ネットワーク運用を楽にする チュートリアル（実践編）

Kaoru Kitauchi
DWANGO Co.,Ltd.

自己紹介

北内 薫 @kaorukit

ドワンゴから来ました

今日話すこと

- 運用コード化あるある
- APIとは？
- WebAPI叩いてみた

今日話すこと

- 運用コード化あるある
- APIとは？
- WebAPI叩いてみた

NW運用コード化あるある

- NW機器のconfig / showを取り込んでｺﾞﾆｮｺﾞﾆｮ
  - Ansibleで叩いたり、自分で叩いたり
- 台帳（例：Excel）を取り込んでｺﾞﾆｮｺﾞﾆｮ
  - csvにして使ったり
- 管理システム（例：Zabbix）を参照してｺﾞﾆｮｺﾞﾆｮ
  - ファイルエクスポートしたり
  - ？？？

NW運用コード化あるある

- NW機器のconfig / showを取り込んでｺﾞﾆｮｺﾞﾆｮ
  - Ansibleで叩いたり、自分で叩いたり
- 台帳（例：Excel）を取り込んでｺﾞﾆｮｺﾞﾆｮ
  - csvにして使ったり
- 管理システム（例：Zabbix）を参照してｺﾞﾆｮｺﾞﾆｮ
  - ファイルエクスポートしたり
  - **WebAPIを使ってみよう！**

今日話すこと

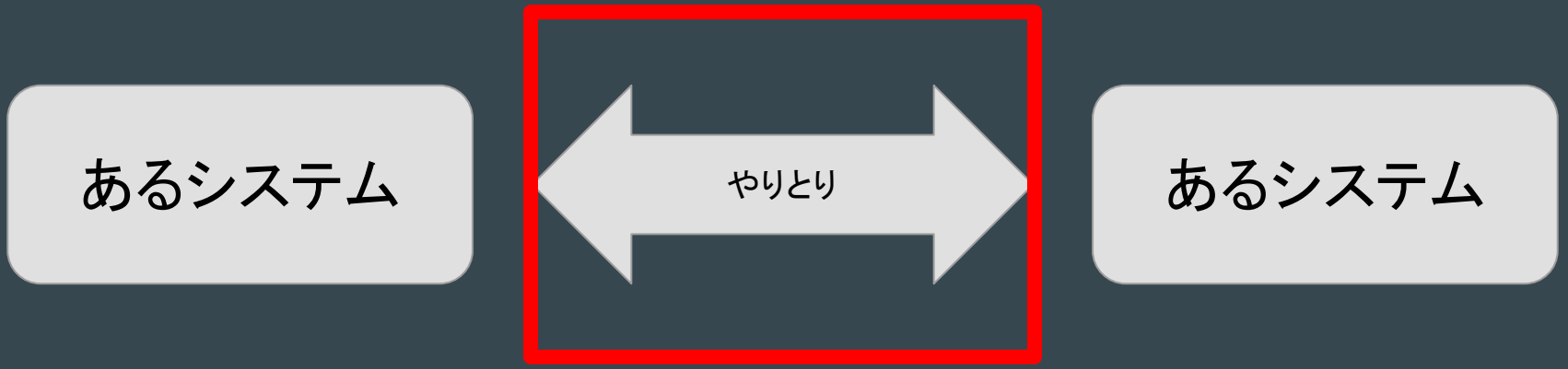
- 運用コード化あるある
- **APIとは？**
- WebAPI叩いてみた

APIとは？

Wikipedia曰く

「アプリケーションプログラミングインタフェース（API、英: Application Programming Interface）とは、広義の意味ではソフトウェアコンポーネントが互いにやりとりするのに使用するインタフェースの仕様である。」

なるほど？つまり？



WebAPIとは？

WebAPIは、大体お作法が公開されています

郵便番号検索API http://zip.cgis.biz/

仕様
リクエストメソッド
GET
リクエスト送信先URL
[XMLで受け取りたい場合] http://zip.cgis.biz/xml/zip.php
[CSVで受け取りたい場合] http://zip.cgis.biz/csv/zip.php

