# FoundriesFactory® Documentation

**v1.0.1**

# Table of contents:

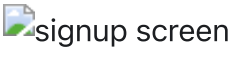
Version: 0.1.1

# Getting Started

Yocto Version: kirkstone

## Signing Up

To begin using FoundriesFactory®, start with creating an account with us.


signup screen
This is the beginning of your journey.

## Creating Your Factory

FoundriesFactory is the start of your embedded OS, tailored specifically for your product. When you create a Factory, we immediately bootstrap the CI build process. This generates an unmodified Linux microPlatform OS Image, which is from this point onward, **owned by you**.

When your account is created, it is not associated with any factories. Create one by clicking `Create Factory`.

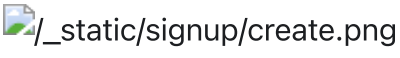

/_static/signup/no-factories.png
Your journey begins empty handed

## Selecting Your Platform

Choose a hardware platform from the dropdown menu in the **Create Factory** wizard and continue. Click `Create Factory` once your details are entered.

> **DANGER**
>
> Once a Factory is created, the chosen platform/machine and Factory name cannot be changed. Create a new Factory or contact support if a mistake is made.

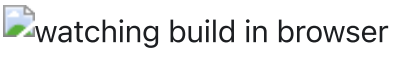
The Linux MicroPlatform supports a wide range of platforms out of the box. This includes QEMU images for ARM and RISC-V architectures.


Create Factory

> **TIP**
>
> Your chosen platform determines the value for the `machines:` key for your builds.

# Watching Your Build

Once you have created your Factory, the initial artifacts from the Foundries.io™ Linux® microPlatform (LmP) will be generated. This is the base to build your product. You can monitor the progress in the `Targets` tab of your Factory. Additionally, you will receive an email once the Factory initial setup is complete.


FoundriesFactory Targets

Targets are a reference to a platform image and Docker applications. When developers push code, FoundriesFactory produces a new target. Registered devices then update and install Targets.

> **NOTE**
>
> If you would like to learn more, we wrote a blog about what Targets are and why we made them the way they are.

The `Targets` tab of the Factory will become more useful as you begin to build your application and produce new Targets for the Factory to build.

**Tags:** signup getting started

Version: 0.1.1

# Flashing Your Device

> **NOTE**
>
> The initial FoundriesFactory® set up and build is finished very quick. Follow its status with steps listed in Watching Your Build.

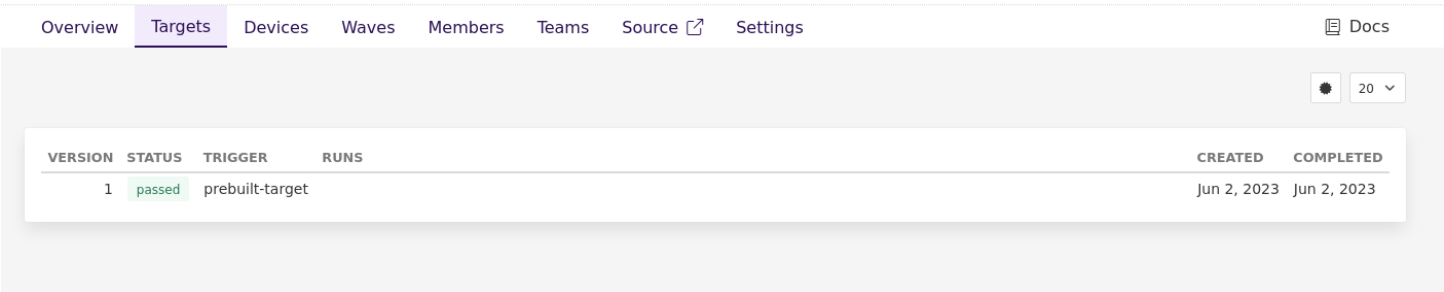
## Prerequisites and Pre-Work

- A supported board which is either:
  - Capable of booting from eMMC, **supported by default if available**
  - **Or** capable of booting from a suitable microSD Card
- Wired or WiFi network with internet access.
  - Ethernet cable (if choosing Wired)
  - Console access to your hardware via UART serial (if choosing WiFi)

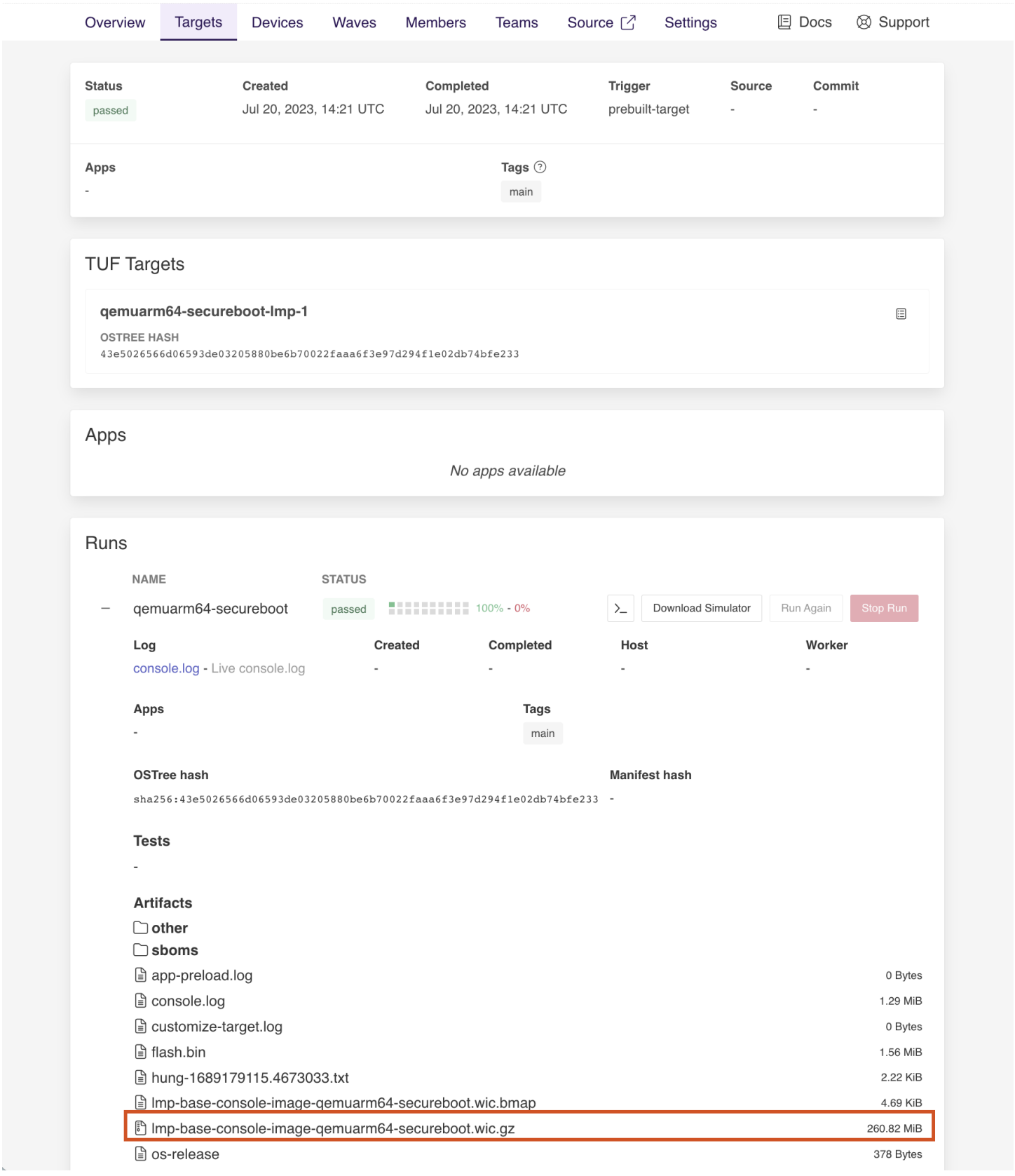
## Downloading the LmP System Image

After a successful build, FoundriesFactory produces build artifacts which can be downloaded from the `Targets` tab of your Factory.

1. Navigate to the `Targets` section of your Factory.
2. Click the Target with the `prebuilt-target` `Trigger`.




3. Expand the `Targets` tab clicking on it. This shows a link to the Factory image artifact. Download the Factory image for your machine, e.g: `lmp-base-console-image-<machine_name>.wic.gz`:


Overview
Targets
Devices
Waves
Members
Teams
Source
Settings
Docs
Support
Status
passed
Created
Jul 20, 2023, 14:21 UTC
Completed
Jul 20, 2023, 14:21 UTC
Trigger
prebuilt-target
Source
-
Commit
-
Apps
-
Tags
main
TUF Targets
qemuarm64-secureboot-lmp-1
OSTREE HASH
43e5026566d06593de03205880be6b70022faaa6f3e97d294f1e02db74bfe233
Apps
No apps available
Runs
NAME
STATUS
qemuarm64-secureboot
passed
100% - 0%
Download Simulator
Run Again
Stop Run
Log
console.log - Live console.log
Created
-
Completed
-
Host
-
Worker
-
Apps
-
Tags
main
OSTree hash
sha256:43e5026566d06593de03205880be6b70022faaa6f3e97d294f1e02db74bfe233
Manifest hash
-
Tests
-
Artifacts
other
sboms
app-preload.log
0 Bytes
console.log
1.29 MiB
customize-target.log
0 Bytes
flash.bin
1.56 MiB
hung-1689179115.4673033.txt
2.22 KiB
lmp-base-console-image-qemuarm64-secureboot.wic.bmap
4.69 KiB
lmp-base-console-image-qemuarm64-secureboot.wic.gz
260.82 MiB
os-release
378 Bytes

NOTE

> Most platforms require more than the `lmp-factory-image-<machine_name>.wic.gz` artifact for flashing. The required artifacts are board specific and listed in respective pages under. Targets publish all needed files for each platform under `Runs`.

# Flashing the Image

The flashing procedure is board specific and we cover separate steps in `ref-boards`{.interpreted-text role="ref"}. Please refer to this section for specifics on flashing your system image using the vendor provided tools. See `ref-qemu`{.interpreted-text role="ref"} for booting Qemu images.

> **NOTE**
>
> LmP enforces eMMC boot whenever possible as this is the path to enable all security features it provides. So for platforms with available eMMC, such as the NXP® i.MX EVKs, booting from eMMC rather than SD is highly recommended and enabled by default.

# Booting and Connecting to the Network

After flashing and booting the board with the respective steps for your hardware, follow these steps to connect to the network.

> **NOTE**
>
> By default, the `username` and `password` to log in your device after boot are `fio`/`fio`. We recommend changing them once you are in development.

**Ethernet (Recommended)** | **Wifi**

Ethernet works out of the box if a DHCP server is available on the local network. Connect an Ethernet cable to the board. Your board will connect to the network via Ethernet soon after booting.

## Logging in via SSH

To login via SSH, run:

```
ssh fio@<machine-name>.local
```

Where `fio` is the username and `<machine-name>` is the hostname of your device. The default password is `fio`.

