Provisioning

Contents

Introduction

This is an introduction to the Provisioning extension.

Overview

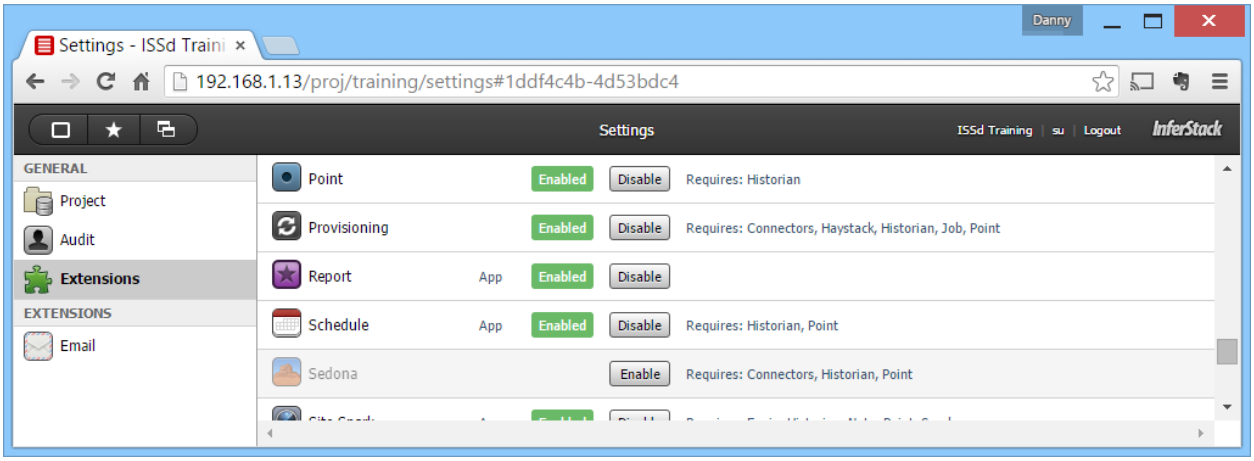
Provisioning provides tools for centrally managing groups of InferStack devices. This tool provides support for:

- Monitoring connection status
- Batch upgrading devices
- Batch patching devices

Once you enable the Provisioning extension in Settings, you will see Provisioning under Haystack in the Connectors app.

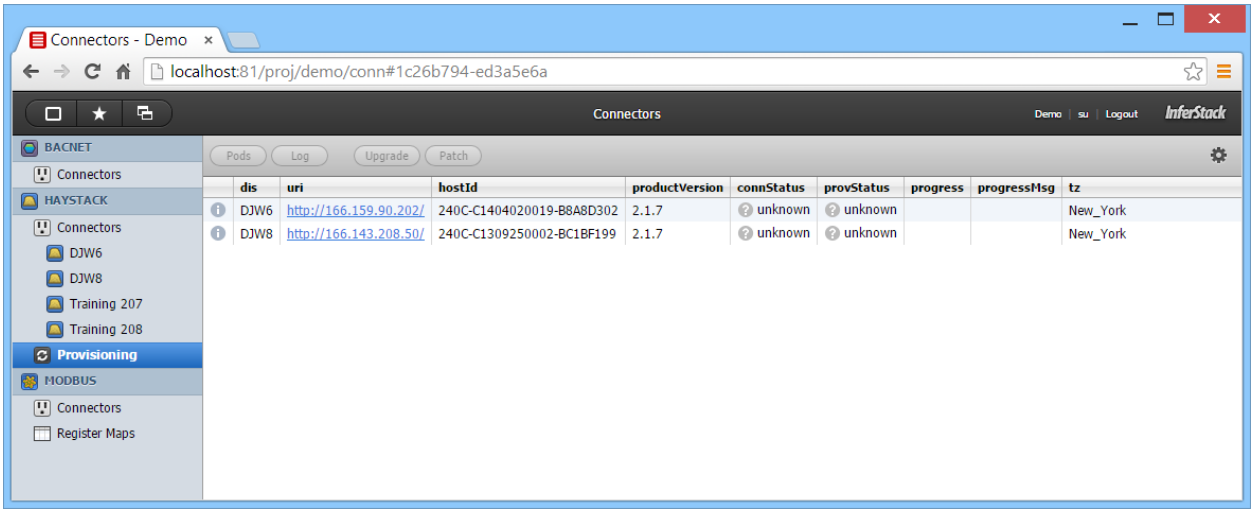
Enable the Extension

Go to Settings and enable the Provisioning extension by pressing the grayed Enabled. It will be Green when enabled.



