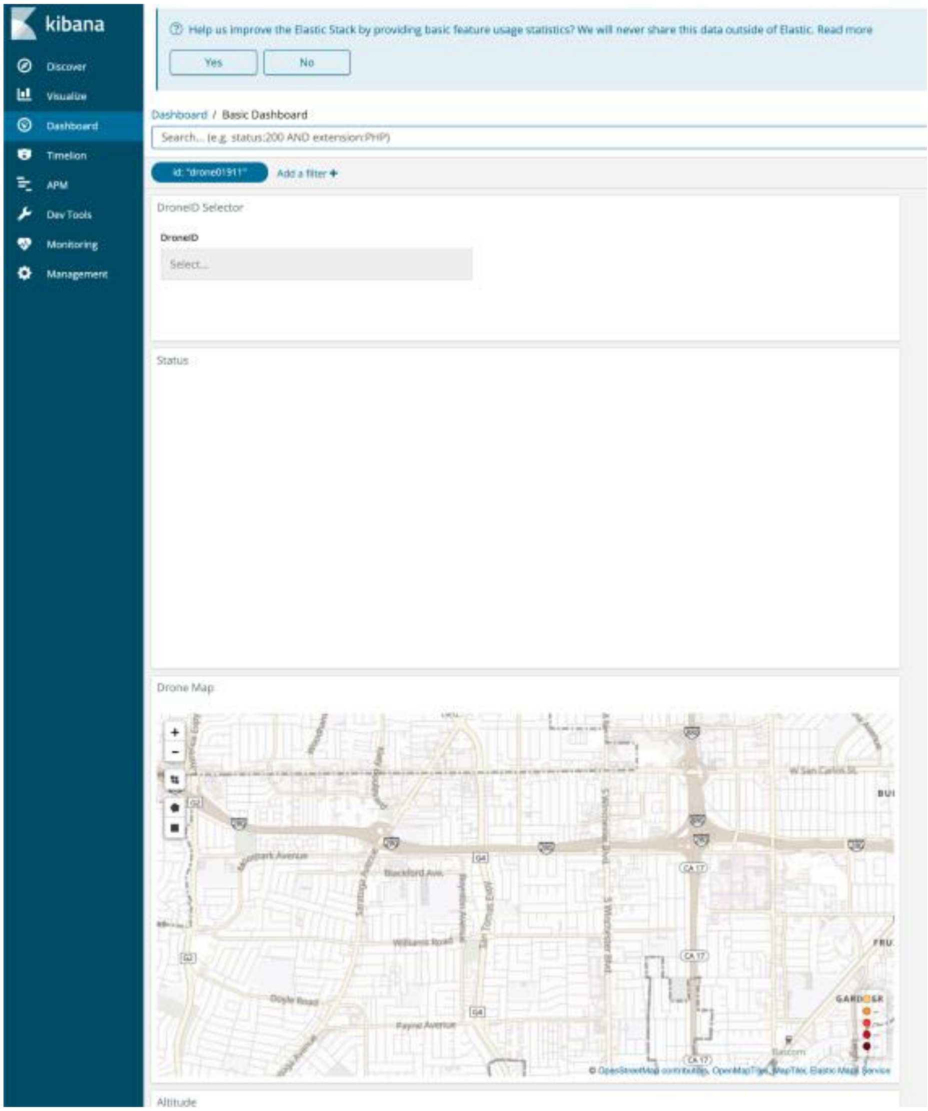
Saved Objects

From here you can delete saved objects, such as saved searches. You can also edit the raw data of saved objects. Typically objects are only modified via their associated application, which is probably what you should use instead of this screen.

□	Type	Title	Actions
□		Basic Dashboard	
□		drone-messages	⚙
□		DroneID Selector	
□		Drone Map	
□		Altitude	
□		Table	
□		Fuel	
□		Status	

Rows per page: 20

5. Now, when you go to the section Dashboard>Basic Dashboard, you can view it deployed. Remove the quick filters and configure the auto-refresh to get the view to update automatically.