ためしに叩いてみる

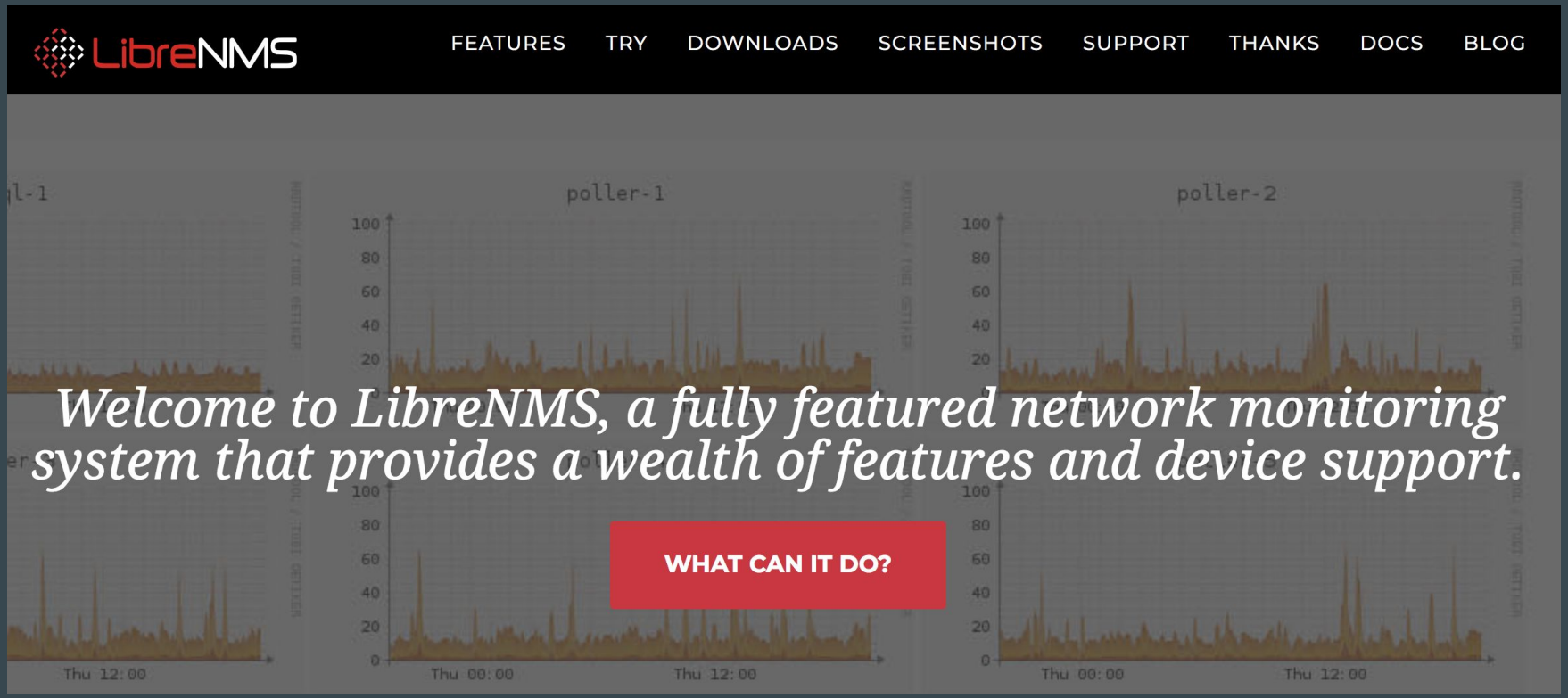
```
~ $ curl "http://zip.cgis.biz/xml/zip.php?zn=1030000"
<?xml version="1.0" encoding="utf-8" ?>
<ZIP_result>
<result name="ZipSearchXML" />
<result version="1.01" />
<result request_url="http%3A%2F%2Fzip.cgis.biz%2Fxml%2Fzip.php%3Fzn%3D1030000" />
<result request_zip_num="1030000" />
<result request_zip_version="none" />
<result result_code="1" />
<result result_zip_num="1030000" />
<result result_zip_version="0" />
<result result_values_count="1" />
<ADDRESS_value>
<value state_kana="トウキョウト" />
<value city_kana="チュウオウク" />
<value address_kana="none" />
<value company_kana="none" />
<value state="東京都" />
<value city="中央区" />
<value address="none" />
<value company="none" />
</ADDRESS_value>
</ZIP_result>
~ $
```