Connectors App

The Connectors app provides access to the Provisioning extension.



Connectors

Since all InferStack devices communicate natively using Haystack, the provisioning tools are built off the Haystack connector ext. So this extension must be enabled in order to take advantage of these tools. If both the `haystackExt` and the `provExt` are enabled, you will see a new `Provisioning` tool in your `ConnApp` under the Haystack section.

Adding Devices

To add devices to your server, add them just like any other Haystack connector, either manually with the `New` command or using the `Discover` feature. The URI for a device will be:

```
uri: `http://x.x.x.x/api/infer`
```

To monitor your network connection to a device, you can add a `connPingFreq` tag to the conn rec which will periodically ping that device:

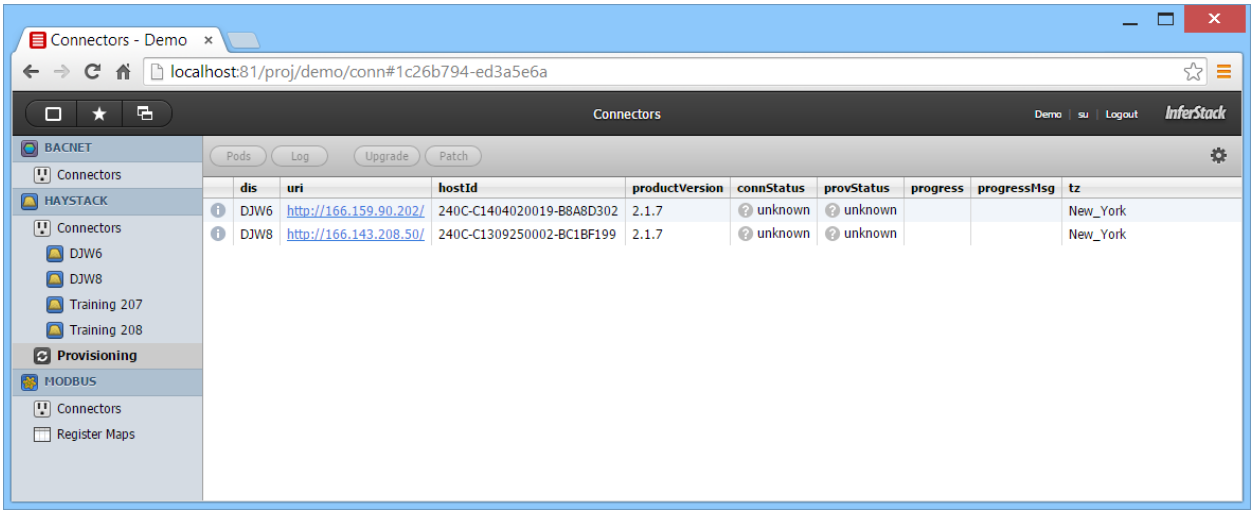
```
connPingFreq: 30sec
```

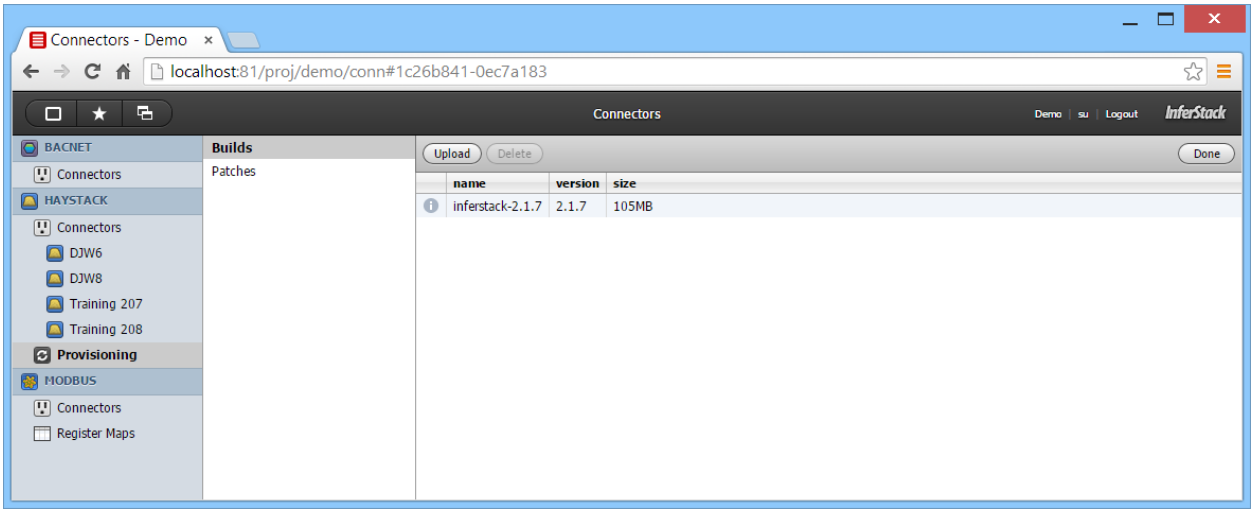
Any connectors that have a device Host ID will automatically be displayed in the Provisioning tool. Once you have pinged your device to query its Host ID it will then be available for management.

