IOx の小さい高山 Linux Docker イメージを設定して下さい

目次

概要

この資料は Docker ベースのアプリケーション IOx 可能なデバイスを on Cisco 作成し、配置し、管理するためにコンフィギュレーションプロセスを説明したものです。

前提条件

要件

このドキュメントに関する固有の要件はありません。

使用するコンポーネント

このドキュメントの情報は、次のソフトウェアとハードウェアのバージョンに基づくものです。

- IOx のために設定される IOx 可能なデバイス:
  設定される IP アドレスゲスト オペレーティング システム（GOS）および Ciscoアプリケーション フレームワーク（CAF）実行CAF （8443） ポートにアクセスのために設定されるネットワーク アドレス変換（NAT）GOS シェル（2222） ポートにアクセスのために設定される NAT
- Linux ホスト（CentOS 最小 7 つはこの技術情報のためにインストール使用されます）
- ダウンロードすることができる IOx クライアント インストール ファイル
  : https://software.cisco.com/download/release.html?mdfid=286306005&softwareid=286306762

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されたものです。 このドキュメントで使用するすべてのデバイスは、クリアな（デフォルト）設定で作業を開始しています。 ネットワークが稼働中の場合は、コマンドが及ぼす潜在的な影響を十分に理解しておく必要があります。

背景説明

IOx はパッケージ主に Java、大蛇、LXC、Virtual Machine （ VM ）等の異なる型をホストまた Docker コンテナーを実行できます。 Cisco はベースイメージおよび完全な Docker ハブ リポジトリを提供します: Docker コンテナーを構築するのに使用できる https://devhub.cisco.com/artifactory/webapp/#/artifacts/browse/tree/General/iox-docker。

方法の詳細なガイドは高山 Linux の使用の簡単な Docker コンテナーを構築するここにあります。 Docker コンテナーのためのベースとして頻繁に使用される高山 Linux は小さい Linux イメージ（ 5MB のまわりで ）です。 この技術情報では、設定された IOx デバイスから開始します、空 CentOS 7 Linux マシンおよび小さい大蛇 Webサーバを構築し、Docker コンテナーで実装し、IOx デバイスでそれを展開します。

設定

1. Linux ホストの IOx クライアントをインストールし、準備して下さい。

IOx クライアントはアプリケーションを実装し、IOx 可能なデバイスと IOx アプリケーションを管理するために通信できるツールです。

ioxclient インストール パッケージをダウンロードした後、次の通りインストールすることができます:

```
[jedepuyd@db ~]$ ll ioxclient_1.3.0.0_linux_amd64.tar.gz
-rw-r--r--. 1 jedepuyd jedepuyd 4668259 Jun 22 09:19 ioxclient_1.3.0.0_linux_amd64.tar.gz

[jedepuyd@db ~]$ tar -xvzf ioxclient_1.3.0.0_linux_amd64.tar.gz
ioxclient_1.3.0.0_linux_amd64/ioxclient
ioxclient_1.3.0.0_linux_amd64/README.md

[jedepuyd@db ~]$ ./ioxclient_1.3.0.0_linux_amd64/ioxclient --version
Config file not found : /home/jedepuyd/.ioxclientcfg.yaml
Creating one time configuration..
Your / your organization's name : Cisco
Your / your organization's URL : www.cisco.com
Your IOx platform's IP address[127.0.0.1] : 10.48.43.197
Your IOx platform's port number[8443] :
Authorized user name[root] : admin
Password for admin :
Local repository path on IOx platform[/software/downloads]:
URL Scheme (http/https) [https]:
API Prefix[/iox/api/v2/hosting/]:
Your IOx platform's SSH Port[2222]:
Activating Profile default
Saving current configuration
ioxclient version 1.3.0.0

[jedepuyd@db ~]$ ./ioxclient_1.3.0.0_linux_amd64/ioxclient --version
ioxclient version 1.3.0.0
```

IOx クライアントと管理できる IOx デバイスのために IOx クライアントの最初の起動でプロファイル生成することができる見てわかるように。 この以降をすることを望んだらまたは/変更追加するために設定ほしいと思えば、このコマンド 以降を実行できます: **ioxclient プロファイルは作成します**