By default, your device hostname is set to a unique string that specify the platform chosen during Factory creation (`machine`). Check `ref-linux-supported`{.interpreted-text role="ref"} for a list of supported platform and their `machine` values.

::: tip

Here are some examples of default hostnames:

| `raspberrypi4-64.local` | `intel-corei7-64.local` | `imx8mm-lpddr4-evk.local` :::

::: note

For this to work, your PC needs to support zeroconf. The hostname must be unclaimed.

If this does not work, see `Troubleshooting <gs-troubleshooting>`{.interpreted-text role="ref"} below for advice. :::

**Troubleshooting**

If the above methods to SSH into your board do not work, there are additional things to try.

1. Get the IP address of your device:

    - Temporarily enable and connect to the UART serial (detailed steps for some platforms can be found in `ref-board` and determine available IP addresses with:

    - Ethernet:

      ```
      ip addr show eth0 scope global
      ```

    - WiFi:

      ```
      ip addr show wlan0 scope global
      ```

- **Or** list the connected devices and their local IP addresses on your network router's administrative interface.

2. Connect to the device by IP address:

```
ssh fio@<ip-address>
```

**Tags:** getting started flashing prebuilt-target

Version: 0.1.1

# Installing Fioctl

`Fioctl™ <ref-fioctl>`{.interpreted-text role="ref"} is a simple tool for interacting with the Foundries.io REST API.

::: seealso Fioctl is based on Foundries.io\'s ota-lite API. :::

`ref-fioctl`{.interpreted-text role="ref"} is used to manage:

- `Tags (per device, per device group and per Factory) <ref-advanced-tagging>`{.interpreted-text role="ref"}
- `Device Configuration <ref-fioconfig>`{.interpreted-text role="ref"}
- `OTA Updates <ref-aktualizr-lite>`{.interpreted-text role="ref"}
- `CI Secrets <ref-container-secrets>`{.interpreted-text role="ref"}
- `Offline TUF Keys <ref-offline-keys>`{.interpreted-text role="ref"}
- `Git Access <gs-git-config>`{.interpreted-text role="ref"}

# Installation

## Manual Installation

We use Github Releases to distribute static golang binaries.

::: tip ::: title Tip :::

Repeating the following steps will overwrite an existing binary, useful for updating or changing version. :::

::: tabs ::: group-tab Linux

::: attention ::: title Attention :::

Make sure you have `curl` installed. :::

1. Download a Linux binary to a directory on your `PATH`.

   For example, to download version on Linux, define the version:

::: parsed-literal FIOCTL_VERSION=\"\" :::

Download the binary with curl:

::: prompt bash host:~\$, auto

host:~\$ sudo curl -o /usr/local/bin/fioctl -LO https://github.com/foundriesio/fioctl/releases/download/$FIOCTL_VERSION/fioctl-linux-amd64 :::

2. Make the `ref-fioctl`{.interpreted-text role="ref"} binary executable:

```
::: prompt
bash host:\~\$, auto

host:\~\$ sudo chmod +x /usr/local/bin/fioctl
:::
```

:::

::: group-tab macOS

::: attention ::: title Attention :::

Make sure you have `curl` installed. :::

1. Download a Darwin binary from the Github Releases page to a directory on your `PATH`.

   For example, to download version on macOS, define the version:

   ::: parsed-literal FIOCTL_VERSION=\"\" :::

   Download the binary with curl:

   ::: prompt bash host:~\$, auto

   host:~\$ sudo curl -o /usr/local/bin/fioctl -L https://github.com/foundriesio/fioctl/releases/download/$FIOCTL_VERSION/fioctl-darwin-amd64 :::

   ::: important ::: title Important :::

   For MacOS running on a Apple M1 processor, replace `fioctl-darwin-amd64` with `fioctl-darwin-arm64`, and set `FIOCTL_VERSION` to v0.21 or newer. :::

2. Make the `ref-fioctl`{.interpreted-text role="ref"} binary executable:

   ```
   ::: prompt
   bash host:\~\$, auto

   host:\~\$ sudo chmod +x /usr/local/bin/fioctl
   :::
   ```

   :::

::: group-tab Windows

1. Download a Windows binary from the Github Releases page.
2. Put it in a folder of your choosing and rename it to `fioctl.exe`
3. Press `Win + R` and type `SystemPropertiesAdvanced`
4. Press `enter` or click `OK`.
5. Click \"Environment Variables...\" in the resultant menu..
6. Click the `Path` **system** variable, then click `Edit...`
7. Click `New` in the \"Edit environment variable\" menu.
8. Enter the path to the folder in which you have placed `ref-fioctl`{.interpreted-text role="ref"}.

   An example path string if installing to a folder on the desktop would look like this.

   `C:\Users\Gavin\Desktop\fio\bin`

You should now be able to open `cmd.exe` or `powershell.exe` and type `fioctl`. ::: :::

# Authenticating Fioctl

With `ref-fioctl`{.interpreted-text role="ref"} installed, authenticate it with our backend. For this, you will generate OAuth2 application credentials for interacting with the FoundriesFactory API:

::: prompt bash host:~\$, auto

host:~\$ fioctl login

: Please visit:

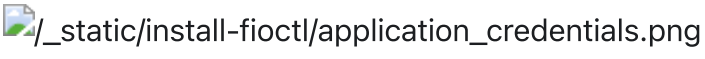
```
<https://app.foundries.io/settings/credentials/>

and create a new \"Application Credential\" to provide inputs below.

Client ID:
```

:::

`ref-fioctl`{.interpreted-text role="ref"} will now ask for your Client ID and Secret. Follow the next steps to generate them.

## Adding Application Credentials

Go to Application Credentials and click on `+ New Credentials`{.interpreted-text role="guilabel"}.


/_static/install-fioctl/application_credentials.png
Application Credentials

Complete with a **Description** and the **Expiration date** and select `next`{.interpreted-text role="guilabel"}.

For Fioctl®, check the `Use for tools like fioctl`{.interpreted-text role="guilabel"} box and select your **Factory**. You can revoke this access and set up a new credential later once you are familiar with the `ref-api-access`{.interpreted-text role="ref"}.


/_static/install-fioctl/fioctl_token.png
API Token

::: tip ::: title Tip :::

We recommend creating a new API token for each computer you plan to use our tools with. For example, if you intend to develop on both a laptop and a desktop, create a new token for each, as you would with SSH keys. This way you can revoke tokens for individual systems, should they be compromised. :::

Use the Client ID and Secret to finish the Fioctl login.


/_static/install-fioctl/token.png
Client ID and Secret

::: prompt bash host:~\$, auto

host:~\$ fioctl login

: Please visit:

```
<https://app.foundries.io/settings/credentials/>

and create a new \"Application Credential\" to provide inputs below.

Client ID: Client secret: You are now logged in to Foundries.io
services.
```

:::

::: seealso `ref-fioctl`{.interpreted-text role="ref"} documentation. :::

# Configuring Git

After `Fioctl <ref-fioctl>`{.interpreted-text role="ref"} is properly setup, it can be leveraged as a Git credential helper to allow pushing to your repositories with `FoundriesFactory® <ref-factory-definition>`{.interpreted-text role="ref"}. With this, Git knows when you connect to `source.foundries.io` and uses Fioctl for authentication when utilizing `git` commands.

## Setting Up Git

Run the following command to add the relevant entries to the Git configuration:

::: prompt bash host:~\$, auto

host:~\$ sudo fioctl configure-git :::

::: important ::: title Important :::

This must run as `sudo` instead of directly as the `root` user. This is because it needs to have privileges to create a symlink in the same directory as where `git` is located. :::

::: warning ::: title Warning :::

* If for some reason the command fails with an error, the following manual steps can be taken to get the exact same result:

git config --global credential.https://source.foundries.io.username fio-oauth2 git config --global credential.https://source.foundries.io.helper fio ln -s /usr/bin/fioctl /usr/bin/git-credential-fio

- Existing users reconfiguring Git access may need to remove the following lines from `.gitconfig` to use `fioctl configure-git` utility:

[http \"https://source.foundries.io\"] extraheader = Authorization: basic \<TOKEN>

- If editing scopes on existing tokens, the user should refresh the local `fioctl` credentials with:

fioctl login --refresh-access-token :::

Verify this has succeeded by cloning a repository from your Factory, such as your `containers.git` repo. Replace `<factory>` with your Factory\'s name:

::: prompt bash host:~\$, auto

host:~\$ git clone https://source.foundries.io/factories/\<factory>/containers.git :::

::: tip ::: title Tip :::

You can also use `git config --global --list` to show the current state of the global Git configuration, where `source.foundries.io` should be referenced along with a username and a helper. :::

::: seealso * `Fioctl Reference Manual <ref-fioctl>`{.interpreted-text role="ref"} * `API Access for factory <ref-api-access>`{.interpreted-text role="ref"} :::

Version: 0.1.1

# fioctl

## fioctl

Manage Foundries Factories

### Options

```
-c, --config string   config file (default is $HOME/.config/fioctl.yaml)
-h, --help            help for fioctl
-v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl completion - Generate completion script
- fioctl config - Manage configuration common to all devices in a factory
- fioctl configure-docker - Configure a hub.foundries.io Docker credential helper
- fioctl configure-git - Configure a source.foundries.io Git credential helper
- fioctl devices - Manage devices registered to a factory
- fioctl el2g - Manage EdgeLock 2Go integration
- fioctl event-queues - Manage event queues configured for a Factory
- fioctl keys - Manage keys in use by your factory fleet
- fioctl login - Access Foundries.io services with your client credentials
- fioctl logout - Remove Foundries.io client credentials from system
- fioctl secrets - Manage secret crendentials configured in a factory
- fioctl status - Get dashboard view of a factory and its devices
- fioctl targets - Manage factory's TUF targets
- fioctl teams - List teams belonging to a FoundriesFactory
- fioctl users - List users with access to a FoundriesFactory
- fioctl version - Show version information of this tool.
- fioctl waves - Manage factory's waves

Version: 0.1.1

# fioctl_completion

## fioctl completion

Generate completion script

```
fioctl completion [bash|zsh|powershell]
```

### Examples

```
# Bash:
$ source <(fioctl completion bash)

# To load completions for each session, execute once:
Linux:
  $ fioctl completion bash > /etc/bash_completion.d/fioctl
MacOS:
  $ fioctl completion bash > /usr/local/etc/bash_completion.d/fioctl

# Zsh:
# If shell completion is not already enabled in your environment you will need
# to enable it.  You can execute the following once:

$ echo "autoload -U compinit; compinit" >> ~/.zshrc

# To load completions for each session, execute once:
$ fioctl completion zsh > "${fpath[1]}/_fioctl"

# You will need to start a new shell for this setup to take effect.

# Fish:
$ fioctl completion fish > ~/.config/fish/completions/fioctl.fish
```

### Options

```
  -h, --help   help for completion
```

## Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

## SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_config

## fioctl config

Manage configuration common to all devices in a factory

### Options