Help us improve the Elastic Stack by providing basic feature usage statistics? We will never share this data outside of Elastic. Read more

Dashboard / Basic Dashboard

Search... (e.g. status:200 AND extension:PHP)

id: "drone01911"

DroneID Selector

DroneID

Select...

Status

© OpenStreetMap contributors, OpenMapTiles, MapTiler, Elastic Maps Service

Remember to follow these steps for both DroneBackend instances.

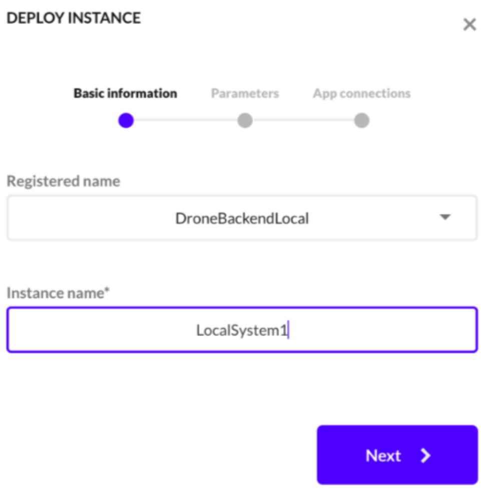
3. Deploying local applications

After deploying the Development and Production backend applications, you need to deploy the two DroneBackendLocal applications. These are named LocalSystem1 and LocalSystem2, and are configured as follows:

- LocalSystem1 is connected to the Device Group subsystem1, and its target instance is ProductionBackend.
- Meanwhile, LocalSystem2 is connected to the Device Group subsystem2, and its target instance is DevelopmentBackend.

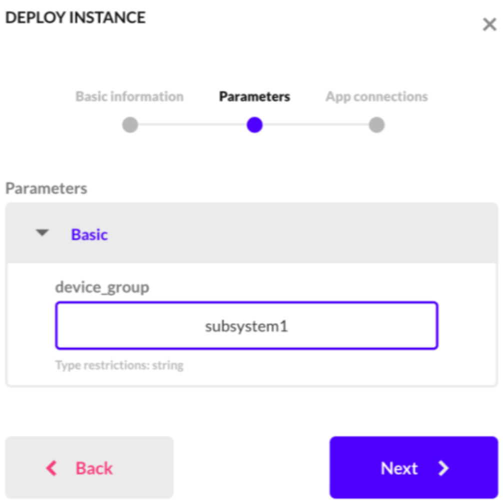
Below you have the screenshots of both deployments, so you can see where each connection is specified.

