mitene internet

Peeringの見える化を自力でやってみた。

自己紹介

mitene internet

熊本 豊(くまもとゆたか)
@hagane5563
36歳
熊本なのに福井出身
ミテネインターネット(地方ISP)@入社14年目

主な仕事
インフラエンジニア
WebやUI/UXのレイヤー高い部分にも興味津々
ストリーミング配信歴も長いです。

趣味
焼き鳥を食べる、Perfume、BabyMetal、
スマホゲーム（パズドラ廃人）、野球観戦etc……

・それぞれのASへのトラフィック量を知りたかった。
→商用フローコレクタを検討するも予算的に厳しい。
→結果的にはお金を産むものだけど結果が見えにくい。

・オープンソースを使っていろいろやってみた。
→sfcapd+NFSEN
→fluentd＋sflowtool＋netflowPlugin＋Elasticsearch＋kibana
実現したいものがなかなかできなかった・・・

結果・・・・・

sfcapd＋nfdumpでCUI操作サイコー

使える人がほとんど居なかった・・・

今回の考え

mitene internet

・プログラムど初心者がプロットを作る。

・言語はまだ一番わかっているPHPを利用。
（ほんとはRubyとかPythonとか使ったらカッコイイ）

・各Peer先の状態をWebから確認できるようにしたい。

・Flowで各AS間のトラフィック量を調べたい。
-表示するFlowは前日のピーク時間帯（22:00-24:00）
のFlowデータを参照したい。

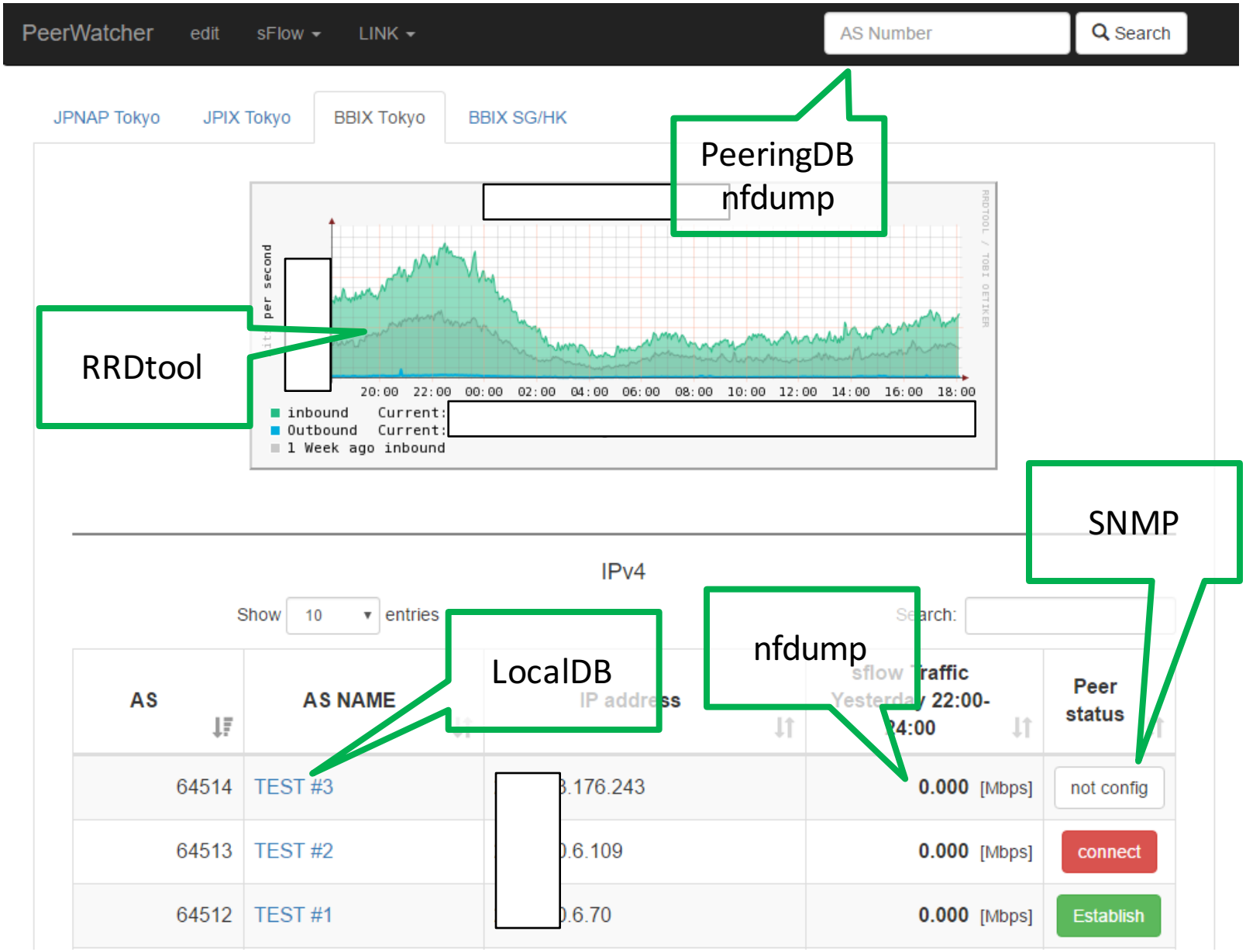
・見た目をよくしたい。
-「Bootstrap」を利用しました。（CSSのフレームワーク）
ボタン等も簡単に見栄え良く作成することができます。（後述）
-ノンデザイナーズ・デザインブックを読みました。