これは便利

今日話すこと

- 運用コード化あるある
- APIとは？
- **WebAPI叩いてみた**

本日のWebAPI


LibreNMS
FEATURES
TRY
DOWNLOADS
SCREENSHOTS
SUPPORT
THANKS
DOCS
BLOG
poller-1
poller-2
100
80
60
40
20
0
Thu 12:00
Thu 00:00
Thu 12:00
Thu 00:00
Thu 12:00
Welcome to LibreNMS, a fully featured network monitoring system that provides a wealth of features and device support.
WHAT CAN IT DO?

やりたいこと

- Tracerouteで経由した（NW機器 / Interface）を知りたい！

```
traceroute 8.8.8.8

 1  vlan100-router1

 2  vlan200-router2

 3  gi1/0/1-router3

□
```

やりたいこと

- Tracerouteで経由した（NW機器 / Interface）を知りたい！

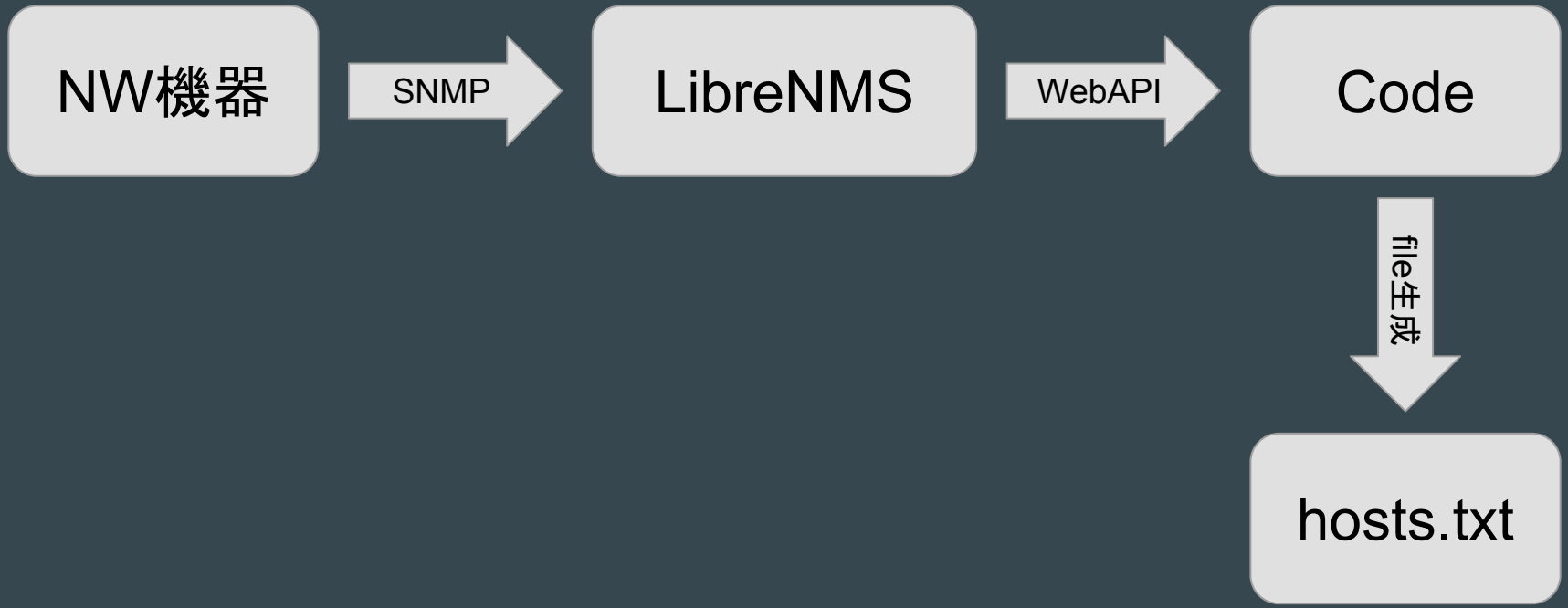
x.x.x.x vlan100-router1

y.y.y.y vlan200-router2

z.z.z.z gi1/0/1-router3

□

やりたいこと



今回必要なのは･･･

- ipアドレス
- interface名
- device名

今回必要なのは・・・

- ipアドレス
- interface名
- device名
- 上記を相互に紐付ける何か←大事！

やってみた

deviceのlistを作る

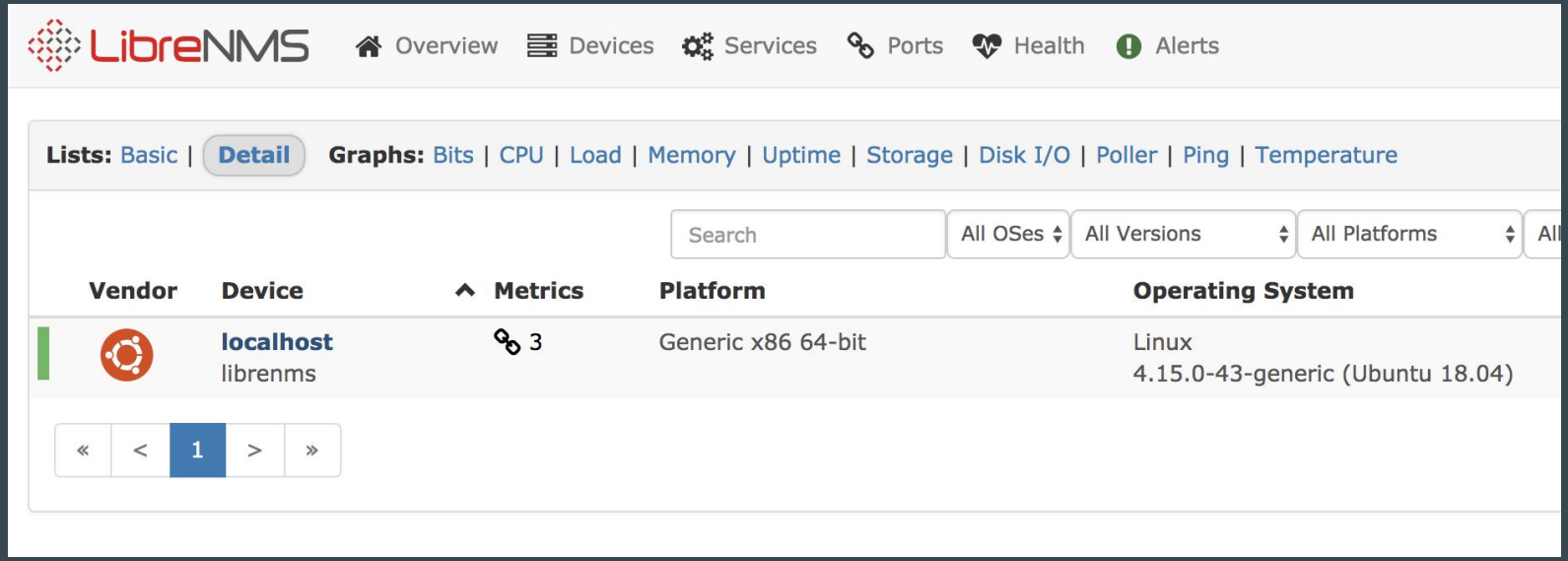
`list_devices`

Return a list of devices.