- LocalSystem1

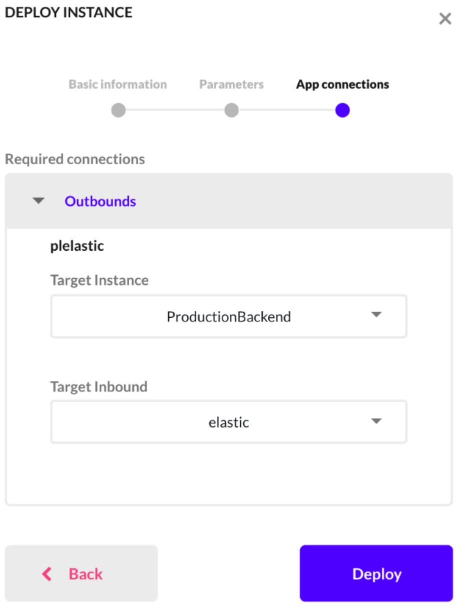


Registered name
DroneBackendLocal

Instance name*
LocalSystem1



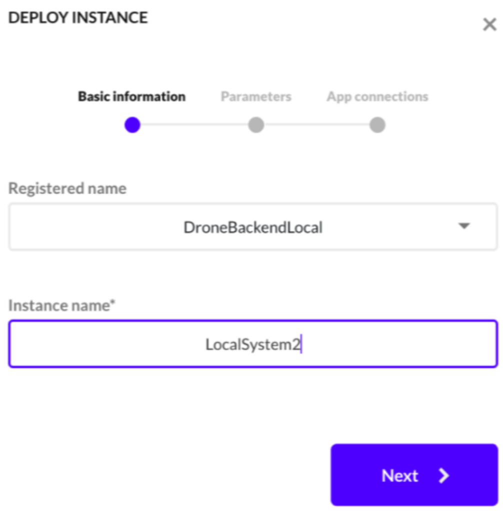
Parameters
Basic
device_group
subsystem1
Type restrictions: string



Required connections
Outbounds
plelastic
Target Instance
ProductionBackend

Target Inbound
elastic

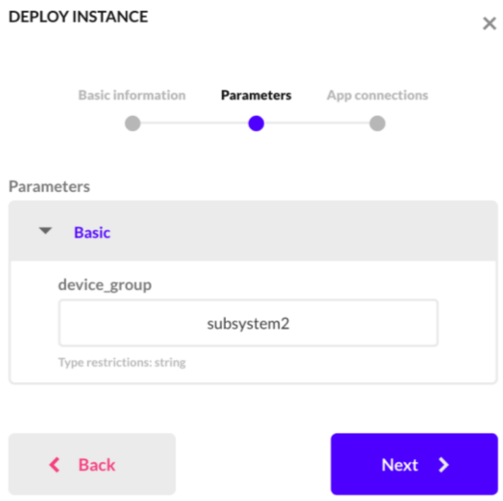
- LocalSystem2



DEPLOY INSTANCE

Registered name: DroneBackendLocal

Instance name*: LocalSystem2



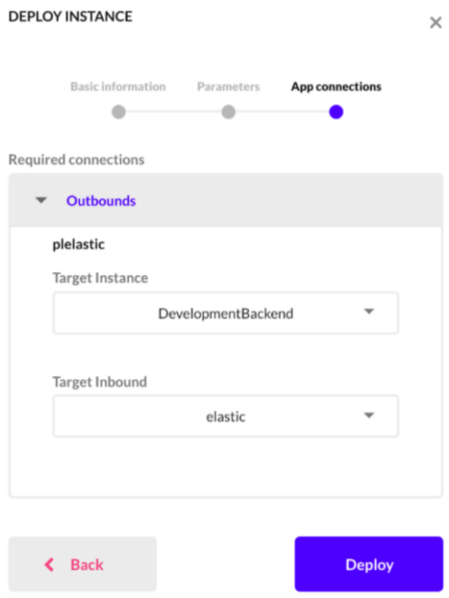
DEPLOY INSTANCE

Parameters

Basic

device_group: subsystem2

Type restrictions: string



DEPLOY INSTANCE

Required connections

Outbounds

plelastic

Target Instance: DevelopmentBackend

Target Inbound: elastic

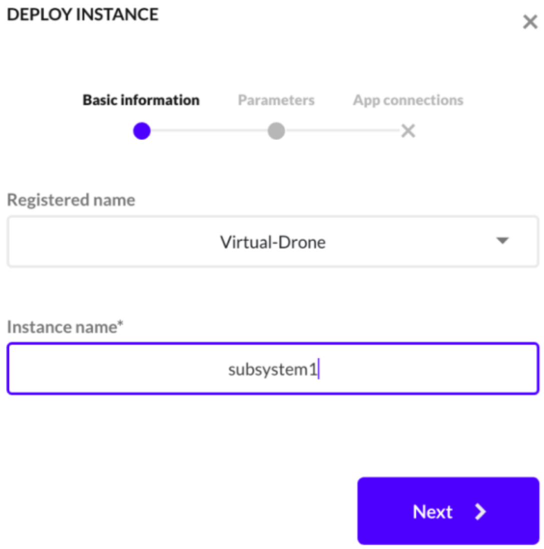
4. Deploying virtual drones

Now that you have the architecture in place, it’s time to launch a virtual drone for each Device Group. Please deploy two instances of the Virtual-Drone application, called subsystem1 and subsystem2.