・PeeringDBのAPIでいろんな情報を引き出したい。

・エラー処理とかその辺の部分は何も考えずプロにお任せしたい。

つくってみました。

mitene internet



BGP ステータス

mitene internet

Establish 6

openconfirm 5

opensent 4

active 3

connect 2

idle 1

not config 返り値なし

```
<button type="button" class="btn btn-success">Establish</button><br>
<button type="button" class="btn btn-danger">openconfirm</button><br>
<button type="button" class="btn btn-danger">opensent</button><br>
<button type="button" class="btn btn-danger">active</button><br>
<button type="button" class="btn btn-danger">connect</button><br>
<button type="button" class="btn btn-danger">idle</button><br>
<button type="button" class="btn btn-default">not config</button><br>
```

PHPのsnmp関数を使ってスイッチのステータスの返り値をみて表示を変える。
ボタンはbootstrapを使っています。すごく便利。

管理画面（LocalDB）

mitene internet

IPv6の各BGPステータスの状況を取りたい。

BrocadeのMIBリファレンスをみても載ってない・・・・（気がする）

諦めかけていた所、取ったこと事がありますよという方が。

他社と同じくIPv4 の場合OID末尾が1.4.{IPv4アドレス}だったところを
IPv6の場合は2.16.{IPv6アドレスを10進にしたもの}
とすれば取れました。