2. Linux ホストの Docker をインストールし、準備して下さい。

Docker がコンテナーを構築し、サンプル アプリケーションの実行をテストするのに使用されています。

Docker をインストールするインストール手順はそれをインストールする Linux OS によって重く異なります。 この技術情報の場合、CentOS 7.を使用できます。 異なる分配のインストール指示に関しては、以下を参照して下さい: https://docs.docker.com/engine/installation/。

インストール前提条件:

```
[jedepuyd@db ~]$ sudo yum install -y yum-utils device-mapper-persistent-data lvm2
...
Complete!
```

Docker レポを追加して下さい:

```
[jedepuyd@db ~]$ sudo yum-config-manager --add-repo
https://download.docker.com/linux/centos/docker-ce.repo
Loaded plugins: fastestmirror
adding repo from: https://download.docker.com/linux/centos/docker-ce.repo
grabbing file https://download.docker.com/linux/centos/docker-ce.repo to
/etc/yum.repos.d/docker-ce.repo
repo saved to /etc/yum.repos.d/docker-ce.repo
```

インストール時) インストール Docker （ GPG キー 確認を受け入れて下さい:

```
[jedepuyd@db ~]$ sudo yum install docker-ce
...
Complete!
```

開始する Docker:

```
[jedepuyd@db ~]$ sudo systemctl start docker[jedepuyd@db iox_docker_pythonweb]$ vi Dockerfile
[jedepuyd@db iox_docker_pythonweb]$ cat Dockerfile
FROM alpine:3.3

RUN apk add --no-cache python
COPY webserver.py /webserver.py
```

規則的なユーザとしてアクセスこと/実行された Docker できるためにこのユーザを Docker グループに追加し、団体会員をリフレッシュして下さい:

```
[jedepuyd@db ~]$ sudo usermod -a -G docker jedepuyd
[jedepuyd@db ~]$ newgrp docker
```

Docker ハブへの口グイン:

Docker ハブは使用できる高山ベースイメージが含まれています。 Docker ID がまだなければ、登録する必要があります: https://hub.docker.com/。

```
[jedepuyd@db ~]$ docker login
Log in with your Docker ID to push and pull images from Docker Hub. If you do not have a Docker
ID, head over to https://hub.docker.com to create one.
Username: jensdepuydt
Password:
Login Succeeded
```

3. 大蛇 Webサーバを作成して下さい。

準備が行われるので、IOx イネーブル デバイスで動作できる実際のアプリケーションを構築し始めることができます。

```
[jedepuyd@db ~]$ mkdir iox_docker_pythonweb
[jedepuyd@db ~]$ cd iox_docker_pythonweb/
[jedepuyd@db iox_docker_pythonweb]$ vi webserver.py
[jedepuyd@db iox_docker_pythonweb]$ cat webserver.py
#!/usr/bin/env python
from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer
```

```
import SocketServer
import os

class S(BaseHTTPRequestHandler):
    def _set_headers(self):
        self.send_response(200)
        self.send_header('Content-type', 'text/html')
        self.end_headers()

    def do_GET(self):
        self._set_headers()
        self.wfile.write("<html><body><h1>IOX python webserver</h1></body></html>")

def run(server_class=HTTPServer, handler_class=S, port=80):
    server_address = ('', port)
    httpd = server_class(server_address, handler_class)
    print 'Starting webserver...'
    log_file_dir = os.getenv("CAF_APP_LOG_DIR", "/tmp")
    log_file_path = os.path.join(log_file_dir, "webserver.log")
    logf = open(log_file_path, 'w')
    logf.write('Starting webserver....\n')
    logf.close()

    httpd.serve_forever()

if __name__ == "__main__":
    from sys import argv

    if len(argv) == 2:
        run(port=int(argv[1]))
    else:
        run()
```

このコードは webserver.py で作成する非常に最小大蛇 Webサーバです。 Webサーバは GET が要求されるとすぐ IOx 大蛇 Webサーバを単に戻します。 Webサーバが開始するポートはポート 80 または webserver.py に与えられる最初の引数のどれである場合もあります。