Note: You can add your hostid manually in Haystack - Connectors by selecting the connector and pressing Edit to provision older revisions.

Managing Software

The Provisioning tool maintains a database of software for managing what builds and patches are available to push to devices. You can access the software manager by pressing the "gear" icon in the top-right hand corner of the Provisioning Tool.



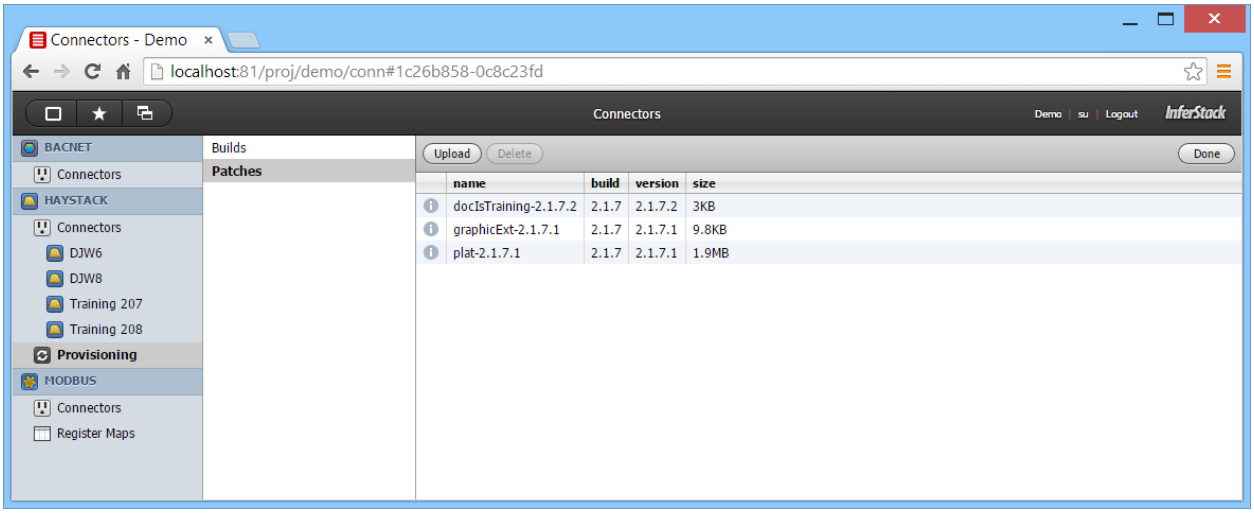


Managing Builds

To add builds for publishing, select `Builds` in the Manager, and select `Upload`. Builds are stored using the unique build version (i.e. 2.1.11). Uploading the same build will replace the existing version. To remove builds use the `Delete` command.

Managing Patches

To add patches for publishing, select `Patches` in the Manager, and select `Upload`. Patches are stored using the unique pod version (i.e. 2.1.11.2). Uploading the same patch will replace the existing version. To remove patches use the `Delete` command.