登録IPv6アドレスが2001:DB8::1:7961:1の場合のOID

1)省略をなくす

2001:0DB8:0000:0000:0000:0001:7961:0001

2)2桁に区切って":"を"."にする

20.01.0D.B8.00.00.00.00.00.00.00.01.79.61.00.01

3)16進→10進へ

32.1.13.184.0.0.0.0.0.0.0.1.121.97.0.1

SNMP IPv6→OIDサンプル

mitene internet

```php
<?php
	$IPADRS = "2001:DB8::1:7961:1";
	//入力値
	print $IPADRS . "<br>¥n";

	//小文字・省略に統一する
	$IPADRS = inet_ntop ( inet_pton( $IPADRS ) );
	print $IPADRS . "<br>¥n";

	//コロンの数
	$COLON = substr_count( $IPADRS , ":" );
	print $COLON . "<br>¥n";

	//コロンの数に応じて省略をなくす
	if( $COLON === 6)
		$convertIPv6	= str_replace( "::" , ":0000:0000:" , $IPADRS );
	else if( $COLON === 5)
		$convertIPv6	= str_replace( "::" , ":0000:0000:0000:" , $IPADRS );
	else if( $COLON === 4)
		$convertIPv6	= str_replace( "::" , ":0000:0000:0000:0000:" , $IPADRS );
	print $convertIPv6 . "<br>¥n";

	//各オクテットを4桁にしてコロンをはずす
	$convertIPv6Ary = explode( ":", $convertIPv6 );
	$convertIPv6	= "";

	for( $i = 0 ; $i < count( $convertIPv6Ary ) ; ++$i ) {
		if( strlen( $convertIPv6Ary[ $i ] ) === 4 )
			$convertIPv6	= "{$convertIPv6}{$convertIPv6Ary[ $i ]}";
		else if( strlen( $convertIPv6Ary[ $i ] ) === 3 )
			$convertIPv6	= "{$convertIPv6}0{$convertIPv6Ary[ $i ]}";
		else if( strlen( $convertIPv6Ary[ $i ] ) === 2 )
			$convertIPv6	= "{$convertIPv6}00{$convertIPv6Ary[ $i ]}";
		else if( strlen( $convertIPv6Ary[ $i ] ) === 1 )
			$convertIPv6	= "{$convertIPv6}000{$convertIPv6Ary[ $i ]}";
		else if( strlen( $convertIPv6Ary[ $i ] ) === 0 )
			$convertIPv6	= "{$convertIPv6}0000{$convertIPv6Ary[ $i ]}";
	}
	print $convertIPv6 . "<br>¥n";

	// 2桁ずつに分けて16進=>10進に変換
	$convertIPv6Ary = str_split( $convertIPv6 , 2 );
	$convertIPv6	= "";

	for( $i = 0 ; $i < count( $convertIPv6Ary ) ; ++$i ) {
		if( $i === 0 )
			$convertIPv6 = "{$convertIPv6}" . intval( hexdec( $convertIPv6Ary[ $i ] ) );
		else
			$convertIPv6 = "{$convertIPv6}." . intval( hexdec( $convertIPv6Ary[ $i ] ) );
	}
	print $convertIPv6 . "<br>¥n";

	return $convertIPv6;
?>
```

SNMP IPv6→OIDサンプル

mitene internet

```php
<?php
	$IPADRS = "2001:DB8::1:7961:1";
	//入力値
	print $IPADRS . "<br>¥n";

	//小文字・省略に統一する
	$IPADRS = inet_ntop ( inet_pton( $IPADRS ) );
	print $IPADRS . "<br>¥n";

	//コロンの数
	$COLON = substr_count( $IPADRS , ":" );
	print $COLON . "<br>¥n";

	//コロンの数に応じて省略をなくす
	if( $COLON === 6)
		$convertIPv6	= str_replace( "::" , ":0000:0000:" , $IPADRS );
	else if( $COLON === 5)
		$convertIPv6	= str_replace( "::" , ":0000:0000:0000:" , $IPADRS );
	else if( $COLON === 4)
		$convertIPv6	= str_replace( "::" , ":0000:0000:0000:0000:" , $IPADRS );
	print $convertIPv6 . "<br>¥n";

	//各オクテットを4桁にしてコロンをはずす
	$convertIPv6Ary = explode( ":" , $convertIPv6 );
	$convertIPv6	= "";

	for( $i = 0 ; $i < count( $convertIPv6Ary ) ; ++$i ) {
		if( strlen( $convertIPv6Ary[ $i ] ) === 4 )
			$convertIPv6	= "{$convertIPv6}{$convertIPv6Ary[ $i ]}";
		else if( strlen( $convertIPv6Ary[ $i ] ) === 3 )
			$convertIPv6	= "{$convertIPv6}0{$convertIPv6Ary[ $i ]}";
		else if( strlen( $convertIPv6Ary[ $i ] ) === 2 )
			$convertIPv6	= "{$convertIPv6}00{$convertIPv6Ary[ $i ]}";
		else if( strlen( $convertIPv6Ary[ $i ] ) === 1 )
			$convertIPv6	= "{$convertIPv6}000{$convertIPv6Ary[ $i ]}";
		else if( strlen( $convertIPv6Ary[ $i ] ) === 0 )
			$convertIPv6	= "{$convertIPv6}0000{$convertIPv6Ary[ $i ]}";
	}
	print $convertIPv6 . "<br>¥n";

	// 2桁ずつに分けて16進=>10進に変換
	$convertIPv6Ary = str_split( $convertIPv6 , 2 );
	$convertIPv6	= "";

	for( $i = 0 ; $i < count( $convertIPv6Ary ) ; ++$i ) {
		if( $i === 0 )
			$convertIPv6 = "{$convertIPv6}" . intval( hexdec( $convertIPv6Ary[ $i ] ) );
		else
			$convertIPv6 = "{$convertIPv6}." . intval( hexdec( $convertIPv6Ary[ $i ] ) );
	}
	print $convertIPv6 . "<br>¥n";

	return $convertIPv6;
?>
```

