

Step by step guide to migration AdBeacons to BalenaOS

V20200703.1

General sequence of process

All process is divided in three phases

- **PHASE 1: AdBeacon OS**

In this phase, two scripts are run on the original AdBeacon operating system. The scripts are the **diagnostic** and, the one that does the **download of the images of BALENAOS and MIGOS** once all the downloads have been made, proceeds to **install MIGOS** in the `/boot` partition. Once MIGOS is installed, the computer restart and phase 2 begins.

- **PHASE 2: MIGOS**

In the second phase, when the computer restart automatically **loads MIGOS in RAM**, it copies the images downloaded from BALENA OS to RAM as well and starts the whole process of **installing and configuring BALENA OS** in the SD of the device. Once this process finishes, the computer restart again.

- **PHASE 3: BALENA OS**

In this last phase, the computer restarts with the new operating system, it automatically registers within the Balena cloud. Once registered, the user's own variables are registered, provisioning is made and finally the device is moved to the application within the Balena cloud.

Prerequisites

There are two ways to do the entire migration process, one manual and the other automated, in each phase the two methods are explained.

For the **automated** process is necessary:

- An Ubuntu Linux 19.04 operating system is recommended, although in theory it also works on Windows and MacOS.
- Have the pusher client installed: https://pusher.com/docs/channels/pusher_cli/overview
- Have the balena client installed: <https://www.balena.io/docs/reference/balena-cli/>

For the **manual** process is necessary:

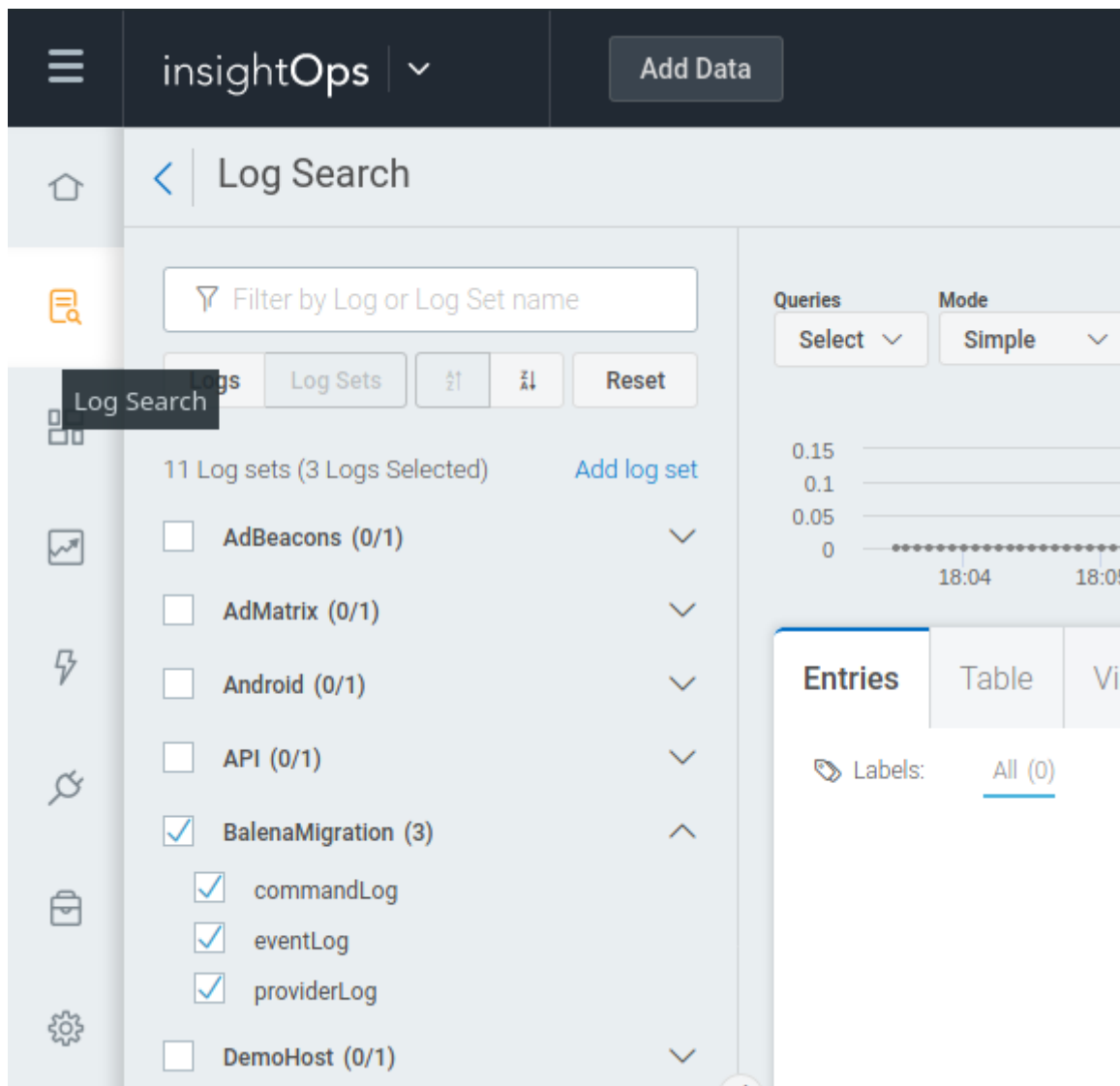
- Have access to pusher dashboard in `admobilize-production`: <https://dashboard.pusher.com/apps/367382>
- Have access to balena dashboard in `BalenaMigration` App: <https://dashboard.balena-cloud.com/apps/1547294>