このコードはまた、実行機能で、ログファイルへの書が含まれています。 ログファイルは IOx クライアントまたは地域管理者からの相談に利用できます。

4. Dockerfile および Docker コンテナーを作成して下さい。

コンテナーで動作する必要があるアプリケーション（webserver.py）があるのでそれは Docker コンテナーを構築する時間です。 コンテナーは Dockerfile で定義されます:

```
[jedepuyd@db iox_docker_pythonweb]$ vi Dockerfile
[jedepuyd@db iox_docker_pythonweb]$ cat Dockerfile
FROM alpine:3.3

RUN apk add --no-cache python
COPY webserver.py /webserver.py
```

見てわかるように、Dockerfile はまた簡素化されます。 高山ベースイメージから開始し、大蛇をインストールし、コンテナーのルートに webserver.py をコピーします。

準備ができた Dockerfile があれば Docker コンテナーを構築できます:

```
jedepuyd@db iox_docker_pythonweb]$ docker build -t ioxpythonweb:1.0 .
Sending build context to Docker daemon 3.584 kB
Step 1/3 : FROM alpine:3.3
3.3: Pulling from library/alpine
10462c29356c: Pull complete
```

```
Digest: sha256:9825fd1a7e8d5feb52a2f7b40c9c4653d477b797f9ddc05b9c2bc043016d4819
Status: Downloaded newer image for alpine:3.3
 ---> 461b3f7c318a
Step 2/3 : RUN apk add --no-cache python
 ---> Running in b057a8183250
fetch http://dl-cdn.alpinelinux.org/alpine/v3.3/main/x86_64/APKINDEX.tar.gz
fetch http://dl-cdn.alpinelinux.org/alpine/v3.3/community/x86_64/APKINDEX.tar.gz
(1/10) Installing libbz2 (1.0.6-r4)
(2/10) Installing expat (2.1.1-r1)
(3/10) Installing libffi (3.2.1-r2)
(4/10) Installing gdbm (1.11-r1)
(5/10) Installing ncurses-terminfo-base (6.0-r6)
(6/10) Installing ncurses-terminfo (6.0-r6)
(7/10) Installing ncurses-libs (6.0-r6)
(8/10) Installing readline (6.3.008-r4)
(9/10) Installing sqlite-libs (3.9.2-r0)
(10/10) Installing python (2.7.12-r0)
Executing busybox-1.24.2-r1.trigger
OK: 51 MiB in 21 packages
 ---> 81e98c806ee9
Removing intermediate container b057a8183250
Step 3/3 : COPY webserver.py /webserver.py
 ---> c9b7474b12b2
Removing intermediate container 4705922100e6
Successfully built c9b7474b12b2

[jedepuyd@db iox_docker_pythonweb]$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ioxpythonweb        1.0                 c9b7474b12b2        11 seconds ago      43.4 MB
alpine              3.3                 461b3f7c318a        2 days ago          4.81 MB
```

Docker ビルド コマンドはベースイメージをダウンロードし、大蛇をインストールし、あなたとして依存関係は、Dockerfile で要求しました。 最後のコマンドは確認のためです。

5. 作成された Docker コンテナーをテストして下さい。

このステップはオプションですが、ちょうど構築された Docker コンテナーが予想通りはたらいて準備ができていることを確認することはよいです。

```
[jedepuyd@db iox_docker_pythonweb]$ docker run -ti ioxpythonweb:1.0
/ # python /webserver.py 9000 &
/ # Starting webserver...

/ # netstat -tlpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:9000            0.0.0.0:*               LISTEN      7/python
/ # exit
```