```
-f, --factory string   Factory to list targets for
-h, --help             help for config
-t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
-c, --config string   config file (default is $HOME/.config/fioctl.yaml)
-v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories
- fioctl config delete - Delete file from the current configuration
- fioctl config device-group - Manage factory device groups
- fioctl config log - Show a changelog of configuration
- fioctl config rotate-certs - Rotate device x509 keypairs in this group used to connect to the device gateway
- fioctl config set - Create a new factory-wide configuration
- fioctl config updates - Configure aktualizr-lite settings for how updates are applied to a device group
- fioctl config wireguard - Show current wireguard server config for factory

Version: 0.1.1

# fioctl_config_delete

## fioctl config delete

Delete file from the current configuration

```
fioctl config delete <file> [flags]
```

### Options

```
  -g, --group string   Device group to use
  -h, --help           help for delete
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl config - Manage configuration common to all devices in a factory

Version: 0.1.1

# fioctl_config_device-group

## fioctl config device-group

Manage factory device groups

### Options

```
  -h, --help   help for device-group
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl config - Manage configuration common to all devices in a factory
- fioctl config device-group create - Create a new device groups
- fioctl config device-group delete - Delete an existing device group
- fioctl config device-group list - Show available device groups
- fioctl config device-group update - Rename an existing device group

Version: 0.1.1

# fioctl_config_device-group_create

## fioctl config device-group create

Create a new device groups

```
fioctl config device-group create <name> [<description>] [flags]
```

### Options

```
  -h, --help   help for create
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl config device-group - Manage factory device groups

Version: 0.1.1

# fioctl_config_device-group_delete

## fioctl config device-group delete

Delete an existing device group

```
fioctl config device-group delete <name> [flags]
```

### Options

```
  -h, --help   help for delete
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl config device-group - Manage factory device groups

Version: 0.1.1

# fioctl_config_device-group_list

## fioctl config device-group list

Show available device groups

```
fioctl config device-group list [flags]
```

### Options

```
  -h, --help   help for list
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl config device-group - Manage factory device groups

Version: 0.1.1

# fioctl_config_device-group_update

## fioctl config device-group update

Rename an existing device group

```
fioctl config device-group update <name> [flags]
```

### Options

```
  -d, --description string   Change a device group description
  -h, --help                 help for update
  -n, --name string          Change a device group name
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl config device-group - Manage factory device groups

Version: 0.1.1

# fioctl_config_log

## fioctl config log

Show a changelog of configuration

```
fioctl config log [flags]
```

### Options

```
  -g, --group string   Device group to use
  -h, --help           help for log
  -n, --limit int      Limit the number of results displayed
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl config - Manage configuration common to all devices in a factory

Version: 0.1.1

# fioctl_config_rotate-certs

## fioctl config rotate-certs

Rotate device x509 keypairs in this group used to connect to the device gateway

### Synopsis

This command will send a fioconfig change to a device to instruct it to perform a certificate rotation using the EST server configured with "fioctl keys est".

This command will only work for devices running LmP version 90 and later.

```
fioctl config rotate-certs [flags]
```

### Options

```
      --dryrun                Show what the fioconfig entry will be and exit
  -p, --est-port int          The EST server port (default 8443)
  -e, --est-resource string   The path the to EST resource on your server (default
"/.well-known/est")
  -g, --group string          Device group to use
  -h, --help                  help for rotate-certs
      --hsm-cert-ids string   Available PKCS11 slot IDs for the client certificate
(default "03,09")
      --hsm-pkey-ids string   Available PKCS11 slot IDs for the private key
(default "01,07")
  -r, --reason string         The reason for changing the cert
      --server-name string    EST server name when not using the Foundries managed
server. e.g. est.example.com
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
```

```
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl config - Manage configuration common to all devices in a factory

Version: 0.1.1

# fioctl_config_set

## fioctl config set

Create a new factory-wide configuration

### Synopsis

Creates a factory wide configuration. The fioconfig daemon running on each device will then be able to grab the latest version of the configuration and the device's configuration and apply it. Use the --group parameter to create a device group wide configuration instead.

```
fioctl config set file=content <file2=content ...> [flags]
```

### Examples

```
  # Basic use
  fioctl config set npmtok="root" githubtok="1234" readme.md==./readme.md

  There are several ways how to pass a file content into this command:
  - with filename="filecontent" format, a file content is passed directly.
  - with filename==/path/to/file format, a file content is read from a specified
file path.

  # The configuration format also allows specifying what command to
  # run after a configuration file is updated on the device. To take
  # advantage of this, the "--raw" flag must be used.
  cat >tmp.json <<EOF
  {
    "reason": "I want to use the on-changed attribute",
    "files": [
      {
        "name": "npmtok",
        "value": "root",
        "on-changed": ["/usr/bin/touch", "/tmp/npmtok-changed"]
      },
```

```
      {
        "name": "A-Readable-Value",
        "value": "This won't be encrypted and will be visible from the API",
        "unencrypted": true
      },
      {
        "name": "githubtok",
        "value": "1234"
      }
    ]
  }
  > EOF
  fioctl config set --raw ./tmp.json

  # fioctl will read in tmp.json and upload it to the OTA server.
  # Instead of using ./tmp.json, the command can take a "-" and will read the
  # content from STDIN instead of a file.
```

## Options

```
      --create          Replace the whole config with these values. Default is to
merge these values in with the existing config values
  -g, --group string    Device group to use
  -h, --help            help for set
      --raw             Use raw configuration file
  -m, --reason string   Add a message to store as the "reason" for this change
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl config - Manage configuration common to all devices in a factory

Version: 0.1.1

# fioctl_config_updates

## fioctl config updates

Configure aktualizr-lite settings for how updates are applied to a device group

### Synopsis

View or change configuration parameters used by aktualizr-lite for updating devices in a device group. When run with no options, this command will print out how the group is currently configured.

```
fioctl config updates [flags]
```

### Examples

```
  # Make devices start taking updates from Targets tagged with "devel":
  fioctl config updates --group beta --tag devel

  # Set the docker apps that devices will run:
  fioctl config updates --group beta --apps shellhttpd

  # Set the docker apps and the tag for devices:
  fioctl config updates --group beta --apps shellhttpd --tag master
```

### Options

```
      --apps string    comma,separate,list
      --dryrun         Only show what would be changed
      --force          DANGER: For a config on a device that might result in
corruption
  -g, --group string   Device group to use
  -h, --help           help for updates
      --tag string     Tag for devices to follow
```

## Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl config - Manage configuration common to all devices in a factory

Version: 0.1.1

# fioctl_config_wireguard

## fioctl config wireguard

Show current wireguard server config for factory

```
fioctl config wireguard [flags]
```

### Options

```
      --disable   Disable VPN access for all devices
  -h, --help      help for wireguard
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl config - Manage configuration common to all devices in a factory

Version: 0.1.1

# fioctl_configure-docker

## fioctl configure-docker

Configure a hub.foundries.io Docker credential helper

### Synopsis

Configure a Docker credential helper that allows Docker to access hub.foundries.io.

This command will likely need to be run as root. It creates a symlink, docker-credential-fio, in the same directory as the docker client binary.

NOTE: The credentials will need the "containers:read" scope to work with Docker

```
fioctl configure-docker [flags]
```

### Options

```
      --creds-path string      Path to install credential helper (default
"/opt/homebrew/Cellar/pyenv-virtualenv/1.2.1/shims")
      --docker-config string   Docker config file to update (default
"/Users/kat/.docker/config.json")
  -h, --help                   help for configure-docker
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_configure-git

## fioctl configure-git

Configure a source.foundries.io Git credential helper

### Synopsis

Configure a Git credential helper that allows Git to access source.foundries.io.

This command will likely need to be run as root. It creates a symlink, git-credential-fio, in the same directory as the git client binary.

NOTE: The credentials will need the "source:read-update" scope to work with Git

```
fioctl configure-git [flags]
```

### Options

```
      --creds-path string   Path to install credential helper. This needs to be
writable and in $PATH (default "/opt/homebrew/Cellar/pyenv-
virtualenv/1.2.1/shims")
  -h, --help                help for configure-git
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_devices

## fioctl devices

Manage devices registered to a factory

### Options

```
-f, --factory string   Factory to list targets for
-h, --help             help for devices
-t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
-c, --config string   config file (default is $HOME/.config/fioctl.yaml)
-v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories
- fioctl devices apps-states - List states of Apps reported by a device
- fioctl devices chown - Change the device's owner
- fioctl devices config - Device configuration
- fioctl devices delete - Delete a device(s) registered to a factory.
- fioctl devices delete-denied - Remove a device UUID from the deny list
- fioctl devices list - List devices registered to factories. Optionally include filepath style patterns to limit to device names. eg device-*
- fioctl devices list-denied - List device UUIDs that have been denied access to the device-gateway
- fioctl devices rename - Rename a device
- fioctl devices show - Show details of a specific device
- fioctl devices updates - Show updates performed on a device

Version: 0.1.1

# fioctl_devices_apps-states

## fioctl devices apps-states

List states of Apps reported by a device

```
fioctl devices apps-states <name> [flags]
```

### Options

```
  -h, --help        help for apps-states
  -n, --limit int   Limit the number of Apps states to display. (default 1)
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_devices_chown

## fioctl devices chown

Change the device's owner

### Synopsis

Change the owner of a device. This command can only be run by factory admins and owners. The new owner-id can be found by running 'fioctl users'

```
fioctl devices chown <device> <new-owner-id> [flags]
```

### Options

```
  -h, --help   help for chown
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_devices_config

## fioctl devices config

Device configuration

### Options

```
  -h, --help   help for config
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices - Manage devices registered to a factory
- fioctl devices config delete - Delete file from the current configuration
- fioctl devices config group - Assign a device to an existing factory device group
- fioctl devices config log - Show a changelog of device's configuration
- fioctl devices config rotate-certs - Rotate a device's x509 keypair used to connect to the device gateway
- fioctl devices config set - Create a secure configuration for the device
- fioctl devices config updates - Configure aktualizr-lite settings for how updates are applied to a device
- fioctl devices config wireguard - Enable or disable wireguard VPN for this device

Version: 0.1.1

# fioctl_devices_config_delete

## fioctl devices config delete

Delete file from the current configuration

```
fioctl devices config delete <device> <file> [flags]
```

### Options

```
  -h, --help   help for delete
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices config - Device configuration

Version: 0.1.1

# fioctl_devices_config_group

## fioctl devices config group

Assign a device to an existing factory device group

```
fioctl devices config group <device> [<group>] [flags]
```

### Options

```
  -h, --help    help for group
      --unset   Unset an associated device group
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices config - Device configuration

Version: 0.1.1

# fioctl_devices_config_log

## fioctl devices config log

Show a changelog of device's configuration

```
fioctl devices config log <device> [flags]
```

### Options

```
  -h, --help         help for log
  -n, --limit int    Limit the number of results displayed.
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices config - Device configuration

Version: 0.1.1

# fioctl_devices_config_rotate-certs

## fioctl devices config rotate-certs

Rotate a device's x509 keypair used to connect to the device gateway

### Synopsis

This command will send a fioconfig change to a device to instruct it to perform a certificate rotation using the EST server configured with "fioctl keys est".

This command will only work for devices running LmP version 90 and later.