Route: `/api/v0/devices`


```
{
 "status": "ok",
 "count": 1,
 "devices": [
  {
   "device_id": "1",
   "hostname": "localhost",
   ...
   "serial": null,
   "icon": null
  }
 ]
}
```

deviceのlistを作る



deviceのlistを作る

```
headers: dict = {"X-Auth-Token": token}
devices_dict = {}

res = requests.get(f"{url}api/v0/devices", headers=headers, verify=False,)
devices_list = json.loads(res.text)["devices"]

for device in devices_list:
    devices_dict[device["device_id"]] = device["sysName"]
```

deviceのlistを作る

device_id　sysName

```
{1: 'librenms'}
```

portのlistを作る

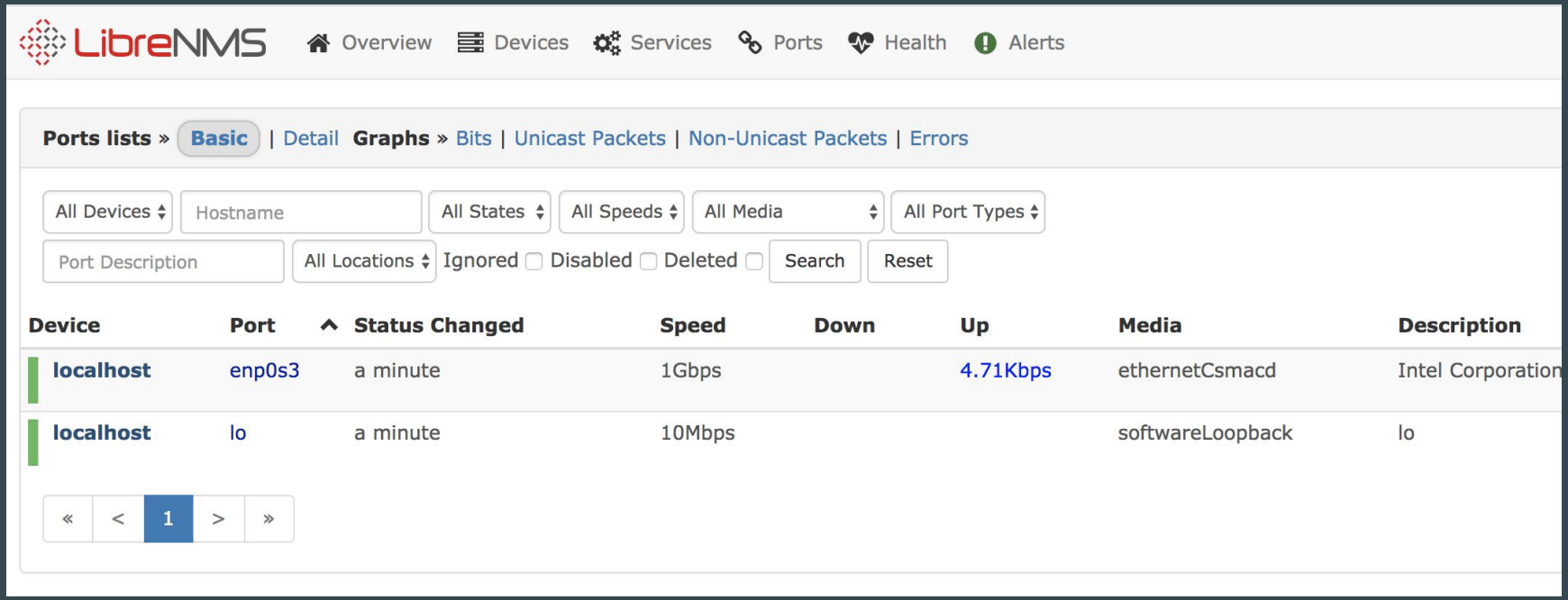
`get_all_ports`

Get info for all ports on all devices. Strongly recommend that you use the `columns` parameter to avoid pulling too much data.