netstat の出力を見てわかるように webserver.py を開始した後、ポート 9000 で受信します。

6. Docker コンテナーで IOx パッケージを作成して下さい。

コンテナーの Webサーバの機能性を確認したので、それは配備の IOx パッケージを準備をし、構築する時間です。 Dockerfile が Docker コンテナーを構築する手順を提供すると同時に IOx パッケージを構築する package.yaml は IOx クライアントに手順を提供します。

```
jedepuyd@db iox_docker_pythonweb]$ vi package.yaml
[jedepuyd@db iox_docker_pythonweb]$ cat package.yaml
descriptor-schema-version: "2.2"

info:
```

```
  name: "iox_docker_pythonweb"
  description: "simple docker python webserver on port 9000"
  version: "1.0"
  author-link: "http://www.cisco.com"
  author-name: "Jens Depuydt"

app:
  cpuarch: "x86_64"
  type: docker
  resources:
    profile: c1.small
    network:
      -
        interface-name: eth0
        ports:
            tcp: [9000]

  startup:
    rootfs: rootfs.tar
    target: ["python","/webserver.py","9000"]
```

package.yaml のコンテンツに関する詳細はここに見つけることができます: https://developer.cisco.com/media/iox-dev-guide-3-10-16/concepts/package_descriptor/。

package.yaml を作成した後、IOx パッケージを構築し始めることができます。

第一歩は Docker イメージのルート FS をエクスポートすることです:

```
[jedepuyd@db iox_docker_pythonweb]$ docker save -o rootfs.tar ioxpythonweb:1.0
```

次に、package.tar を構築できます:

```
[jedepuyd@db iox_docker_pythonweb]$ ../ioxclient_1.3.0.0_linux_amd64/ioxclient package .
Currently active profile:  default
Command Name: package
Checking if package descriptor file is present.
Validating descriptor file /home/jedepuyd/iox_docker_pythonweb/package.yaml with package schema
definitions
Parsing descriptor file.
Found schema version  2.2
Loading schema file for version  2.2
Validating package descriptor file..
File /home/jedepuyd/iox_docker_pythonweb/package.yaml is valid under schema version 2.2
Created Staging directory at :  /tmp/700740789
Copying contents to staging directory
Checking for application runtime type
Couldn't detect application runtime type
Creating an inner envelope for application artifacts
Generated  /tmp/700740789/artifacts.tar.gz
Calculating SHA1 checksum for package contents..
Parsing Package Metadata file :  /tmp/700740789/.package.metadata
Wrote package metadata file :  /tmp/700740789/.package.metadata
Root Directory :  /tmp/700740789
Output file:  /tmp/335805072
Path:  .package.metadata
SHA1 : 55614e72481a64726914b89801a3276a855c728a
Path:  artifacts.tar.gz
SHA1 : 816c7bbfd8ae76af451642e652bad5cf9592370c
Path:  package.yaml
SHA1 : ae75859909f6ea6947f599fd77a3f8f04fda0709
Generated package manifest at  package.mf
Generating IOx Package..
Package generated at /home/jedepuyd/iox_docker_pythonweb/package.tar
```

ビルドの結果は Docker コンテナーが含まれている IOx で展開されること準備ができた IOx パッケージ（package.tar）、です。

> **注:** IOxclient は 1 つのステップの docker save コマンドをまたすることができます。 CentOS で、これは rootfs.tar の代りにプロセスのトラブル以降を与えるデフォルト rootfs.img に工クスポートするために生じます。 作成するべき 1 つのステップは使用との実行することができます: IOx クライアント docker パッケージ IOxpythonweb:1.0。

8. 導入は、アクティブ化 IOx デバイスのパッケージを開始し。

最後のステップは IOx パッケージを IOx デバイスへ展開し、アクティブにし、開始することです。 これらのステップは IOx クライアント、地域管理者または霧 Network Director の使用と実行することができます。 この技術情報の場合、IOx クライアントを使用できます。

パッケージを IOx デバイスに展開するために、ネーム python_web を使用して下さい:

```
[jedepuyd@db iox_docker_pythonweb]$ ../ioxclient_1.3.0.0_linux_amd64/ioxclient app install
python_web package.tar
Currently active profile:  default
Command Name: application-install
Installation Successful. App is available at:
https://10.48.43.197:8443/iox/api/v2/hosting/apps/python_web
Successfully deployed
```