- Subsystem1 is linked to the Device Group subsystem1, and thus uses its deviceGroupID and deviceGroupAPIKEY.
- Subsystem2 is, as you may have guessed, linked to the Device Group subsystem2, and uses the parameters that describe it.

Retrieve the file you saved when creating the Device Groups in step 1 of this tutorial and use the information in it to configure the Virtual-Drone instances appropriately. We provide some screenshots to illustrate where to use this information below.

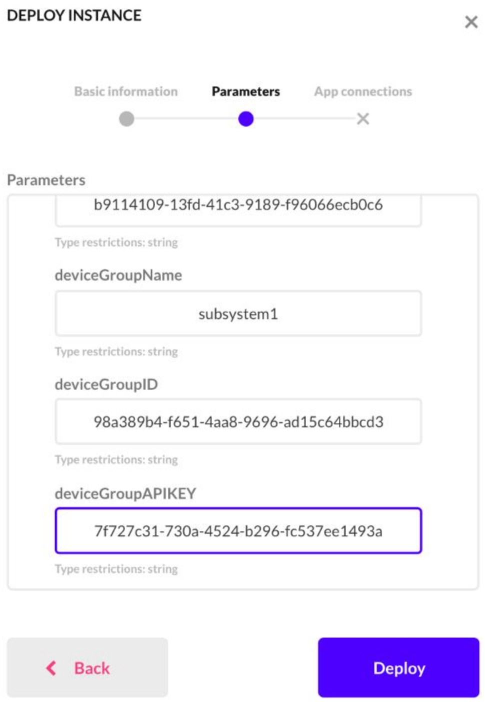
- Subsystem1



DEPLOY INSTANCE

Registered name: Virtual-Drone

Instance name*: subsystem1



DEPLOY INSTANCE

Parameters

b9114109-13fd-41c3-9189-f96066ecb0c6

Type restrictions: string

deviceGroupName: subsystem1

Type restrictions: string

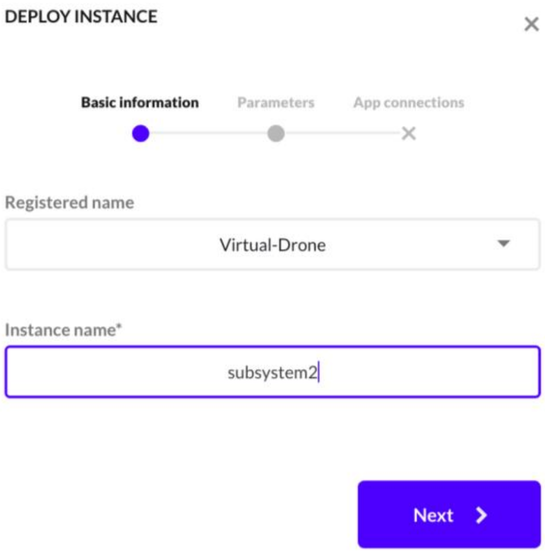
deviceGroupID: 98a389b4-f651-4aa8-9696-ad15c64bbcd3