Route: `/api/v0/ports`


```
{
  "status": "ok",
  "message": "",
  "ports": [
        {
          "ifName": "Gi0/0/0",
          "port_id": "1"
        },
        {
          "ifName": "Gi0/0/1",
          "port_id": "2"
        },
        ...
        {
          "ifName": "Vlan 3615",
          "port_id": "5488"
        }
    ]
}
```

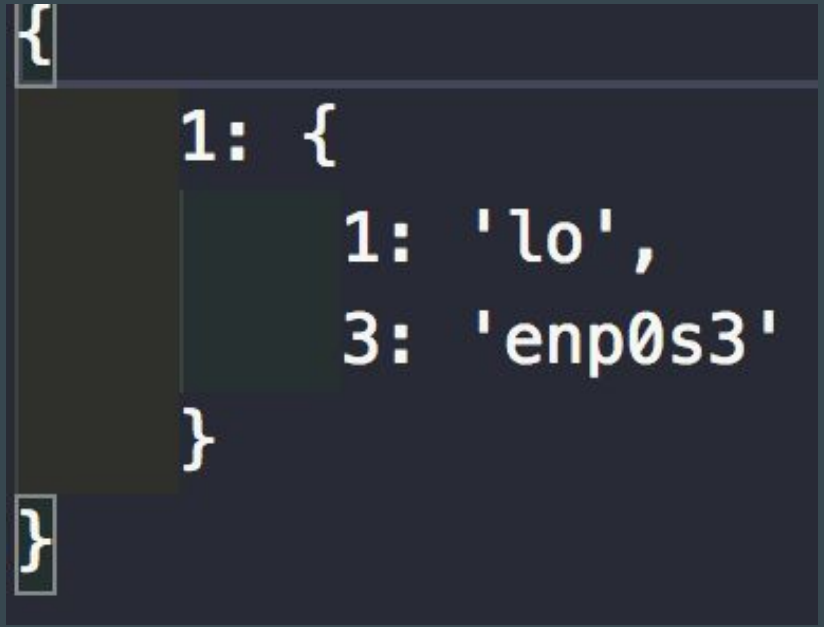
portのlistを作る



portのlistを作る

```
headers: dict = {"X-Auth-Token": token}
  You, a day ago • commit
res = ""
res = requests.get(
    f"{url}api/v0/ports?columns=ifName%2Cport_id%2Cdevice_id", headers=headers, verify=False)

ports_list = json.loads(res.text)["ports"]

if json.loads(res.text)["status"] == "ok":
    for port in ports_list:
        if ports_dict.get(port["device_id"]):
            ports_dict[port["device_id"]].update({port["port_id"]: port["ifName"]})
        else:
            ports_dict[port["device_id"]] = {port["port_id"]: port["ifName"]}
```

portのlistを作る



device_id

port_id: ifName

port_id: ifName

addressのlistを作る

`get_device_ip_addresses`

Get a list of IP addresses (v4 and v6) associated with a device.

Route: `/api/v0/devices/:hostname/ip`

- hostname can be either the device hostname or id


```
{
    "status": "ok",
    "message": "",
    "addresses": [
        {
          "ipv4_address_id": "290",
          "ipv4_address": "192.168.99.292",
          "ipv4_prefixlen": "30",
          "ipv4_network_id": "247",
          "port_id": "323",
          "context_name": ""
        }
    ]
}
```