Note: If you are developing a pod, it requires a special meta tag:

```
"inferstack.build":"2.1.13"
```

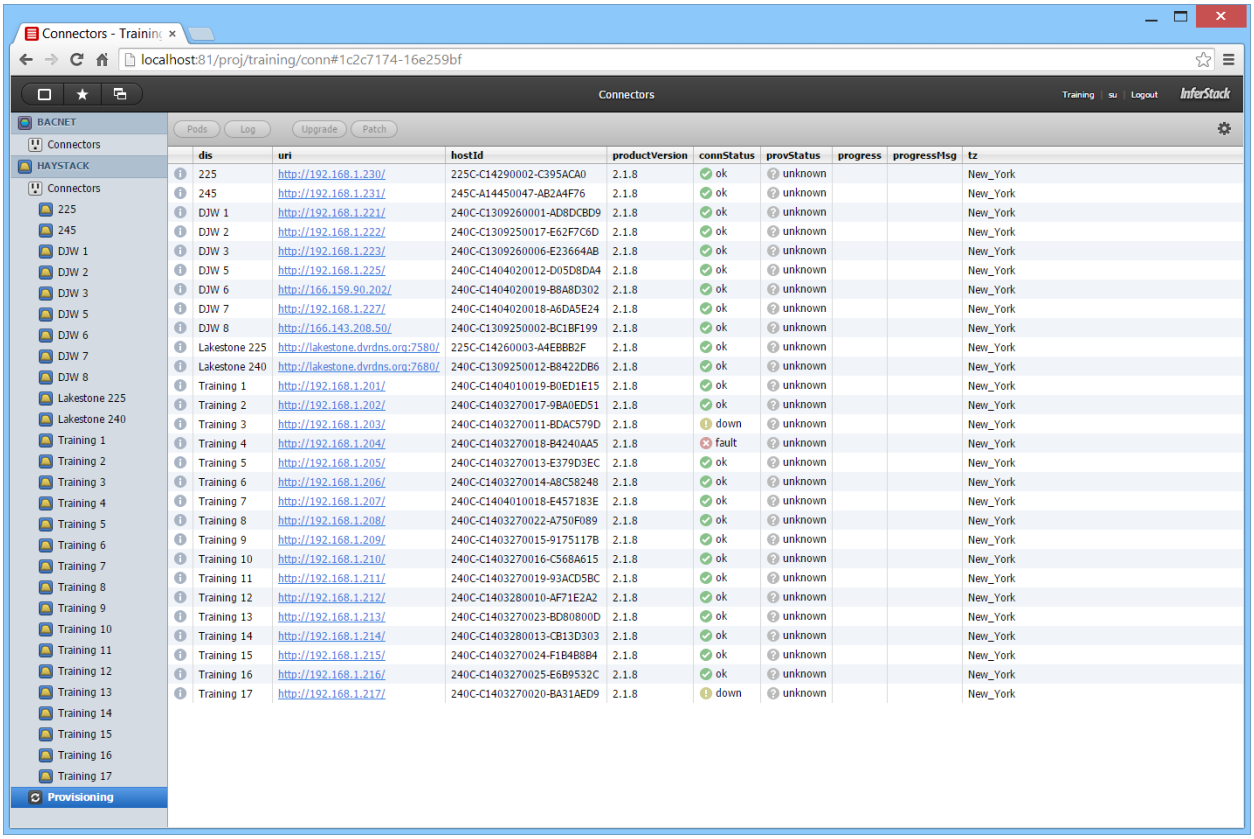
So make sure when you build your pod you have the correct `inferstack.build` for the corresponding build.

Upgrading Devices

Once you have added one or move devices, and have added software into the provisioning database, you will be able to push software updates to devices.

Installed Pods

The current build installed on each device will be displayed in the Provisioning Tool under the `productVersion` column. In order to see what pods are installed on a specific device, select the device and press the `Pods` button. This will bring up a dialog listing information on all pods currently installed on that device.