```
fioctl devices config rotate-certs <device> [flags]
```

### Options

```
      --dryrun                Show what the fioconfig entry will be and exit
  -p, --est-port int          The EST server port (default 8443)
  -e, --est-resource string   The path the to EST resource on your server (default
"/.well-known/est")
  -h, --help                  help for rotate-certs
      --hsm-cert-ids string   Available PKCS11 slot IDs for the client certificate
(default "03,09")
      --hsm-pkey-ids string   Available PKCS11 slot IDs for the private key
(default "01,07")
  -r, --reason string         The reason for changing the cert
      --server-name string    EST server name when not using the Foundries managed
server. e.g. est.example.com
```

### Options inherited from parent commands

```

```

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl devices config - Device configuration

Version: 0.1.1

# fioctl_devices_config_set

## fioctl devices config set

Create a secure configuration for the device

### Synopsis

Creates a secure configuration for device encrypting the contents each file using the device's public key. The fioconfig daemon running on each device will then be able to grab the latest version of the device's configuration and apply it. The maximum size of a config is 1Mb.

```
fioctl devices config set <device> <file1=content> <file2=content ...> [flags]
```

### Examples

```
  # Basic use can be done with command line arguments:
  fioctl device config set my-device npmtok="root" githubtok="1234"
readme.md==./readme.md

  There are several ways how to pass a file content into this command:
  - with filename="filecontent" format, a file content is passed directly.
  - with filename==/path/to/file format, a file content is read from a specified
file path.

  # The device configuration format also allows specifying what command
  # to run after a configuration file is updated on the device. To take
  # advantage of this, the "--raw" flag must be used.
  cat >tmp.json <<EOF
  {
    "reason": "I want to use the on-changed attribute",
    "files": [
      {
        "name": "npmtok",
        "value": "root",
        "on-changed": ["/usr/bin/touch", "/tmp/npmtok-changed"]
```

```
      },
      {
        "name": "A-Readable-Value",
        "value": "This won't be encrypted and will be visible from the API",
        "unencrypted": true
      },
      {
        "name": "githubtok",
        "value": "1234"
      }
    ]
  }
  > EOF
  fioctl devices config set my-device --raw ./tmp.json

  # fioctl will read in tmp.json, encrypt its contents, and upload it
  # to the OTA server. Instead of using ./tmp.json, the command can take
  # a "-" and will read the content from STDIN instead of a file.
```

## Options

```
      --create          Replace the whole config with these values. Default is to
merge these values in with the existing config values
  -h, --help            help for set
      --raw             Use raw configuration file
  -m, --reason string   Add a message to store as the "reason" for this change
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl devices config - Device configuration

Version: 0.1.1

# fioctl_devices_config_updates

## fioctl devices config updates

Configure aktualizr-lite settings for how updates are applied to a device

### Synopsis

View or change configuration parameters used by aktualizr-lite for updating a device. When run with no options, this command will print out how the device is currently configured and reporting.

```
fioctl devices config updates <device> [flags]
```

### Examples

```
  # Make a device start taking updates from Targets tagged with "devel"
  fioctl devices config updates <device> --tag devel

  # Set the docker apps a device will run:
  fioctl devices config updates <device> --apps shellhttpd

  # Set the docker apps and the tag:
  fioctl devices config updates <device> --apps shellhttpd --tag master
```

### Options

```
      --apps string   comma,separate,list
      --dryrun        Only show what would be changed
      --force         DANGER: For a config on a device that might result in
corruption
  -h, --help          help for updates
      --tag string    Target tag for device to follow
```

## Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl devices config - Device configuration

Version: 0.1.1

# fioctl_devices_config_wireguard

## fioctl devices config wireguard

Enable or disable wireguard VPN for this device

```
fioctl devices config wireguard <device> [enable|disable] [flags]
```

### Options

```
  -h, --help   help for wireguard
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices config - Device configuration

Version: 0.1.1

# fioctl_devices_delete-denied

## fioctl devices delete-denied

Remove a device UUID from the deny list

### Synopsis

Remove a device UUID from the deny list so that the UUID can be re-used. This is handy for Factories using HSMs and a factory-registration-reference server.

```
fioctl devices delete-denied <uuid> [<uuid>...] [flags]
```

### Options

```
  -h, --help   help for delete-denied
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_devices_delete

## fioctl devices delete

Delete a device(s) registered to a factory.

```
fioctl devices delete [flags]
```

### Options

```
  -h, --help   help for delete
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_devices_list-denied

## fioctl devices list-denied

List device UUIDs that have been denied access to the device-gateway

### Synopsis

Devices created using a factory-registration-reference server get created created on-demand. Because of this, devices are placed into a deny-list when they are deleted, so that they can't continue to access the system by getting re-created.

```
fioctl devices list-denied [flags]
```

### Options

```
  -h, --help         help for list-denied
  -n, --limit int    Number of devices to paginate by. Allowed values:
10,20,30,40,50,100,200,500,1000 (default 500)
  -p, --page int     Page of devices to display when pagination is needed (default
1)
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_devices_list

## fioctl devices list

List devices registered to factories. Optionally include filepath style patterns to limit to device names. eg device-*

### Synopsis

Available columns for display:

- apps
- created-at
- created-by
- curent-update
- device-group
- factory
- is-prod
- is-wave
- last-seen
- name
- ostree-hash
- owner
- status
- tag
- target
- up-to-date
- updated-at
- updated-by
- uuid

```
fioctl devices list [pattern] [flags]
```

## Options

```
  -g, --by-group string                    Only list devices belonging to this
group (factory is mandatory)
      --by-tag string                      Only list devices configured with the
given tag
      --by-target string                   Only list devices updated to the given
target name
      --columns strings                    Specify which columns to display
(default [name,target,status,apps,up-to-date,is-prod])
  -h, --help                               help for list
      --just-mine                          Only include devices owned by you
  -n, --limit int                          Number of devices to paginate by.
Allowed values: 10,20,30,40,50,100,200,500,1000 (default 500)
      --offline-threshold int              List the device as 'OFFLINE' if not
seen in the last X hours (default 4)
  -p, --page int                           Page of devices to display when
pagination is needed (default 1)
      --sort-by-last-seen string[="asc"]   Sort by last-seen (asc, desc); default
sort is by owner and name
      --sort-by-name string[="asc"]        Sort by name (asc, desc); default sort
is by owner and name
      --uuid string                        Find device with the given UUID
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_devices_rename

## fioctl devices rename

Rename a device

```
fioctl devices rename <current name> <new name> [flags]
```

### Options

```
  -h, --help   help for rename
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_devices_show

## fioctl devices show

Show details of a specific device

```
fioctl devices show <name> [flags]
```

### Options

```
      --aktoml   Show aktualizr-lite toml config
  -h, --help     help for show
  -i, --hwinfo   Show HW Information
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_devices_updates

## fioctl devices updates

Show updates performed on a device

```
fioctl devices updates <device> [<update-id>] [flags]
```

### Examples

```
# List all updates performed on a device:
fioctl devices updates <device1>

# List the last 2 updates:
fioctl devices updates <device> -n2

# Show the details of an update:
fioctl devices updates <device> <update-id>

# Show the most recent update with bash help:
fioctl devices updates <device> $(fioctl devices updates <device> -n1 | tail -n1 |
cut -f1 -d\ )
```

### Options

```
  -h, --help         help for updates
  -n, --limit int    Limit the number of updates displayed.
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
```

```
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl devices - Manage devices registered to a factory

Version: 0.1.1

# fioctl_el2g

## fioctl el2g

Manage EdgeLock 2Go integration

### Synopsis

This is an optional feature that must be enabled by Foundries.io customer support

### Options

```
  -f, --factory string   Factory to list targets for
  -h, --help             help for el2g
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories
- fioctl el2g config-aws-iot - Setup EdgeLock 2Go support in AWS IOT
- fioctl el2g config-device-gateway - Setup EdgeLock 2Go support for device gateway
- fioctl el2g devices - Manage devices for EdgeLock 2Go
- fioctl el2g status - Show the overall status of the Edgelock 2Go integration

Version: 0.1.1

# fioctl_el2g_config-aws-iot

## fioctl el2g config-aws-iot

Setup EdgeLock 2Go support in AWS IOT

```
fioctl el2g config-aws-iot [flags]
```

### Options

```
  -h, --help   help for config-aws-iot
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl el2g - Manage EdgeLock 2Go integration

Version: 0.1.1

# fioctl_el2g_config-device-gateway

## fioctl el2g config-device-gateway

Setup EdgeLock 2Go support for device gateway

```
fioctl el2g config-device-gateway [flags]
```

### Examples

```
fioctl el2g config-device-gateway --pki-dir /tmp/factory-pki
```

### Options

```
-h, --help             help for config-device-gateway
    --pki-dir string   Directory container factory PKI keys
```

### Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl el2g - Manage EdgeLock 2Go integration

Version: 0.1.1

# fioctl_el2g_devices

## fioctl el2g devices

Manage devices for EdgeLock 2Go

### Options

```
  -h, --help   help for devices
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl el2g - Manage EdgeLock 2Go integration
- fioctl el2g devices add - Grant device access to EdgeLock 2GO
- fioctl el2g devices delete - Revoke device access to EdgeLock 2GO
- fioctl el2g devices list - List devices configured to use EdgeLock 2Go
- fioctl el2g devices show - Show the integrations details for a device

Version: 0.1.1

# fioctl_el2g_devices_add

## fioctl el2g devices add

Grant device access to EdgeLock 2GO

```
fioctl el2g devices add <NC12 product-id> <device-id> [flags]
```

### Examples

```
# Add a device with an SE050 (product ID: 935389312472)
# The product IDs configured for you factory can be found by running
#  fioctl el2g status
# Device ID can be found on a device by running:
#  $ ssscli se05x uid | grep "Unique ID:" | cut -d: -f2
#  ssscli se05x uid | grep "Unique ID:" | cut -d: -f2
#  04005001eee3ba1ee96e60047e57da0f6880
# This ID is hexadecimal and must be prefixed in the CLI with 0x. For example:
fioctl el2g devices add 935389312472 0x04005001eee3ba1ee96e60047e57da0f6880

# Add a production device with an SE051 HSM (product ID: 935414457472)
fioctl el2g devices add --production 935414457472
0x04005001eee3ba1ee96e60047e57da0f6880
```

### Options

```
  -h, --help         help for add
      --production   A production device
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
```

```
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl el2g devices - Manage devices for EdgeLock 2Go

Version: 0.1.1

# fioctl_el2g_devices_delete

## fioctl el2g devices delete

Revoke device access to EdgeLock 2GO

```
fioctl el2g devices delete <NC12 product-id> <device-id> [flags]
```

### Options

```
  -h, --help         help for delete
      --production   A production device
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl el2g devices - Manage devices for EdgeLock 2Go

Version: 0.1.1

# fioctl_el2g_devices_list

## fioctl el2g devices list

List devices configured to use EdgeLock 2Go

```
fioctl el2g devices list [flags]
```

### Options

```
  -h, --help   help for list
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl el2g devices - Manage devices for EdgeLock 2Go

Version: 0.1.1

# fioctl_el2g_devices_show

## fioctl el2g devices show

Show the integrations details for a device

```
fioctl el2g devices show <device-id> [flags]
```

### Options

```
  -h, --help   help for show
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl el2g devices - Manage devices for EdgeLock 2Go

Version: 0.1.1

# fioctl_el2g_status

## fioctl el2g status

Show the overall status of the Edgelock 2Go integration

```
fioctl el2g status [flags]
```

### Options

```
  -h, --help   help for status
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl el2g - Manage EdgeLock 2Go integration

Version: 0.1.1

# fioctl_event-queues

## fioctl event-queues

Manage event queues configured for a Factory

### Synopsis

Event queues provide a way for customers to receive notifications about events happening in a Factory such as when a device is first seen or has started an over-the-air update.