For **both** process is necessary:

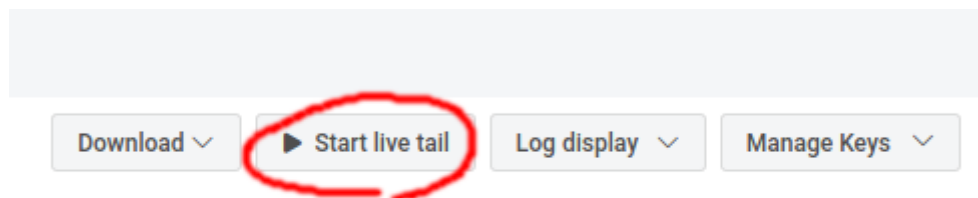
- Have access to `insightOps` dashboard: <https://insight.rapid7.com>
- Have the **ID** of the device to migrate, for example: `b8_27_eb_a0_a8_71` (with this device ID will be done all the examples in this document)

How to see the Logs of the migration process

- Once you have accessed the `insightOps` dashboard in the `Log Search` section, select only the items referring to `BalenaMigration`



- To make the visualization more fluid it is recommended to select the option `Start live tail`



Migration phase 1

Automated way

- Download the script `migPusher.sh` from: <https://storage.cloud.google.com/balenamigration/migscripts/migPusher.sh>

- Open one terminal, inside the directory where the previous script was downloaded, subscribe to the Pusher messages with the `device ID` to migrate, like this:

```
./migPusher.sh cli b8_27_eb_a0_a8_71 subscribe
```

- In another terminal exec:

```
./migPusher.sh cli b8_27_eb_a0_a8_71 Diagnostic
```

This will remotely run the **diagnostic script** on the specified device. The short result of this script can be seen in the console where the `subscribe` script is running.

Additionally in the dashboard of `insightOps` you can view the logs of the entire process.

If the result is **DIAGNOSTIC SUCCESS** you can proceed to execute the next script, if not, check the logs to see what was the error.

- In the same terminal where the diagnostic was run, execute the command for the installation script

```
./migPusher.sh cli b8_27_eb_a0_a8_71 InstallMIGOS
```

As in the previous script, you can see the result of the script in the console where `subscribe` was executed and the log of the entire process in `insightOps`.