dis	uri	hostId	productVersion	connStatus	provStatus	progress	progressMsg	tz
225	http://192.168.1.230/	225C-C14290002-C395ACA0	2.1.8	ok	unknown			New_York
245	http://192.168.1.231/	245C-A14450047-AB2A4F76	2.1.8	ok	unknown			New_York
DJW 1	http://192.168.1.221/	240C-C1309260001-AD8DCBD9	2.1.8	ok	unknown			New_York
DJW 2	http://192.168.1.222/	240C-C1309250017-E62F7C6D	2.1.8	ok	unknown			New_York
DJW 3	http://192.168.1.223/	240C-C1309260006-E23664AB	2.1.8	ok	unknown			New_York
DJW 5	http://192.168.1.225/	240C-C1404020012-D05D8DA4	2.1.8	ok	unknown			New_York
DJW 6	http://166.159.90.202/	240C-C1404020019-B8A8D302	2.1.8	ok	unknown			New_York
DJW 7	http://192.168.1.227/	240C-C1404020018-A6DA5E24	2.1.8	ok	unknown			New_York
DJW 8	http://166.143.208.50/	240C-C1309250002-BC1BF199	2.1.8	ok	unknown			New_York
Lakestone 225	http://lakestone.dvrdns.org:7580/	225C-C14260003-A4EBBB2F	2.1.8	ok	unknown			New_York
Lakestone 240	http://lakestone.dvrdns.org:7680/	240C-C1309250012-B8422DB6	2.1.8	ok	unknown			New_York
Training 1	http://192.168.1.201/	240C-C1404010019-B0ED1E15	2.1.8	ok	unknown			New_York
Training 2	http://192.168.1.202/	240C-C1403270017-9BA0ED51	2.1.8	ok	unknown			New_York
Training 3	http://192.168.1.203/	240C-C1403270011-BDAC579D	2.1.8	down	unknown			New_York
Training 4	http://192.168.1.204/	240C-C1403270018-B4240AA5	2.1.8	fault	unknown			New_York
Training 5	http://192.168.1.205/	240C-C1403270013-E379D3EC	2.1.8	ok	unknown			New_York
Training 6	http://192.168.1.206/	240C-C1403270014-A8C58248	2.1.8	ok	unknown			New_York
Training 7	http://192.168.1.207/	240C-C1404010018-E457183E	2.1.8	ok	unknown			New_York
Training 8	http://192.168.1.208/	240C-C1403270022-A750F089	2.1.8	ok	unknown			New_York
Training 9	http://192.168.1.209/	240C-C1403270015-9175117B	2.1.8	ok	unknown			New_York
Training 10	http://192.168.1.210/	240C-C1403270016-C568A615	2.1.8	ok	unknown			New_York
Training 11	http://192.168.1.211/	240C-C1403270019-93ACD5BC	2.1.8	ok	unknown			New_York
Training 12	http://192.168.1.212/	240C-C1403280010-AF71E2A2	2.1.8	ok	unknown			New_York
Training 13	http://192.168.1.213/	240C-C1403270023-BD80800D	2.1.8	ok	unknown			New_York
Training 14	http://192.168.1.214/	240C-C1403280013-CB13D303	2.1.8	ok	unknown			New_York
Training 15	http://192.168.1.215/	240C-C1403270024-F1B4B8B4	2.1.8	ok	unknown			New_York
Training 16	http://192.168.1.216/	240C-C1403270025-E6B9532C	2.1.8	ok	unknown			New_York
Training 17	http://192.168.1.217/	240C-C1403270020-BA31AED9	2.1.8	down	unknown			New_York

Pushing Builds

To upgrade devices to a new build:

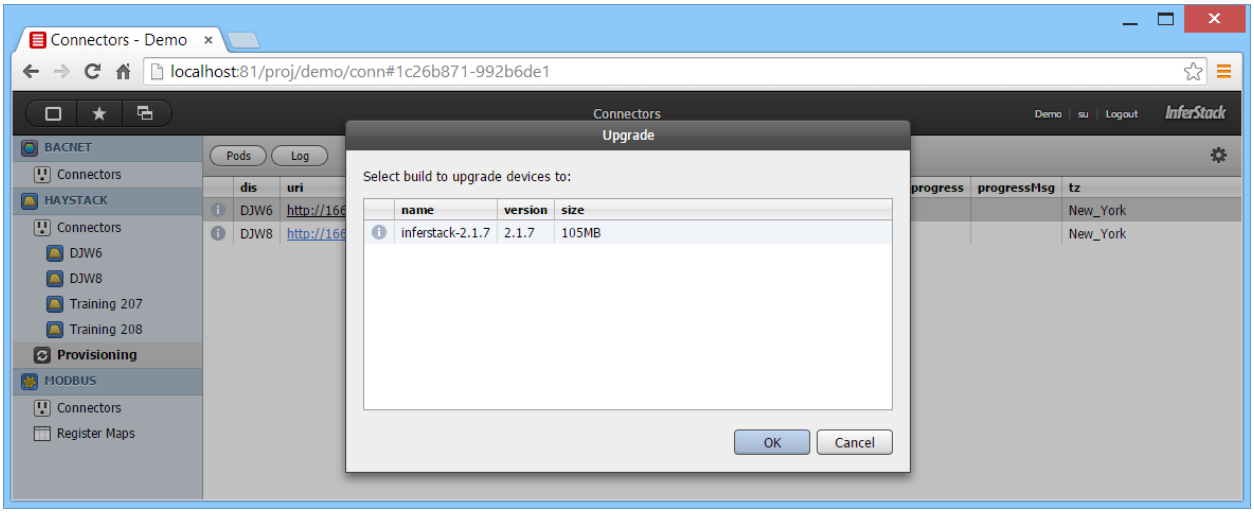
1. Select the devices you wish to update
2. Press the `Upgrade` button