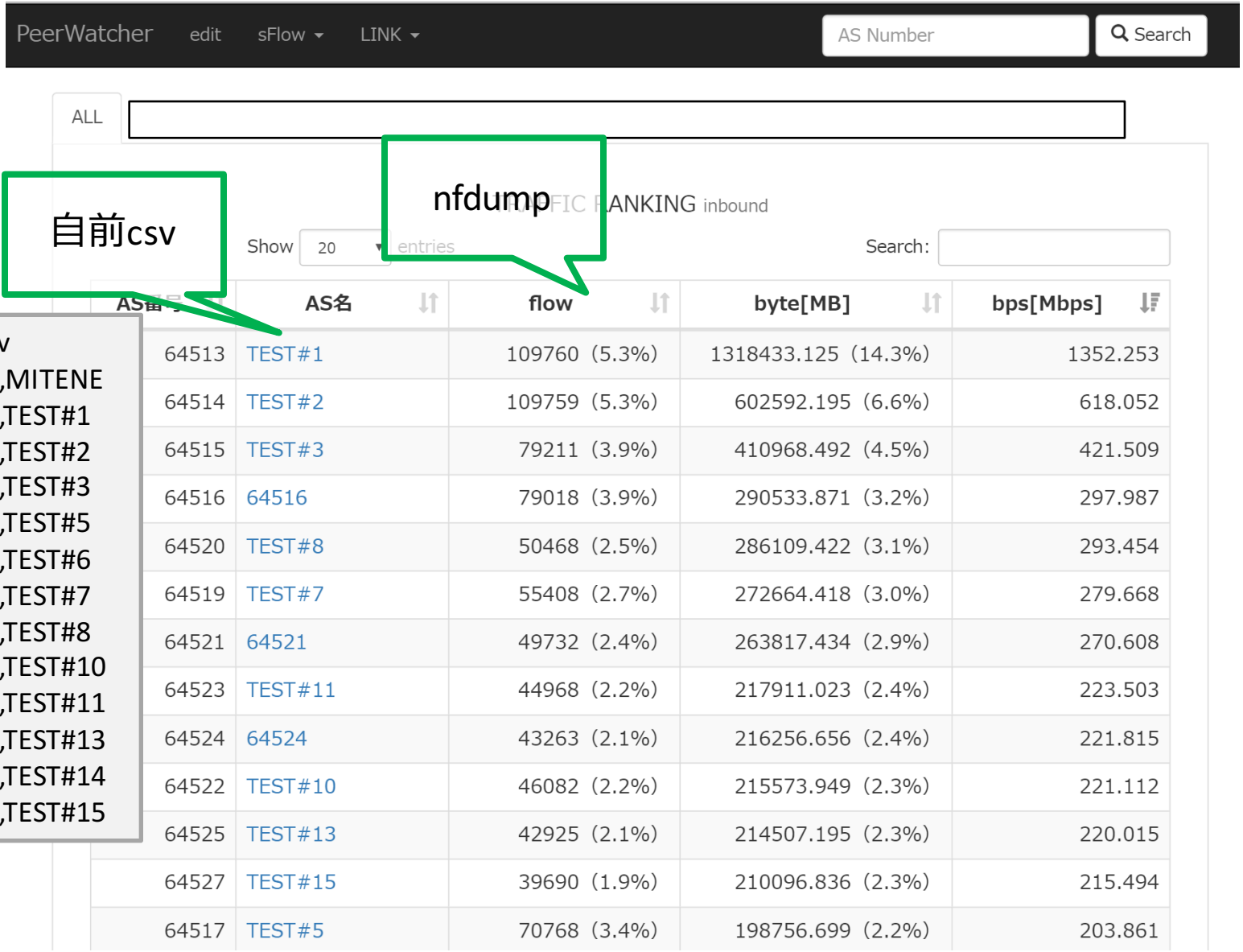
Rubyだったら
IPAddrで一発らしい・・・・

```
require 'ipaddr'
addr=IPAddr.new('2001:DB8::1:7961:1')
puts addr.to_string
```