There are two types of event queues: push and pull. A pull queue works like a traditional message queue system. Push queues are synonymous with web hooks.

### Options

```
-f, --factory string   Factory to list targets for
-h, --help             help for event-queues
-t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
-c, --config string   config file (default is $HOME/.config/fioctl.yaml)
-v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories
- fioctl event-queues list - List configured event queues
- fioctl event-queues listen - Listen to events sent to a pull queue
- fioctl event-queues mk-pull - Create a message queue that can be polled for events
- fioctl event-queues mk-push - Create an event queue that will ingest events at the URL
- fioctl event-queues rm - Remove an event queue

Version: 0.1.1

# fioctl_event-queues_list

## fioctl event-queues list

List configured event queues

```
fioctl event-queues list [flags]
```

### Options

```
  -h, --help   help for list
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl event-queues - Manage event queues configured for a Factory

Version: 0.1.1

# fioctl_event-queues_listen

## fioctl event-queues listen

Listen to events sent to a pull queue

### Synopsis

Listens to pull queue events. This command is useful for debugging or as a reference implementation of queue listener.

```
fioctl event-queues listen <label> <creds file> [flags]
```

### Options

```
  -h, --help   help for listen
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl event-queues - Manage event queues configured for a Factory

Version: 0.1.1

# fioctl_event-queues_mk-pull

## fioctl event-queues mk-pull

Create a message queue that can be polled for events

### Synopsis

Create a message queue that can be polled for events via the Google PubSub API:

https://cloud.google.com/pubsub/docs/reference/libraries

The command creates a credentials file to a scoped service account capable of polling the resulting PubSub subscription.

```
fioctl event-queues mk-pull <label> <pubsub creds file> [flags]
```

### Options

```
  -h, --help   help for mk-pull
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl event-queues - Manage event queues configured for a Factory

Version: 0.1.1

# fioctl_event-queues_mk-push

## fioctl event-queues mk-push

Create an event queue that will ingest events at the URL

```
fioctl event-queues mk-push <label> <url> [flags]
```

### Options

```
  -h, --help   help for mk-push
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl event-queues - Manage event queues configured for a Factory

Version: 0.1.1

# fioctl_event-queues_rm

## fioctl event-queues rm

Remove an event queue

```
fioctl event-queues rm <label> [flags]
```

### Options

```
  -h, --help   help for rm
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl event-queues - Manage event queues configured for a Factory

Version: 0.1.1

# fioctl_keys

## fioctl keys

Manage keys in use by your factory fleet

### Options

```
  -f, --factory string   Factory to list targets for
  -h, --help             help for keys
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories
- fioctl keys ca - Manage Public Key Infrastructure for your device gateway
- fioctl keys tuf - Manage The Update Framework Keys for your factory

Version: 0.1.1

# fioctl_keys_ca

## fioctl keys ca

Manage Public Key Infrastructure for your device gateway

### Synopsis

Every factory can have its own dedicated device gateway. This allows customers to own the PKI infrastructure of their factory. This infrastructure is used to manage mutual TLS between your devices and the Foundries.io device gateway.

### Options

```
  -h, --help   help for ca
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys - Manage keys in use by your factory fleet
- fioctl keys ca create - Create PKI infrastructure to manage mutual TLS for the device gateway
- fioctl keys ca show - Show what certificates are known to the factory
- fioctl keys ca update - Update the list of CAs that can create client certificates for devices

Version: 0.1.1

# fioctl_keys_ca_create

## fioctl keys ca create

Create PKI infrastructure to manage mutual TLS for the device gateway

### Synopsis

Perform a one-time operation to set up PKI infrastructure for managing the device gateway. This command creates a few things:

### Root of trust for your factory: factory_ca.key / factory_ca.pem

The factory_ca keypair is generated by this command to define the PKI root of trust for this factory.

- factory_ca.key - An EC prime256v1 private key that should be STORED OFFLINE.
- factory_ca.pem - The public x509 certificate that is shared with Foundries.io. Once set, all future PKI related changes will require proof you own this certificate.

### online-ca - A Foundries.io owned keypair to support lmp-device-register

In order for lmp-device-register to work, Foundries.io needs the ability to sign client certificates for devices. If enabled, the factory_ca keypair will sign the certificate signing request returned from the API.

This is optional.

### local-ca - A keypair you own

This keypair can be used for things like your manufacturing process where you may set up devices without having to communicate with Foundries.io web services. This keypair is capable of signing client certificates for devices. If enabled, the local-ca.pem will be shared with the Foundries.io device gateway so that it will trust the client certificate of devices signed with this keypair.

This is optional.

```
fioctl keys ca create <PKI Directory> [flags]
```

## Options

```
  -h, --help                     help for create
      --hsm-module string        Create key on an PKCS#11 compatible HSM using
this module
      --hsm-pin string           The PKCS#11 PIN to set up on the HSM, if using
one
      --hsm-token-label string   The label of the HSM token created for this
(default "device-gateway-root")
      --local-ca                 Create a local CA that you can use for signing
your own device certificates (default true)
      --online-ca                Create an online CA owned by Foundries that works
with lmp-device-register (default true)
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl keys ca - Manage Public Key Infrastructure for your device gateway

Version: 0.1.1

# fioctl_keys_ca_show

## fioctl keys ca show

Show what certificates are known to the factory

```
fioctl keys ca show [flags]
```

### Options

```
  -h, --help              help for show
      --just-device-cas   Only show device authenticate certificates trusted by
the device-gateway
      --just-root         Only show the Factory root CA certificate
      --just-tls          Only show the device-gateway TLS certificate
      --pretty            Display human readable output of each certificate
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys ca - Manage Public Key Infrastructure for your device gateway

Version: 0.1.1

# fioctl_keys_ca_update

## fioctl keys ca update

Update the list of CAs that can create client certificates for devices

```
fioctl keys ca update <ca-crts file> [flags]
```

### Options

```
  -h, --help   help for update
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys ca - Manage Public Key Infrastructure for your device gateway

Version: 0.1.1

# fioctl_keys_tuf

## fioctl keys tuf

Manage The Update Framework Keys for your factory

### Synopsis

These sub-commands allow you to manage your Factory's TUF private keys to ensure that you are in complete control of your OTA metadata.

### Options

```
  -h, --help   help for tuf
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys - Manage keys in use by your factory fleet
- fioctl keys tuf rotate-all-keys - Rotate all online and offline TUF signing keys for the Factory
- fioctl keys tuf rotate-offline-key - Rotate the offline TUF signing key for the Factory
- fioctl keys tuf show-root - Show the Factory's TUF root metadata
- fioctl keys tuf updates - Manage updates to the TUF root for your factory (expert mode)

Version: 0.1.1

# fioctl_keys_tuf_rotate-all-keys

## fioctl keys tuf rotate-all-keys

Rotate all online and offline TUF signing keys for the Factory

### Synopsis

Rotate the following TUF keys for the Factory:

- offline root signing key;
- offline targets signing key;
- online targets signing key;
- online snapshot signing key;
- online timestamp signing key.

The new signing keys are rotated in both CI and production TUF root transactionally.

When you rotate all TUF signing leys:

- if there are CI or production targets in your factory, they are re-signed using the new keys.
- if there is an active wave in your factory, this command is not allowed.
- new CI targets upload is temporarily disabled for the duration of transaction.

```
fioctl keys tuf rotate-all-keys --keys=<offline-creds.tgz> [flags]
```

### Examples

```
Migrate an old factory to use Ed25519 key type for all TUF signing keys (online
and offline):
  fioctl keys tuf rotate-all-keys --key-type=ed25519 \
    --keys=offline-tuf-root-keys.tgz --targets-keys=offline-tuf-targets-keys.tgz
```

## Options

```
  -m, --changelog string      Reason for doing rotation. Saved in root metadata
for tracking change history.
      --first-time            Used for the first customer rotation. The command
will download the initial root key.
  -h, --help                  help for rotate-all-keys
  -y, --key-type string       Key type, supported: Ed25519, RSA. (default
"ED25519")
  -k, --keys string           Path to <offline-creds.tgz> used to sign TUF root.
  -K, --targets-keys string   Path to <offline-targets-creds.tgz> used to sign
prod & wave TUF targets.
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl keys tuf - Manage The Update Framework Keys for your factory

Version: 0.1.1

# fioctl_keys_tuf_rotate-offline-key

## fioctl keys tuf rotate-offline-key

Rotate the offline TUF signing key for the Factory

### Synopsis

Rotate the TUF root or TUF targets offline signing key for the Factory.

The new signing key is rotated in both CI and production TUF root transactionally.

When you rotate the TUF targets offline signing key:

- if there are production targets in your factory, they are re-signed using the new key.
- if there is an active wave in your factory, the TUF targets rotation is not allowed.