アプリケーションをアクティブにすることができる前にネットワークコンフィギュレーションがどのようにあるか定義する必要があります。 これを行うために、JSON ファイルを作成する必要があります。 アクティベーション要求にアクティブ化、それ接続することができる時。

```
[jedepuyd@db iox_docker_pythonweb]$ vi activate.json
[jedepuyd@db iox_docker_pythonweb]$ cat activate.json
{
    "resources": {
        "profile": "c1.small",
        "network": [{"interface-name": "eth0", "network-name": "iox-nat0"}]
    }
}
[jedepuyd@db iox_docker_pythonweb]$ ../ioxclient_1.3.0.0_linux_amd64/ioxclient app activate
python_web --payload activate.json
Currently active profile :  default
Command Name: application-activate
Payload file : activate.json. Will pass it as application/json in request body..
App python_web is Activated
```

ここの最後の操作はちょうど展開し、アクティブにしたアプリケーションを開始することです:

```
[jedepuyd@db iox_docker_pythonweb]$ ../ioxclient_1.3.0.0_linux_amd64/ioxclient app start
python_web
Currently active profile :  default
Command Name: application-start
App python_web is Started
```

岩山 HTTP 要求をポート 9000 で聞き取るために IOx アプリケーションを設定したのでまだ IOx デバイスからの容器として容器へのポートが NAT の後ろにあること転送する必要があります。 そうするためにこの IOS® を on Cisco 行って下さい。

```
BRU-IOT-809-1#sh iox host list det | i IPV4
    IPV4 Address of Host:       192.168.1.2
BRU-IOT-809-1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
BRU-IOT-809-1(config)#ip nat inside source static tcp 192.168.1.2 9000 interface
```

```
GigabitEthernet0 9000
BRU-IOT-809-1(config)#exit
```
最初のコマンド リスト GOS の内部 IP アドレス (スタート/ストップに/IOx コンテナーを実行して下さい責任がある) 。

第 2 コマンドは GOS に IOS 側の Gi0 インターフェイスのポート 9000 のために静的ポートを順方向に設定したものです。 デバイスが (可能性が高い IR829 のケース) L2 ポートによって接続されれば、設定される ip nat outside 文がある正しい VLAN と Gi0 インターフェイスを取替える必要があります。