3. Choose the desired build (see Managing Builds)

Note: You must delete the Guest account in the device to provision it.

After selecting your build, a background job will be kicked off for each device to perform the upgrade. You can monitor the progress in the tool table:

1. The build image will be uploaded to the device
2. The device will be rebooted to complete installation
3. The server will verify device has booted back up properly



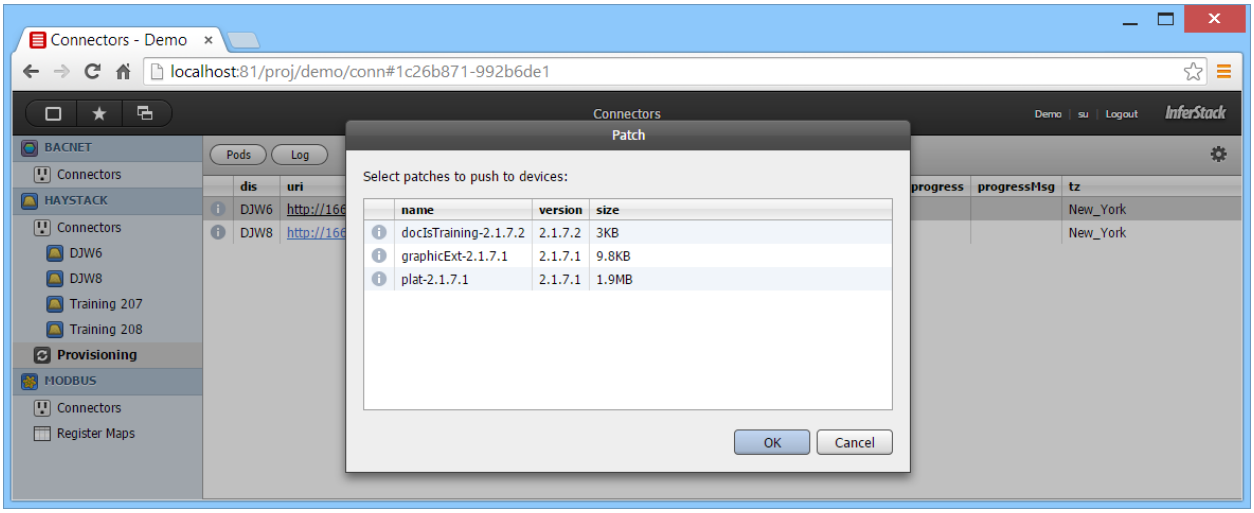
Pushing Patches

To patch devices:

1. Select the devices you wish to patch
2. Press the `Patch` button
3. Choose the desired patches available for the installed build (see Managing Patches)