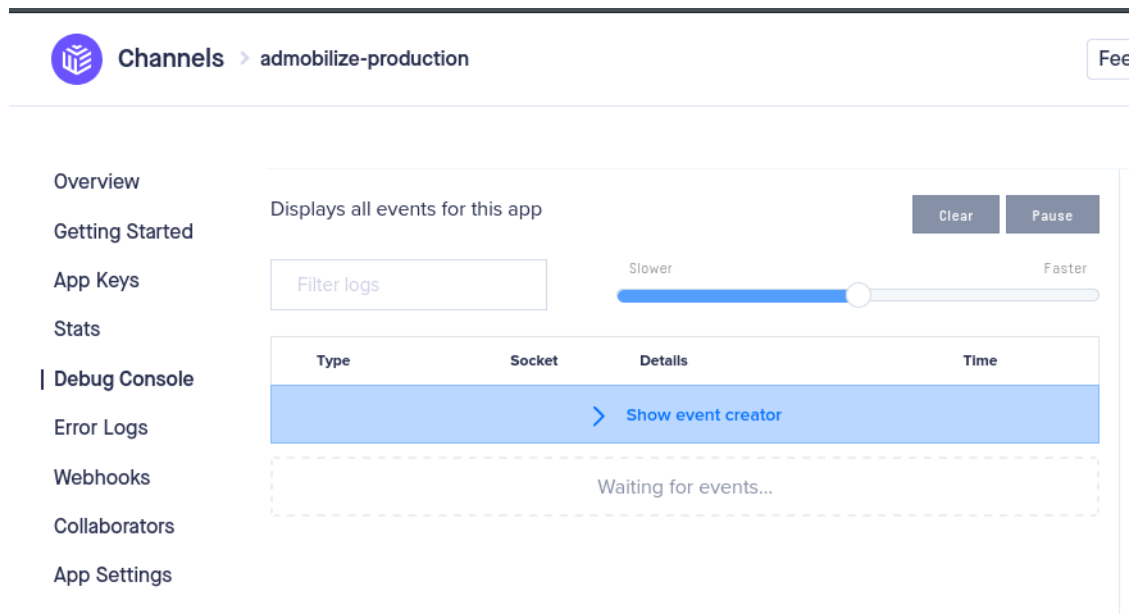
If the result is successful, you will see **INSTALL MIGOS SUCCESS**

- Once the installation is successful, the device is restarted with:

```
./migPusher.sh cli b8_27_eb_a0_a8_71 reboot
```

Manual way

- Log in to pusher dashboard into the `channel` of `admobilize-production` and to the `Debug Console` section



The screenshot shows the Pusher Channels dashboard for the channel 'admobilize-production'. The 'Debug Console' section is active, displaying a table with columns: Type, Socket, Details, and Time. The table is currently empty, showing a 'Waiting for events...' message. A 'Show event creator' button is visible in the Details column. The dashboard also includes a sidebar with navigation links: Overview, Getting Started, App Keys, Stats, Debug Console (selected), Error Logs, Webhooks, Collaborators, and App Settings. At the top right, there are 'Clear' and 'Pause' buttons.

- Click in `Show event creator` and put the next info:

In `Channel` the id of device: `b8_27_eb_a0_a8_71`

In `Event` the word `request`

In `Data` clear all and put this:

```
{
  "command":"cd /tmp && wget
https://storage.googleapis.com/balenamigration/migscripts/migDiagnostic.sh -
O migDiagnostic.sh && wget
https://storage.googleapis.com/balenamigration/migscripts/migDiagnostic.sh.m
d5 -O migDiagnostic.sh.md5 && md5sum --check migDiagnostic.sh.md5 && bash
migDiagnostic.sh"
}
```

Click on **Send event**.

The result of the command can be seen by looking for the response within the messages received on the same page taking into account the **device ID** of the device.

Also, in the dashboard of **insightOps** you can also see the log of the entire process.

If everything goes well, you should read the message **SUCCESS OF DIAGNOSIS**.

- To execute the installation script, click again in **Show event creator** and put the next info:

In **Channel** the device ID: **b8_27_eb_a0_a8_71**

In **Event** the word **request**

In **Data** clear all and put this:

```
{
  "command":"cd /tmp && wget
https://storage.googleapis.com/balenamigration/migscripts/migInstallMIGOS.sh
-O migInstallMIGOS.sh && wget
https://storage.googleapis.com/balenamigration/migscripts/migInstallMIGOS.sh
.md5 -O migInstallMIGOS.sh.md5 && md5sum --check migInstallMIGOS.sh.md5 &&
bash migInstallMIGOS.sh"
}
```

Click on **Send event**.

The result of the command can be seen, looking for the answer within the messages received on the same page, taking into account the device's device ID.

Additionally in the dashboard of **insightOps** you can also see the log of the whole process.

If everything goes well, you should see the message **INSTALL MIGOS SUCCESS**.