Type restrictions: string

deviceGroupAPIKEY: 7f727c31-730a-4524-b296-fc537ee1493a

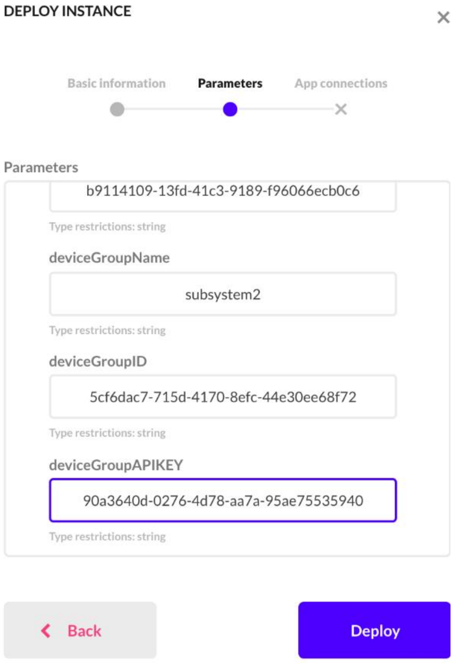
Type restrictions: string

- Subsystem2


DEPLOY INSTANCE

Registered name: Virtual-Drone

Instance name*: subsystem2


DEPLOY INSTANCE

Parameters

b9114109-13fd-41c3-9189-f96066ecb0c6

Type restrictions: string

deviceGroupName: subsystem2

Type restrictions: string

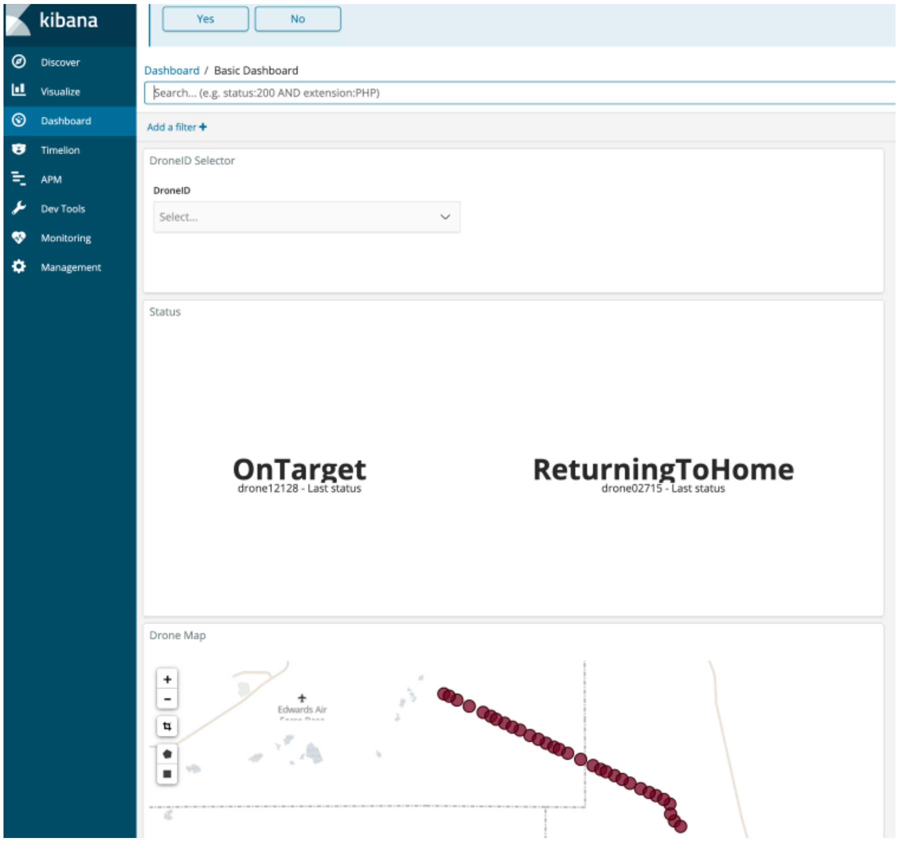
deviceGroupID: 5cf6dac7-715d-4170-8efc-44e30ee68f72

Type restrictions: string

deviceGroupAPIKEY: 90a3640d-0276-4d78-aa7a-95ae75535940

Type restrictions: string

Now the system is live, and you can check that the drones are sending the correct data in Kibana's Dashboard.