```
fioctl keys tuf rotate-offline-key --role root|targets --keys=<offline-creds.tgz>
[flags]
```

### Examples

```
  # Take ownership of TUF root and targets keys for a new factory, keep them in
separate files:
  fioctl keys tuf rotate-offline-key --role=root \
    --keys=offline-tuf-root-keys.tgz --first-time
  fioctl keys tuf rotate-offline-key --role=targets \
    --keys=offline-tuf-root-keys.tgz --targets-keys=offline-tuf-targets-keys.tgz

  # Rotate offline TUF targets key using the Ed25519 elliptic curve to generate a
new key pair:
  fioctl keys tuf rotate-offline-key --role=targets --key-type=ed25519 \
    --keys=offline-tuf-root-keys.tgz --targets-keys=offline-tuf-targets-keys.tgz
```

### Options

```
  -m, --changelog string      Reason for doing rotation. Saved in root metadata
for tracking change history.
      --first-time            Used for the first customer rotation. The command
will download the initial root key.
  -h, --help                  help for rotate-offline-key
  -y, --key-type string       Key type, supported: Ed25519, RSA. (default
"ED25519")
  -k, --keys string           Path to <offline-creds.tgz> used to sign TUF root.
  -r, --role string           TUF role name, supported: Root, Targets.
  -K, --targets-keys string   Path to <offline-targets-creds.tgz> used to sign
prod & wave TUF targets.
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl keys tuf - Manage The Update Framework Keys for your factory

Version: 0.1.1

# fioctl_keys_tuf_show-root

## fioctl keys tuf show-root

Show the Factory's TUF root metadata

```
fioctl keys tuf show-root [flags]
```

### Options

```
  -h, --help   help for show-root
      --prod   Show the production version
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys tuf - Manage The Update Framework Keys for your factory

Version: 0.1.1

# fioctl_keys_tuf_updates

## fioctl keys tuf updates

Manage updates to the TUF root for your factory (expert mode)

### Synopsis

These sub-commands allow you to transactionally stage and apply changes to your Factory's TUF private keys in a granular way familiar for TUF experts.

The TUF updates transaction starts by running the "fioctl keys tuf updates init" command. That command returns a unique secure Transaction ID which is then required for other actions. The admin initiating the transaction should save that TXID for the timespan of the transaction. It must only be shared with those Factory admins which will participate in the transaction.

Typically, admin(s) will run other subcommands to make changes to the TUF root (see examples). The staged changes can be checked using the "fioctl keys tuf updates review" command.

Finally, the transaction can be applied using the "fioctl keys tuf updates apply" command. If admin decides to abandon the staged changes they can run "fioctl keys tuf updates cancel".

For increased safety there can be only one active TUF updates transaction at a time.

### Examples

```
- Take ownership of TUF root and targets keys for a new factory, keep them on
separate machines:
  1. On TUF root admin's shell:
     fioctl keys tuf updates init --first-time --keys=tuf-root-keys.tgz
  2. The above command prints a transaction ID (e.g. abcdef42) to be shared with
TUF targets admin.
  3. On TUF targets admin's shell:
     fioctl keys tuf updates rotate-offline-key \
        --role=targets --txid=abcdef42 --targets-keys=tuf-targets-keys.tgz
  4. On TUF root admin's shell:
```

```
   fioctl keys tuf updates rotate-offline-key \
      --role=root --txid=abcdef42 --keys=tuf-root-keys.tgz --sign
5. On TUF root admin's shell:
   fioctl keys tuf updates apply --txid=abcdef42
```

## Options

```
-h, --help   help for updates
```

## Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl keys tuf - Manage The Update Framework Keys for your factory
- fioctl keys tuf updates apply - Apply staged TUF root updates for the Factory
- fioctl keys tuf updates cancel - Cancel staged TUF root updates for the Factory
- fioctl keys tuf updates init - Start a new transaction to update TUF root keys
- fioctl keys tuf updates review - Show the Factory's TUF root metadata
- fioctl keys tuf updates rotate-offline-key - Stage rotation of the offline TUF signing key for the Factory
- fioctl keys tuf updates rotate-online-key - Stage rotation of the online TUF signing key for the Factory
- fioctl keys tuf updates sign - Sign the staged TUF root for your Factory with the offline root key

Version: 0.1.1

# fioctl_keys_tuf_updates_apply

## fioctl keys tuf updates apply

Apply staged TUF root updates for the Factory

```
fioctl keys tuf updates apply [flags]
```

### Options

```
  -h, --help          help for apply
  -x, --txid string   TUF root updates transaction ID.
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys tuf updates - Manage updates to the TUF root for your factory (expert mode)

Version: 0.1.1

# fioctl_keys_tuf_updates_cancel

## fioctl keys tuf updates cancel

Cancel staged TUF root updates for the Factory

```
fioctl keys tuf updates cancel [flags]
```

### Options

```
  -h, --help   help for cancel
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys tuf updates - Manage updates to the TUF root for your factory (expert mode)

Version: 0.1.1

# fioctl_keys_tuf_updates_init

## fioctl keys tuf updates init

Start a new transaction to update TUF root keys

```
fioctl keys tuf updates init [flags]
```

### Options

```
  -m, --changelog string   Reason for doing this operation. Saved in root metadata
to track change history.
      --first-time         Used for the first customer rotation. The command will
download the initial root key.
  -h, --help               help for init
  -k, --keys string        Path to <offline-creds.tgz> used to store initial root
key.
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys tuf updates - Manage updates to the TUF root for your factory (expert mode)

Version: 0.1.1

# fioctl_keys_tuf_updates_review

## fioctl keys tuf updates review

Show the Factory's TUF root metadata

```
fioctl keys tuf updates review [flags]
```

### Options

```
      --diff   Show the unified diff between current and staged root.json
  -h, --help   help for review
      --prod   Show the production root.json
      --raw    Show the raw root.json
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys tuf updates - Manage updates to the TUF root for your factory (expert mode)

Version: 0.1.1

# fioctl_keys_tuf_updates_rotate-offline-key

## fioctl keys tuf updates rotate-offline-key

Stage rotation of the offline TUF signing key for the Factory

### Synopsis

Stage rotation of the offline TUF signing key for the Factory.

The new offline signing key will be used in both CI and production TUF root.

When you rotate the TUF targets offline signing key:

- if there are production targets in your factory, they are re-signed using the new key.
- if there is an active wave in your factory, the TUF targets rotation is not allowed.

```
fioctl keys tuf updates rotate-offline-key --role root|targets --txid=<txid> --
keys=<tuf-root-keys.tgz> [flags]
```

### Examples

```
- Rotate offline TUF root key and re-sign the new TUF root with both old and new
keys:
  fioctl keys tuf updates rotate-offline-key \
    --txid=abc --role=root --keys=tuf-root-keys.tgz --sign
- Rotate offline TUF root key explicitly specifying new key type (and signing
algorithm):
  fioctl keys tuf updates rotate-offline-key \
    --txid=abc --role=root --keys=tuf-root-keys.tgz --key-type=ed25519
- Rotate offline TUF targets key and re-sign the new TUF root with offline TUF
root key:
  fioctl keys tuf updates rotate-offline-key \
```

```
    --txid=abc --role=targets --keys=tuf-root-keys.tgz --sign
- Rotate offline TUF targets key and store the new key in a separate file (and re-
sign TUF root):
  fioctl keys tuf updates rotate-offline-key \
    --txid=abc --role=targets --keys=tuf-root-keys.tgz --targets-keys=tuf-targets-
keys.tgz --sign
```

## Options

```
  -h, --help                  help for rotate-offline-key
  -y, --key-type string       Key type, supported: Ed25519, RSA. (default
"ED25519")
  -k, --keys string           Path to <tuf-root-keys.tgz> used to sign TUF root.
  -r, --role string           TUF role name, supported: Root, Targets.
  -s, --sign                  Sign the new TUF root using the offline root keys.
  -K, --targets-keys string   Path to <tuf-targets-keys.tgz> used to sign prod &
wave TUF targets.
  -x, --txid string           TUF root updates transaction ID.
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl keys tuf updates - Manage updates to the TUF root for your factory (expert mode)

Version: 0.1.1

# fioctl_keys_tuf_updates_rotate-online-key

## fioctl keys tuf updates rotate-online-key

Stage rotation of the online TUF signing key for the Factory

### Synopsis

Stage rotation of the online TUF signing key for the Factory.

The new online signing key will be used in both CI and production TUF root.

When you rotate the TUF online signing key:

- if there are CI or production targets in your factory, they are re-signed using the new key.
- if there is an active wave in your factory, the TUF online key rotation is not allowed.
- the new wave cannot be created until you apply the online keys rotation.

When you apply the online key rotation, these features are temporarily disabled until it succeeds:

- new CI targets upload (including the targets upload during CI builds).
- automatic re-signing of expired TUF roles using online keys (both CI and production targets).

```
fioctl keys tuf updates rotate-online-key --role targets|snapshot|timestamp [--
txid=<txid>] [flags]
```

### Examples

```

- Rotate online TUF targets key and re-sign the new TUF root:
  fioctl keys tuf updates rotate-online-key \
    --txid=abc --role=targets --keys=tuf-root-keys.tgz --sign
- Rotate all online TUF keys explicitly specifying new key type (and signing
```

```
algorithm):
  fioctl keys tuf updates rotate-online-key \
    --txid=abc --role=targets,snapshot,timestamp --key-type=ed25519
```

## Options

```
  -h, --help              help for rotate-online-key
  -y, --key-type string   Key type, supported: Ed25519, RSA. (default "ED25519")
  -k, --keys string       Path to <tuf-root-keys.tgz> used to sign TUF root.
  -r, --role strings      TUF role name, supported: Targets, Snapshot, Timestamp.
  -s, --sign              Sign the new TUF root using the offline root keys.
  -x, --txid string       TUF root updates transaction ID.
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl keys tuf updates - Manage updates to the TUF root for your factory (expert mode)

Version: 0.1.1

# fioctl_keys_tuf_updates_sign

## fioctl keys tuf updates sign

Sign the staged TUF root for your Factory with the offline root key

### Synopsis

Sign the staged TUF root for your Factory with the offline root key

```
fioctl keys tuf updates sign --txid=<txid> --keys=<tuf-root-keys.tgz> [flags]
```

### Options

```
  -h, --help          help for sign
  -k, --keys string   Path to <tuf-root-keys.tgz> used to sign TUF root.
  -x, --txid string   TUF root updates transaction ID.
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl keys tuf updates - Manage updates to the TUF root for your factory (expert mode)

Version: 0.1.1

# fioctl_login

## fioctl login

Access Foundries.io services with your client credentials

```
fioctl login [flags]
```

### Options

```
  -h, --help                   help for login
      --oauth-url string       OAuth URL to authenticate with (default
"https://app.foundries.io/oauth")
      --refresh-access-token   Refresh your current oauth2 access token. This is
used when a token's scopes have been updated in app.foundries.io
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_logout

## fioctl logout

Remove Foundries.io client credentials from system

```
fioctl logout [flags]
```

### Options

```
  -h, --help   help for logout
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_secrets

## fioctl secrets

Manage secret crendentials configured in a factory

### Options

```
-f, --factory string   Factory to list targets for
-h, --help             help for secrets
-t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
-c, --config string   config file (default is $HOME/.config/fioctl.yaml)
-v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories
- fioctl secrets list - List secret credentials configured in the factory
- fioctl secrets update - Update secret(s) in a factory

Version: 0.1.1

# fioctl_secrets_list

## fioctl secrets list

List secret credentials configured in the factory

```
fioctl secrets list [flags]
```

### Options

```
  -h, --help   help for list
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl secrets - Manage secret crendentials configured in a factory

Version: 0.1.1

# fioctl_secrets_update

## fioctl secrets update

Update secret(s) in a factory

```
fioctl secrets update secret_name=secret_val... [flags]
```

### Examples

```
# Create or update a secret
fioctl secrets update githubtok=foo

# Create or update a secret with value from a file
fioctl secrets update ssh-github.key==/tmp/ssh-github.key

# Delete a secret by setting it to an empty value. eg:
fioctl secrets update secret_name=
```

### Options

```
-h, --help   help for update
```

### Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl secrets - Manage secret crendentials configured in a factory

Version: 0.1.1

# fioctl_status

## fioctl status

Get dashboard view of a factory and its devices

```
fioctl status [flags]
```

### Options

```
  -f, --factory string          Factory to list targets for
  -h, --help                    help for status
      --offline-threshold int   Consider device 'OFFLINE' if not seen in the last
X hours (default 4)
  -t, --token string            API token from
https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_targets

## fioctl targets

Manage factory's TUF targets

### Options