確認

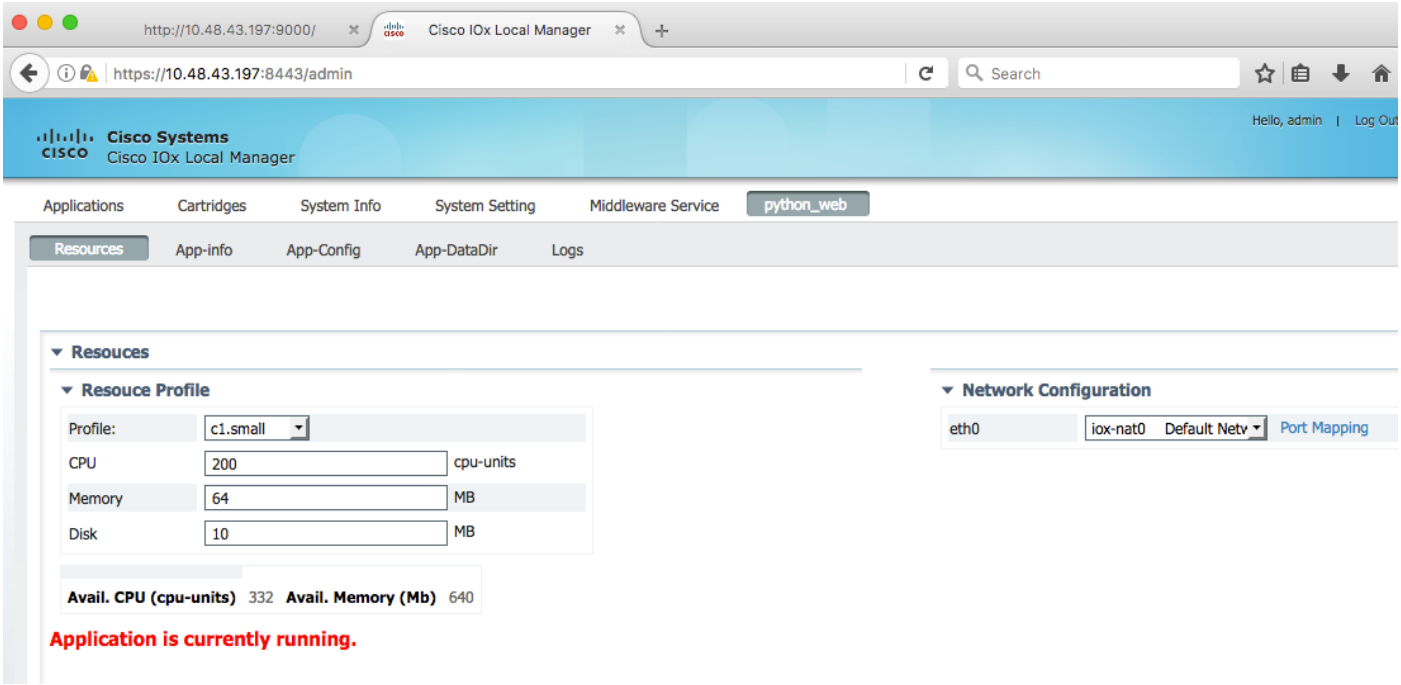
このセクションでは、設定が正常に機能していることを確認します。

Webサーバがきちんと動作し、応答するかどうか確認するために、このコマンドで Webサーバにアクセスすることを試みることができます。

```
[jedepuyd@db iox_docker_pythonweb]$ curl http://10.48.43.197:9000/
<html><body><h1>IOX python webserver</h1></body></html>
```
または、イメージに示すように実質ブラウザから。

また IOxclient CLI からのアプリケーション ステータスを確認できます:

```
[jedepuyd@db iox_docker_pythonweb]$ ../ioxclient_1.3.0.0_linux_amd64/ioxclient app status
python_web
Currently active profile :  default
Command Name: application-status
Saving current configuration
App python_web is RUNNING
```
そしてまたイメージに示すようにローカル マネージャGUI からのアプリケーション ステータスを確認できます。



webserver.py にに書くログファイルの一覧するため:

```
[jedepuyd@db iox_docker_pythonweb]$ ../ioxclient_1.3.0.0_linux_amd64/ioxclient app logs info
python_web
Currently active profile :  default
Command Name: application-logs-info

Log file information for :  python_web
Size_bytes : 711
Download_link : /admin/download/logs?filename=python_web-watchDog.log
Timestamp : Thu Jun 22 08:21:18 2017
Filename : watchDog.log

Size_bytes : 23
Download_link : /admin/download/logs?filename=python_web-webserver.log
Timestamp : Thu Jun 22 08:21:23 2017
Filename : webserver.log

Size_bytes : 2220
Download_link : /admin/download/logs?filename=python_web-container_log_python_web.log
Timestamp : Thu Jun 22 08:21:09 2017
Filename : container_log_python_web.log
```

トラブルシューティング

このセクションでは、設定のトラブルシューティングに役立つ情報を提供します。

アプリケーションやコンテナーを解決するために、動作する最も簡単な方法はアプリケーションのコンソールに接続することです:

```
[jedepuyd@db iox_docker_pythonweb]$ ../ioxclient_1.3.0.0_linux_amd64/ioxclient app console
python_web
Currently active profile:  default
Command Name: application-console
Console setup is complete..
Running command: [ssh -p 2222 -i python_web.pem appconsole@10.48.43.197]
The authenticity of host '[10.48.43.197]:2222 ([10.48.43.197]:2222)' can't be established.
ECDSA key fingerprint is 1d:e4:1e:e1:99:8b:1d:d5:ca:43:69:6a:a3:20:6d:56.
Are you sure you want to continue connecting (yes/no)? yes
```

```
/ # netstat -tlpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:9000            0.0.0.0:*               LISTEN      19/python
/ # ps aux | grep python
   19 root       0:00 python /webserver.py 9000
```