addressのlistを作る

`get_device_ip_addresses`

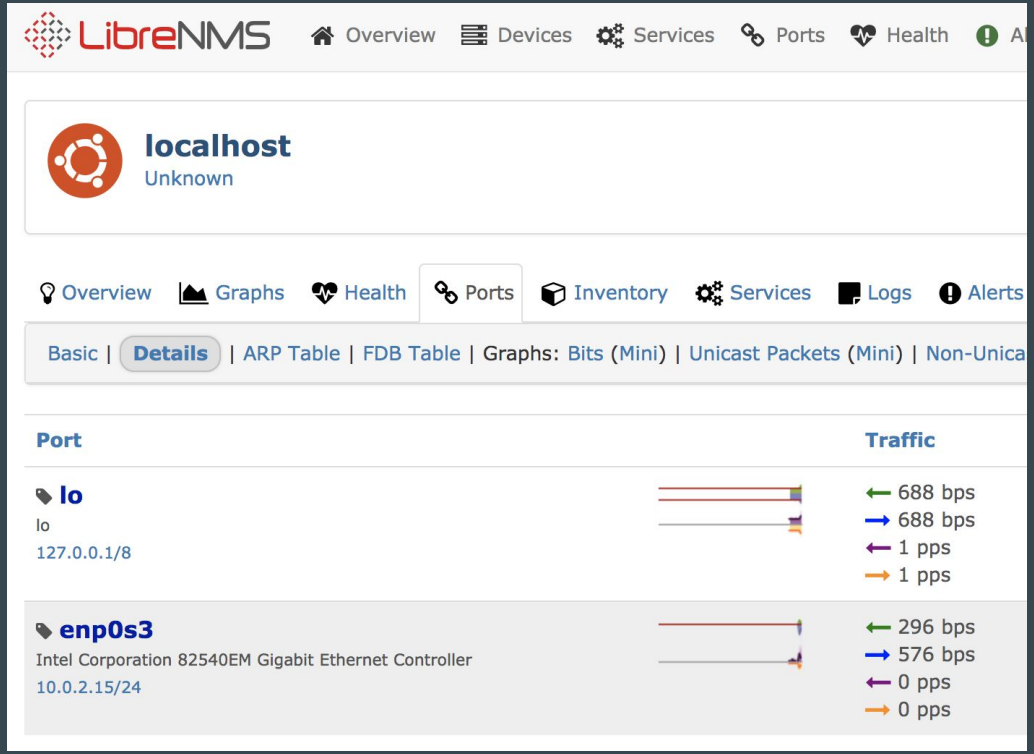
Get a list of IP addresses (v4 and v6) associated with a device.

Route: `/api/v0/devices/:hostname/ip`

- hostname can be either the device hostname or id


```
{
    "status": "ok",
    "message": "",
    "addresses": [
        {
          "ipv4_address_id": "290",
          "ipv4_address": "192.168.99.292",
          "ipv4_prefixlen": "30",
          "ipv4_network_id": "247",
          "port_id": "323",
          "context_name": ""
        }
    ]
}
```