```
-f, --factory string   Factory to list targets for
-h, --help             help for targets
-t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
-c, --config string   config file (default is $HOME/.config/fioctl.yaml)
-v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories
- fioctl targets add - Compose and add Targets to Factory's TUF targets metadata
- fioctl targets artifacts - Show artifacts created in CI for a Target
- fioctl targets image - Generate a system image with pre-loaded container images
- fioctl targets list - List targets.
- fioctl targets offline-update - Download Target content for an offline update
- fioctl targets prune - Prune target(s)
- fioctl targets show - Show details of a specific target.
- fioctl targets static-deltas - Generate static deltas to the given target version to make OTAs faster
- fioctl targets tag - Apply a comma separated list of tags to one or more targets.
- fioctl targets tail - Tail the console output of a live CI Run
- fioctl targets tests - Show testing done against a target

Version: 0.1.1

# fioctl targets add

Compose and add Targets to Factory's TUF targets metadata

## Synopsis

Compose new Targets out of the latest Targets tagged with the specified source tag and the specified via the command arguments eiter OSTree commit hashes or App URIs.

```
fioctl targets add --type <ostree | app> --tags <comma,separate,list of Target
tags> --src-tag <source Target tag> [--targets-creator <something about Targets
originator>]\
    <hardware ID> <ostree commit hash> [<hardware ID> <ostree commit hash>]  (for
ostree type)
    <App #1 URI> [App #N URI] (for app type)
```

```
fioctl targets add [flags]
```

## Examples

```
Add new ostree Targets:
    fioctl targets add --type ostree --tags dev,test --src-tag dev --targets-creator
ostree build" intel-corei7-64 00b2ad4a1dd7fe1e856a6d607ed492c354a423be22a44bad644092
raspberrypi4-64 5e05a59529dcdd54310945b2628d73c0533097d76cc483334925a901845b3794

Add new App Targets:
    fioctl targets add --type app --tags dev,test --src-tag dev
hub.foundries.io/factory/simpleapp@sha256:be955ad958ef37bcce5afaaad32a21b783b3cc29ec
hub.foundries.io/factory/app-03@sha256:59b080fe42d7c45bc81ea17ab772fc8b3bb5ef0950f74
```

## Options

```
      --dry-run                  don't post generated new Targets
  -h, --help                     help for add
      --quiet                    don't print generated new Targets to stdout
      --src-tag string           OSTree Target tag to base app targets on
      --tags string              comma,separate,list of Target tags
      --targets-creator string   optional name/comment/context about Targets
origination (default "fioctl")
      --type string              Target type
```

# Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

# SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_artifacts

## fioctl targets artifacts

Show artifacts created in CI for a Target

```
fioctl targets artifacts <target> [<artifact name>] [flags]
```

### Examples

```
# List all artifacts for Target 12
fioctl targets artifacts 12

# Dump console.log artifact to STDOUT
fioctl targets artifacts 12 publish-compose-apps/console.log

# Download an artifact. Progress is printed to STDERR, the contents is
# re-directed /tmp/tmp.gz
fioctl-linux-amd64 targets artifacts 207 \
  raspberrypi3-64/lmp-factory-image-raspberrypi3-64.wic.gz >/tmp/tmp.gz
```

### Options

```
-h, --help   help for artifacts
```

### Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_image

## fioctl targets image

Generate a system image with pre-loaded container images

```
fioctl targets image <target-name> [flags]
```

### Examples

```
fioctl targets image raspberrypi4-64-lmp-464 // preload all Target apps
fioctl targets image raspberrypi4-64-lmp-464 --apps app-00,app-01 // preload app-
00 and app-01
```

### Options

```
      --apps string              comma,separate,list of Target apps to preload
into a resultant image. All apps of Target are preloaded if the flag is not
defined or empty
      --ci-scripts-ref string    Override to a specific git-ref of ci-scripts
(default "master")
      --ci-scripts-repo string   Override to custom version of ci-scripts (default
"https://github.com/foundriesio/ci-scripts")
  -h, --help                     help for image
      --no-tail                  Don't tail output of CI Job
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_list

## fioctl targets list

List targets.

### Synopsis

Available columns for display:

- apps
- containers-sha
- hardware-ids
- manifest-sha
- origin
- overrides-sha
- tags
- version

```
fioctl targets list [flags]
```

### Options

```
      --by-tag string     Only list targets that match the given tag
      --columns strings   Specify which columns to display (default
[version,tags,apps,origin])
  -h, --help              help for list
      --production        Show the production version targets.json
  -r, --raw               Print raw targets.json
```

### Options inherited from parent commands

```

```

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_offline-update

## fioctl targets offline-update

Download Target content for an offline update

```
fioctl targets offline-update <target-name> <dst> --tag <tag> [--prod] [--expires-in-days <days>] [--tuf-only] [flags]
```

### Examples

```
    # Download update content of the production target #1451 tagged by "release-01" for "intel-corei7-64" hardware type
    fioctl targets offline-update intel-corei7-64-lmp-1451 /mnt/flash-drive/offline-update-content --tag release-01 --prod

    # Download update content of the CI target #1451 tagged by "devel" for "raspberrypi4-64" hardware type
    fioctl targets offline-update raspberrypi4-64-lmp-1448 /mnt/flash-drive/offline-update-content --tag devel --expires-in-days 15
```

### Options

```
  -e, --expires-in-days int   Desired metadata validity period in days (default 30)
  -h, --help                  help for offline-update
      --no-apps               Skip fetching Target Apps
      --prod                  Instruct to fetch content of production Target
      --tag string            Target tag
  -m, --tuf-only              Fetch only TUF metadata
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_prune

## fioctl targets prune

Prune target(s)

```
fioctl targets prune <target> [<target>...] [flags]
```

### Examples

```
  # prune a single target by name:
  fioctl targets prune intel-corei7-64-lmp-123

  # prune all targets with the "custom.tags" set to ["devel"]:
  fioctl targets prune --by-tag devel

  # prune all targets with the "custom.tags" set to ["devel"] except for the most
recent 10:
  fioctl targets prune --by-tag devel --keep-last=10

  # prune all targets with the "custom.tags" set to ["devel", "my-test"]:
  fioctl targets prune --by-tag devel my-test

  # see the list of targets to be pruned (based on the above example), don't prune
them:
  fioctl targets prune --by-tag devel my-test --dryrun
```

### Options

```
      --by-tag          Prune all targets by tags instead of name
      --dryrun          Only show what would be pruned
  -h, --help            help for prune
      --keep-last int   Keep the last X number of builds for a tag when pruning
      --no-tail         Don't tail output of CI Job
```

## Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_show

## fioctl targets show

Show details of a specific target.

```
fioctl targets show <version> [flags]
```

### Examples

```
# Show details of all Targets with version 42:
fioctl targets show 42

# Show a specific Target by name:
fioctl targets show intel-corei7-64-lmp-42
```

### Options

```
-h, --help                    help for show
    --production-tag string   Look up target from the production tag
```

### Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl targets - Manage factory's TUF targets

- fioctl targets show compose-app - Show details of a specific compose app.
- fioctl targets show sboms - Show SBOMs for a specific target.

Version: 0.1.1

# fioctl_targets_show_compose-app

## fioctl targets show compose-app

Show details of a specific compose app.

```
fioctl targets show compose-app <version> <app> [flags]
```

### Options

```
  -h, --help       help for compose-app
      --manifest   Show an app docker manifest
```

### Options inherited from parent commands

```
  -c, --config string           config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string          Factory to list targets for
      --production-tag string   Look up target from the production tag
  -t, --token string            API token from
https://app.foundries.io/settings/tokens/
  -v, --verbose                 Print verbose logging
```

### SEE ALSO

- fioctl targets show - Show details of a specific target.

Version: 0.1.1

# fioctl_targets_show_sboms

## fioctl targets show sboms

Show SBOMs for a specific target.

```
fioctl targets show sboms <version> [<build/run> [<artifact>]]  [flags]
```

### Examples

```
  # Show all SBOM files for Target version 42:
  fioctl targets show sboms 42

  # Show a subset of the SBOMS for this target. In this case, the 32-bit Arm
  # container SBOMS:
  fioctl targets show sboms 42 41/build-armhf

  # Show overview of a specific SBOM:
  fioctl targets show sboms 42 41/build-armhf alpine:latest/arm.spdx.json

  # Show overview of a specific SBOM as CSV:
  fioctl targets show sboms --format csv 42 41/build-armhf
alpine:latest/arm.spdx.json

  # Download all SBOMS for a target to /tmp:
  fioctl targets show sboms 42 --download /tmp

  # Download a filtered list of SBOMs to /tmp:
  fioctl targets show sboms 42 41/build-armhf alpine:latest/arm.spdx.json --
download /tmp

  # Download a specific SBOM as cyclonedx:
  fioctl targets show sboms 42 41/build-armhf --download /tmp --format cyclonedx

  # Download all SBOMS for a target to /tmp as CSV:
  fioctl targets show sboms 42 --download /tmp --format csv
```

## Options

```
      --download string   Download SBOM(s) to a directory
      --format string     The format to download/display. Must be one of table,
spdx, cyclonedx, or csv (default "table")
  -h, --help              help for sboms
```

## Options inherited from parent commands

```
  -c, --config string           config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string          Factory to list targets for
      --production-tag string   Look up target from the production tag
  -t, --token string            API token from
https://app.foundries.io/settings/tokens/
  -v, --verbose                 Print verbose logging
```

## SEE ALSO

- fioctl targets show - Show details of a specific target.