- Once the installation is successful, the device should be restarted. To do this, click again on **Show event creator** and enter the following information:

In **Channel** the device ID: **b8_27_eb_a0_a8_71**

In **Event** the word **request**

In **Data** clear all and put this:

```
{
  "command":"[ ! -f /root/migstate/MIG_DIAGNOSTIC_IS_RUNING ] && [ ! -f
/root/migstate/MIG_INSTALL_MIGOS_IS_RUNING ] && [ ! -f
/root/migstate/MIG_RESTORE_RASPB_BOOT_IS_RUNING ] && reboot"
}
```

Click in **Send event**.

Migration phase 2

This phase is fully automated, so there is no need to do any kind of intervention, just follow the entire process by reviewing the logs that the device generates in the `insightOps` dashboard.

In the logs of the process, **MIGOS SUCCESS SUPERVISOR** should be seen at the end, which indicates that the entire migration process of phase two concluded successfully. If all goes well, the computer will automatically restart and start phase 3.

Migration phase 3

Automated way

- For this phase it is necessary to download two files, one contains the information of the migrated devices `devices_migrated.csv` and the other is the script that configures those devices in the balena cloud `migProvider.sh`

https://storage.cloud.google.com/balenamigration/migscripts/devices_migrated.csv

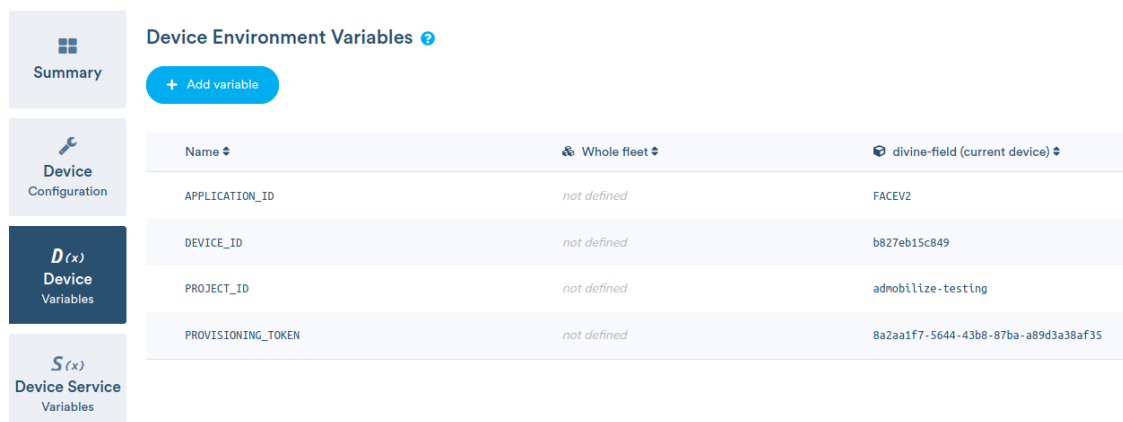
<https://storage.cloud.google.com/balenamigration/migscripts/migProvider.sh>

- Once downloaded, the script runs without any argument, it connects to the balena cloud and scans for the migrated devices, once found, it will configure the new device automatically and move it to the final destination application.

```
./migProvider.sh
```

Manual way

- Log in to balena dashboard in the app `BalenaMigration`: <https://dashboard.balena-cloud.com/apps/1547294>
- Once the migrated device appears, select it and proceed to the section `Device Variables`



Device Environment Variables ⓘ		
+ Add variable		
Name ↕	Whole fleet ↕	divine-field (current device) ↕
APPLICATION_ID	not defined	FACEV2
DEVICE_ID	not defined	b827eb15c849
PROJECT_ID	not defined	admobilize-testing
PROVISIONING_TOKEN	not defined	8a2aa1f7-5644-43b8-87ba-a89d3a38af35

- Here, the following 4 variables will be entered:

`APPLICATION_ID` will have the value of `FACEV2`

`PROJECT_ID` the value of `admobilize-testing`

`DEVICE_ID` the same one that has been used but without underscores `b827eba0a871`

And for the variable `PROVISIONING_TOKEN` search inside the file `devices_migrated.csv` the `deviceId` and its corresponding `deviceProvisioningToken`

- Once the variables have been entered, the device is moved to its final application. The rest of the provisioning and updating process will be done automatically.

↺ Reboot

↺ Restart



TYPE



Raspberry

SUPERVISOR VER

10.3.7

IP ADDRESS

10.0.0.107



PUBLIC DEVICE URL



Move device

Pin to release

Enable Lock override

Enable local mode

Grant support access

Manage tags

Purge data

Shutdown

Delete