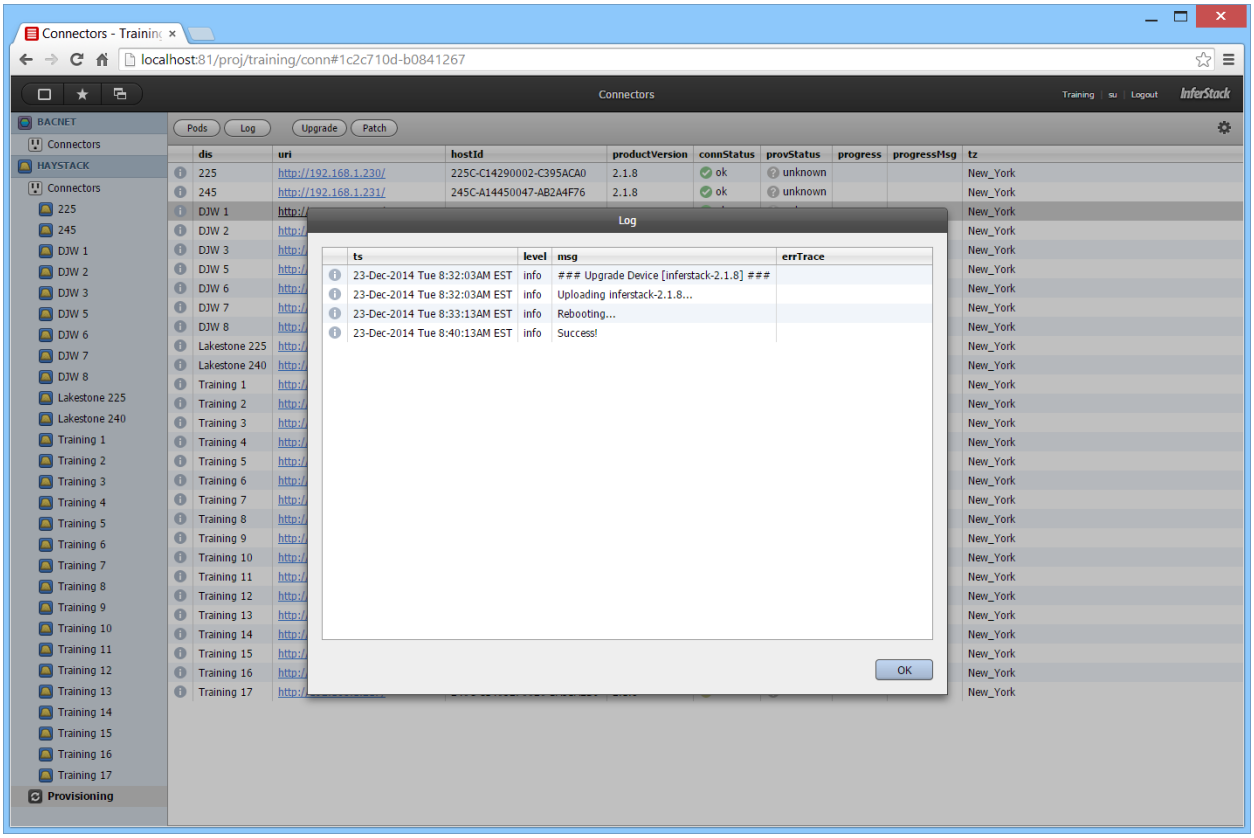
Note that when patching multiple devices, all selected devices must be running the same build version. Once you've selected the patches to install, a background job will be kicked for each device. You can monitor the progress in the tool table:

1. Each patch will be uploaded to the device
2. The device will be rebooted to complete installation
3. The server will verify device has booted back up properly



Provision Logs

Each device maintains a log history of actions performed on it. To view a device log, select a device and press the `Log` button.



Scripting

When managing large pools of devices, it is often desirable to use custom scripts which can automate updating builds, installing specific pods, and creating project configuration.

Provision scripts are created as normal funcs in the FuncApp which take a single `session` argument, and should be tagged with the `provFunc` marker tag. They can then be invoked on a selection of devices using the `Run` button.

The following funcs are available funcs inside a provision script:

- `provUpgrade`: upgrade device build
- `provPatch`: apply patches or install new pods to device
- `provEval`: evaluate an axon expr on device project
- `provEvalAll`: evaluate a list of axon exprs on a device project
- `provCommit`: commit changes to device project
- `provReboot`: remotely reboot device

The sample script below can be used as a starting point:

```
// Sample provision script
(session) => do

  // first update and patch devices
  provUpdate(session, "inferstack-2.1.11")
  provPatch(session, ["modbusExt-2.1.11.1"])

  // setup site
  site: provCommit(session, { dis:"Change Me", site, tz:"New_York" })

  // setup hostMeta to enable device-level alarming
  info: provEval(session, "hostInfo()")
  deviceEquip: projCommit(session, { dis:info->hostModel, equip,
siteRef:site->id })
  provCommit(session, { hostMeta, equipRef:deviceEquip->id })

 end
```

See the Prov Extension documentation for more information http://licensing.intellastar.com/doc/ext-prov/index .

Reference

The reference information is available in the Help app on your device or server.

Online docs are available at http://licensing.intellastar.com/doc/ .