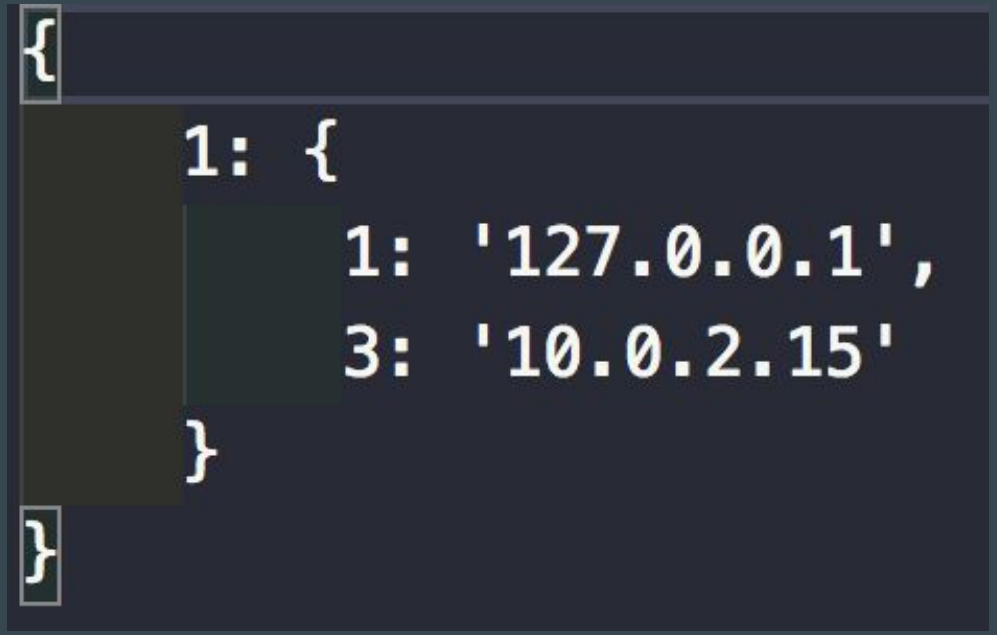
addressのlistを作る



addressのlistを作る

```
headers: dict = {"X-Auth-Token": token}

for device_id in tqdm(devices_dict.keys()):

    addresses_list: list = []
    res = ""

    res = requests.get(
        f"{url}api/v0/devices/{device_id}/ip", headers=headers, verify=False)

    if json.loads(res.text)["status"] == "ok":
        addresses_list = json.loads(res.text)["addresses"]

        for address in addresses_list:
            if addr_dict.get(device_id):
                addr_dict[device_id].update(({address["port_id"]: address["ipv4_address"]}))
            else:
                addr_dict[device_id] = {address["port_id"]: address["ipv4_address"]}
```

addressのlistを作る



device_id

port_id: ipv4_address

port_id: ipv4_address

出来上がった3つのlistを組み合わせてhosts出力

```
with open("raw_hosts.txt", "a") as raw_hosts:
    for device_id, address_dict in addr_dict.items():
        for port_id in address_dict.keys():
            raw_hosts.write(
                f"{addr_dict[device_id][port_id]} {ports_dict[device_id][port_id]}.{devices_dict[device_id]}\n")
```

```
127.0.0.1 lo.librenms
10.0.2.15 enp0s3.librenms
```

出来上がった3つのlistを組み合わせてhosts出力

```
with open("raw_hosts.txt", "a") as raw_hosts:
    for device_id, address_dict in addr_dict.items():
        for port_id in address_dict.keys():
            raw_hosts.write(
                f"{addr_dict[device_id] port_id]} {ports_dic [device_id] [port_id]}.{devices_dic [device_id] \n")
```

```
127.0.0.1 lo.librenms
10.0.2.15 enp0s3.librenms
```

もうちょっと
ｺﾞﾆｮｺﾞﾆｮしたい

自社内で使っているprefixだけ抽出したい

```
DEFINED_PREFIX = [
    "10.0.0.0/8",
    "172.16.0.0/12",
    "192.168.0.0/16"
]

hosts_list: list = []

with open(raw_hosts, "r") as raw_hosts:
    for row in raw_hosts:
        ipv4_addr = row.split()[0]

        for ipv4_nw in DEFINED_PREFIX:
            if ipaddress.IPv4Address(ipv4_addr) in ipaddress.IPv4Network(ipv4_nw):
                hosts_list.append(row)
```

禁則文字とかdotを置換したい

```
PROHIBITION_CHARACTER = r"(/|@|_|\.| |#|\$)"

hosts_list: list = []

for row in raw_hosts:

    addr = row.split()[0]
    domain = re.sub(PROHIBITION_CHARACTER, "-", row.split()[-1])

    hosts_list.append(f"{addr} {domain.lower()}\n")
```

これがこうじゃ！

```
127.0.0.1 lo.librenms
10.0.2.15 enp0s3.librenms
```

```
10.0.2.15 enp0s3-librenms
```

さいごに

- WebAPIコワクナイヨ
  - ドキュメントはちゃんと読もう
- WebAPI最強、デハナイヨ
  - 適材適所を考えよう
- NW運用コード化タノシイヨ
  - Happy Hacking Hackathon!