OnTarget
drone12128 - Last status

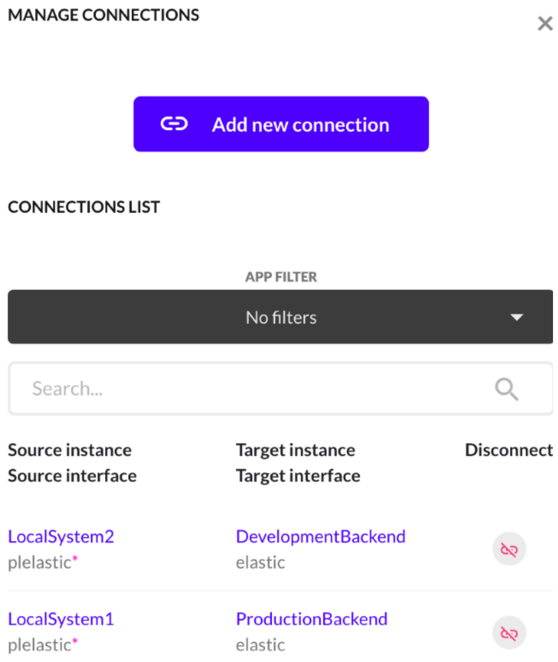
ReturningToHome
drone02715 - Last status

If you can’t see the path the drones are following in the map, click on the third icon, and the map will focus on the area that has data to show.

5. Change the network connections

Now you need to move what was in the development backend to the production backend. You can do that by removing the application network connection between LocalSystem2 and DevelopmentBackend, and then create a new one between LocalSystem2 and ProductionBackend.

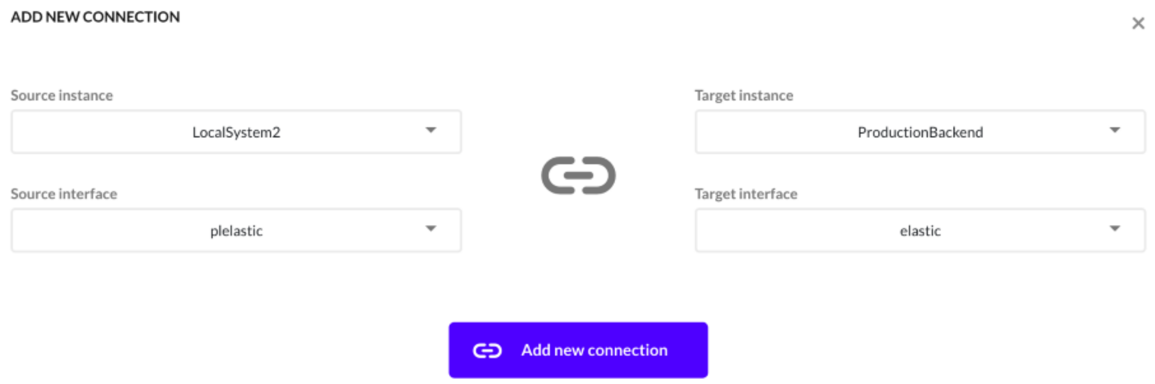
To remove the connection, click on the “Manage connections” button in the upper right corner of the Application view. Then, click Disconnect in the connection you don’t want anymore (LocalSystem2 – DevelopmentBackend).



APP FILTER: No filters

Source instance Source interface	Target instance Target interface	Disconnect
LocalSystem2 pLelastic*	DevelopmentBackend elastic	
LocalSystem1 pLelastic*	ProductionBackend elastic	

After that, you can create a new connection clicking on the instance to go to its detailed view, and adding a new connection there:



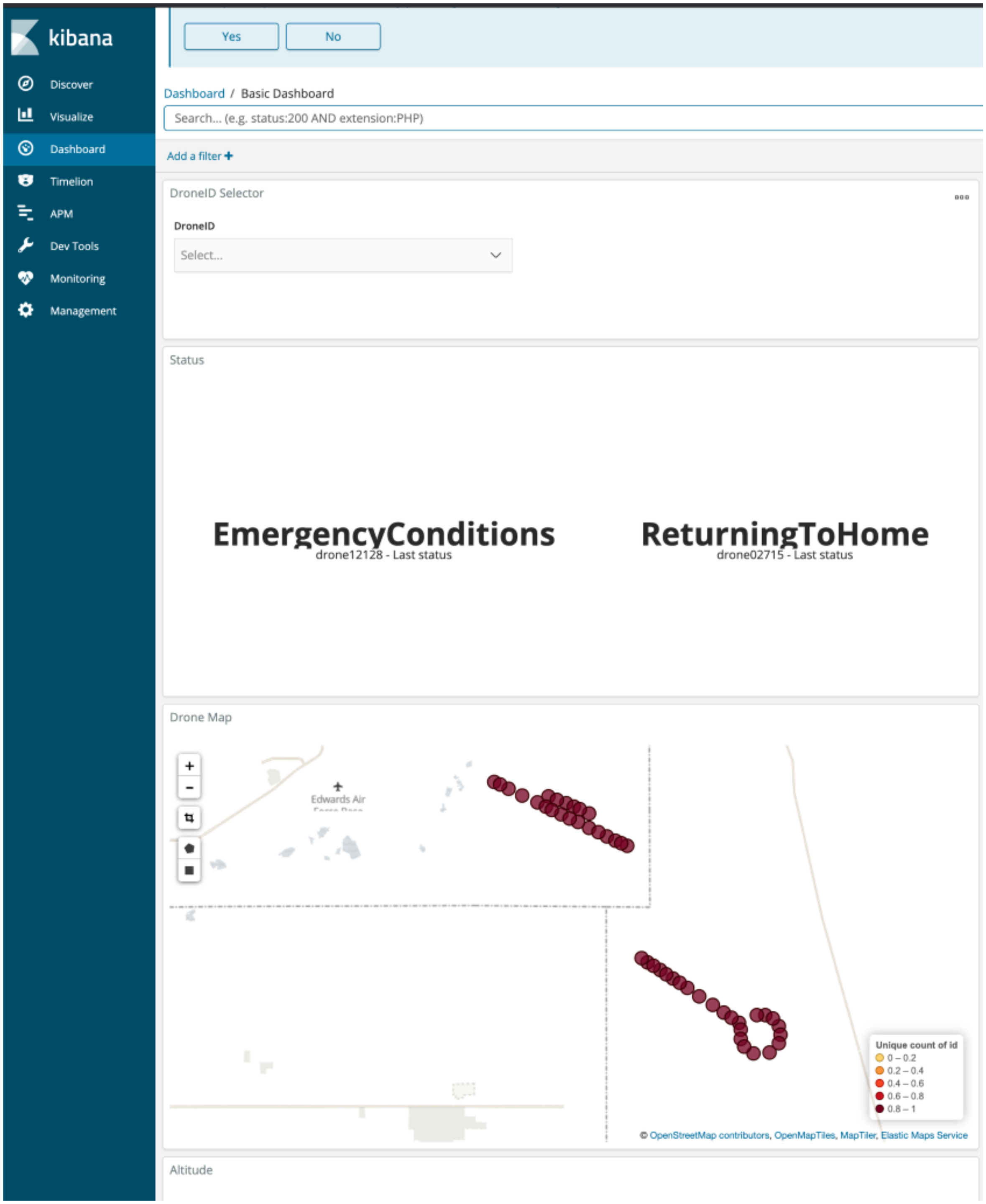
Source instance: LocalSystem2

Source interface: plelastic

Target instance: ProductionBackend

Target interface: elastic

And you're done! To check if everything is correctly configured, please go to Kibara's Dashboard and check that all the drones are sending data to the production backend.



Dashboard / Basic Dashboard

Search... (e.g. status:200 AND extension:PHP)

DroneID Selector

DroneID

Select...

Status

EmergencyConditions
drone12128 - Last status

ReturningToHome
drone02715 - Last status

Drone Map

Unique count of id
● 0 – 0.2
● 0.2 – 0.4
● 0.4 – 0.6
● 0.6 – 0.8
● 0.8 – 1

© OpenStreetMap contributors, OpenMapTiles, MapTiler, Elastic Maps Service

Altitude