Version: 0.1.1

# fioctl_targets_static-deltas

## fioctl targets static-deltas

Generate static deltas to the given target version to make OTAs faster

### Synopsis

In many cases OTA updates will have many OSTree changes. These updates can be downloaded significantly faster by generating OSTree static deltas. Static deltas are generated with a "from(sha) -> to(sha)" type logic. This command takes the given Target version and will produce a number of static deltas to ensure devices will be updated efficiently.

```
fioctl targets static-deltas <target-version> [<from-version>...] [flags]
```

### Examples

```

  # There are two ways to run this command:

  # Generate static deltas for 30->42 and 31->42
  fioctl targets static-deltas 42 30 31

  # Find the target versions of all devices configured to the "prod" tag.
  # Generate a static delta from those versions to version 42.
  fioctl targets static-deltas --by-tag prod 42
```

### Options

```
      --by-tag string   Find from-versions devices on the given tag
      --dryrun          Only show what deltas would be produced
  -h, --help            help for static-deltas
      --hw-id string    Filter from and to targets by the given hardware ID
      --no-tail         Don't tail output of CI Job
```

## Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_tag

## fioctl targets tag

Apply a comma separated list of tags to one or more targets.

```
fioctl targets tag <target> [<target>...] [flags]
```

### Examples

```
# Promote Target #42 currently tagged as master
fioctl targets tag --tags master,promoted --by-version 42

# Tag a specific Target by name
fioctl targets tag --tags master,testing intel-corei7-64-lmp-42
```

### Options

```
    --append        Append the given tags rather than set them
    --by-version    Apply tags to all targets matching the given version(s)
    --dryrun        Just show the changes that would be applied
-h, --help          help for tag
    --no-tail       Don't tail output of CI Job
-T, --tags string   comma,separate,list
```

### Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_tail

## fioctl targets tail

Tail the console output of a live CI Run

```
fioctl targets tail <target> <run> [flags]
```

### Examples

```
  fioctl targets tail 12 build-amd64
```

### Options

```
  -h, --help   help for tail
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_targets_tests

## fioctl targets tests

Show testing done against a target

```
fioctl targets tests [<target> [<test-id> [<artifact name>]]] [flags]
```

### Examples

```
# List all testing performed in factory
fioctl targets tests

# Show tests run against Target 12
fioctl targets tests 12

# Show details of a specific test
fioctl targets tests 12 <test-id>

# Display a test artifact
fioctl targets tests 12 <test-id> console.log
```

### Options

```
-h, --help   help for tests
```

### Options inherited from parent commands

```
-c, --config string    config file (default is $HOME/.config/fioctl.yaml)
-f, --factory string   Factory to list targets for
-t, --token string     API token from https://app.foundries.io/settings/tokens/
-v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl targets - Manage factory's TUF targets

Version: 0.1.1

# fioctl_teams

## fioctl teams

List teams belonging to a FoundriesFactory

```
fioctl teams [<team_name>] [flags]
```

### Options

```
  -f, --factory string   Factory to list targets for
  -h, --help             help for teams
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_users

## fioctl users

List users with access to a FoundriesFactory

```
fioctl users [<user_id>] [flags]
```

### Options

```
  -f, --factory string   Factory to list targets for
  -h, --help             help for users
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_version

## fioctl version

Show version information of this tool.

```
fioctl version [flags]
```

### Options

```
  -h, --help   help for version
```

### Options inherited from parent commands

```
  -c, --config string   config file (default is $HOME/.config/fioctl.yaml)
  -v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories

Version: 0.1.1

# fioctl_waves

## fioctl waves

Manage factory's waves

### Options

```
-f, --factory string   Factory to list targets for
-h, --help             help for waves
-t, --token string     API token from https://app.foundries.io/settings/tokens/
```

### Options inherited from parent commands

```
-c, --config string   config file (default is $HOME/.config/fioctl.yaml)
-v, --verbose         Print verbose logging
```

### SEE ALSO

- fioctl - Manage Foundries Factories
- fioctl waves cancel - Cancel a given wave by name
- fioctl waves complete - Complete a given wave by name to make it generally available
- fioctl waves init - Create a new wave from targets of a given version
- fioctl waves list - Show available waves
- fioctl waves rollout - Roll out a wave to a subset of production devices
- fioctl waves show - Show a given wave by name
- fioctl waves status - Show a status for a given wave by name

Version: 0.1.1

# fioctl_waves_cancel

## fioctl waves cancel

Cancel a given wave by name

### Synopsis

Cancel a given wave by name. Once canceled a wave is no longer available as an update source for production devices. However, those devices that has already updated to a wave version will remain on that version until a new version is rolled out.

```
fioctl waves cancel <wave> [flags]
```

### Options

```
  -h, --help   help for cancel
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl waves - Manage factory's waves

Version: 0.1.1

# fioctl_waves_complete

## fioctl waves complete

Complete a given wave by name to make it generally available

### Synopsis

Complete a given wave by name. Once complete a wave becomes generally available as an update source for all production devices. A subsequent wave might become a new source for a part of production devices again.

```
fioctl waves complete <wave> [flags]
```

### Options

```
  -h, --help   help for complete
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl waves - Manage factory's waves

Version: 0.1.1

# fioctl_waves_init

## fioctl waves init

Create a new wave from targets of a given version

### Synopsis

Create a new wave from targets of a given version.

This command only initializes a wave, but does not provision its updates to devices. Use a "fioctl wave rollout `<wave>`{=html} `<group>`{=html}" to trigger updates of this wave to a device group. Use a "fioctl wave complete `<wave>`{=html}" to update all devices (make it globally available). Use a "fioctl wave cancel `<wave>`{=html} to cancel a wave (make it no longer available).

```
fioctl waves init <wave> <version> <tag> [flags]
```

### Examples

```
Start a new wave for the target version 4 and the 'production' device tag:
$ fioctl wave init -k ~/path/to/keys/targets.only.key.tgz wave-name 4 production

Start a new wave for the target version 16 and also prune old production versions
1,2,3 and 4 in this case:
$ fioctl wave init -k ~/path/to/keys/targets.only.key.tgz wave-name 16 production
--prune 1,2,3,4
```

### Options

```
  -d, --dry-run             Don't create a wave, print it to standard output.
  -E, --expires-at string   Role expiration date and time in RFC 3339 format.
                            The same expiration will be used for production
targets when a wave is complete.
                            When set this value overrides an 'expires-days'
argument.
```

```
                            Example: 2020-01-01T00:00:00Z
  -e, --expires-days int    Role expiration in days; default 365.
                            The same expiration will be used for production
targets when a wave is complete.
  -h, --help                help for init
  -k, --keys string         Path to <offline-creds.tgz> used to sign wave targets.
      --prune strings       Prune old unused Target(s) from the production
metadata.
                            Example: 1,2,3
      --source-tag string   Match this tag when looking for target versions.
Certain advanced tagging configurations may require this argument.
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl waves - Manage factory's waves

Version: 0.1.1

# fioctl_waves_list

## fioctl waves list

Show available waves

```
fioctl waves list [flags]
```

### Options

```
  -h, --help         help for list
  -n, --limit uint   Limit the number of results displayed. (default 20)
  -p, --page int     Page of waves to display when pagination is needed (default
1)
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl waves - Manage factory's waves

Version: 0.1.1

# fioctl_waves_rollout

## fioctl waves rollout

Roll out a wave to a subset of production devices

### Synopsis

Roll out a wave to a subset of production devices matching a wave's tag. Upon rollout a wave becomes available as an update source for a given subset of production devices. A rollout is not instant, rather each device will update to the wave's targets at some point. The exact update time is determined by many factors: device up and down lifecycle, its update schedule, networking between a device and update servers, etc. At least one command flag is required to limit the subset of devices to roll out to. If you want to roll out to all matching devices in a factory, please, use the "complete" command.

```
fioctl waves rollout <wave> [flags]
```

### Examples

```
Rollout a wave to all devices in the "us-east" device group:
$ fioctl waves rollout --group us-east

Rollout a wave to 10 devices in the "us-east" device group:
$ fioctl waves rollout --group us-east --limit=10

Rollout a wave to 2 specific devices in the "us-east" device group:
$ fioctl waves rollout --group us-east --uuids=uuid1,uuid2

Rollout a wave to 10% devices in your factory:
$ fioctl waves rollout --limit=10%

Rollout a wave to specific devices in your factory, device UUIDs provided by a
file:
$ fioctl waves rollout --uuids=@/path/to/file

Rollout a wave to 10% of specific devices in your factory, device UUIDs provided
```

```
by a file:
$ fioctl waves rollout --uuids=@/path/to/file --limit=10%

In all of the above examples:
- When using the "uuids" flag, each device in a list is verified to match wave
requirements.
  In addition, if the "group" flag is provided, each device must also belong to
that device group.
- When using the "limit" flag, a list of rolled out devices is auto-selected by
the API.
  The most recently active devices have a higher chance to get into this
selection.
  A device is excluded from the selection, if a wave was already rolled out to it
ealier.
- Using both "uuids" and "limit" flags constrains auto-selection to a given device
list.
  This can be combined with the "group" flag to further constrain it to a given
device group.
- The following characters are supported as a separator for the device list in the
"uuids" flag:
  a comma (","), a semicolon (";"), a pipe ("|"), white space, tabs, and line
breaks.
  The user is responsible for properly escaping these characters in a shell
script.
  It is recommended to pass a list of UUIDs via a file if their number is big
enough.
```

## Options

```
      --dry-run        Only show what would happen without an actual rollout. Most
useful with --print-xxx flags.
  -g, --group string   A device group to roll out a wave to
  -h, --help           help for rollout
  -l, --limit string   A number of devices to roll out a wave to.
                       It can be an exact number (e.g. 10), or as a percentage of
all matching devices (e.g. 10%).
                       An actual number of rolled out devices can be less than the
specified value.
                       A maximum number of devices rolled out using this flag
cannot exceed 10000.
      --print-names    Print names of devices to which a wave was rolled out
(would be rolled out with --dry-run).
      --print-uuids    Print UUIDs of devices to which a wave was rolled out
```

```
(would be rolled out with --dry-run).
      --uuids string   A comma-separated list of exact device UUIDs to roll out a
wave to.
                       Also accepts a filename containing a comma-separated list
via "--uuids=@path/to/file.name".
                       A maximum number of devices rolled out using this flag
cannot exceed 10000.
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl waves - Manage factory's waves

Version: 0.1.1

# fioctl_waves_show

## fioctl waves show

Show a given wave by name

```
fioctl waves show <wave> [flags]
```

### Options

```
  -h, --help           help for show
  -s, --show-targets   Show wave targets
```

### Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

### SEE ALSO

- fioctl waves - Manage factory's waves

Version: 0.1.1

# fioctl_waves_status

## fioctl waves status

Show a status for a given wave by name

### Synopsis

Show a status for a given wave by name. When no wave name is provided - show a status for a currently active wave.

For an active wave this command shows an overview of a wave, followed by an overview of device groups participating in a wave, and after that a detailed information for each rollout group.

For finished waves a detailed per group information is not shown as it is no more relevant.

When counting a total number of devices participating in a wave, each production device that has a tag equal to a wave tag counts. In particular, devices outside rollout groups also count if they satisfy this condition.

All other numbers are calculated relative to this total number. For example, online devices in each group are counted among only those production devices, that belong to a given group and also have a tag equal to a group tag. This number can be lower than a total number of online devices in this group.

In a device group overview all wave rollout groups are shown first in an order of rollout time. After that follow other groups that have devices with matching tag (if they contain at least one such device). The last row is for devices not belonging to any group (if at least one such device matches a wave tag).

A number of updated devices depends onto a wave status: For active wave it is a number of devices in rollout groups with target version >= wave version. For finished waves it is a number of all devices with target version >= wave version.

Meaning of scheduled vs unscheduled (for update) device number also depends onto a wave status: For active wave, scheduled for update are devices in rollout groups with target version < wave version. For complete wave, scheduled are all devices (regardless a group) with target version < wave version. For canceled wave, all devices with target version < wave version are unscheduled (scheduled is always zero).

For finished waves all numbers are calculated for a current date (not a date of a wave finishing). This can be used to monitor how an update progresses after a wave has been complete.

```
fioctl waves status [<wave>] [flags]
```

## Options

```
  -h, --help                    help for status
      --offline-threshold int   Consider device 'OFFLINE' if not seen in the last
X hours (default 4)
```

## Options inherited from parent commands

```
  -c, --config string    config file (default is $HOME/.config/fioctl.yaml)
  -f, --factory string   Factory to list targets for
  -t, --token string     API token from https://app.foundries.io/settings/tokens/
  -v, --verbose          Print verbose logging
```

## SEE ALSO

- fioctl waves - Manage factory's waves