Pythonでもできるらしい
http://qiita.com/Mabuchin/items/161d33f845ec0aeeb777

flow画面

mitene internet




```
#as.csv
17961,MITENE
64513,TEST#1
64514,TEST#2
64515,TEST#3
64517,TEST#5
64518,TEST#6
64519,TEST#7
64520,TEST#8
64522,TEST#10
64523,TEST#11
64525,TEST#13
64526,TEST#14
64527,TEST#15
```

mitene internet

●実行コマンド

/usr/local/bin/sfcapd -T all -t 600 -w -l {保存場所} -p {sflowポート番号}

●inbound TOP300出力

```
/usr/local/bin/nfdump ¥
-R $DIR/nfcapd.${DATE}2200:nfcapd.${DATE}2350 ¥
-s as -n 300 -o csv ¥
'(dst as 0 or dst as xxxxx)' ¥
> /home/hoge/csv/hogeIX-down.csv
```

前日の22:00-24:00のflowを利用する。

csvで出力

●outbound TOP300出力

```
/usr/local/bin/nfdump ¥
-R $DIR/nfcapd.${DATE}2200:nfcapd.${DATE}2350 ¥
-s as -n 300 -o csv ¥
'(src as 17961 or src as xxxxx)' ¥
> /home/hoge/csv/hoge-up.csv
```

src,dstでin/outを出す

●inbound TOP100出力@IX指定

```
/usr/local/bin/nfdump ¥
-R $DIR/nfcapd.${DATE}2200:nfcapd.${DATE}2350 ¥
-s as -n 100 -o csv ¥
'(vlan xx and router ip xx.xx.xx.xx) and (dst as 0 or dst as xxxxx)' ¥
> /home/hoge/csv/hoge-up.csv
```

ルータIPとVLANをIX・トランジットに合わせる

間隔は1分に

できるだけ過去に遡れるように大きな値にしておく。

●create

```
/usr/bin/rrdtool  create ¥
/home/rrdtool/rrd/hogehogebbn.rrd  ¥
--step 60  ¥
DS:traffic_in:COUNTER:600:0:U ¥
DS:traffic_out:COUNTER:600:0:U ¥
RRA:AVERAGE:0.5:1:50000 ¥
RRA:AVERAGE:0.5:6:50000 ¥
RRA:AVERAGE:0.5:24:50000 ¥
RRA:AVERAGE:0.5:288:50000 ¥
RRA:MAX:0.5:1:50000 ¥
RRA:MAX:0.5:6:50000 ¥
RRA:MAX:0.5:24:50000 ¥
RRA:MAX:0.5:288:50000 ¥
```

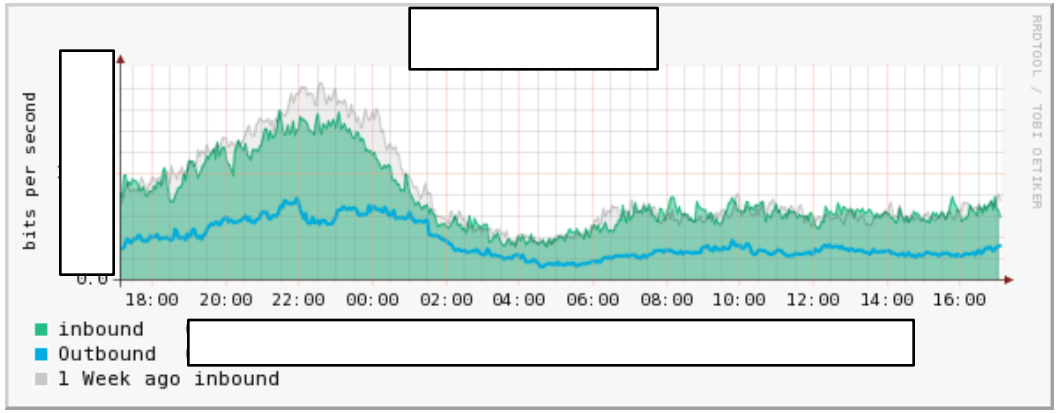
●update（1分に1度）

```
/usr/bin/rrdtool  update ¥
/home/rrdtool/rra/hogehogebbn.rrd  ¥
 --template traffic_in:traffic_out ¥
 `date +%s`:`{SNMPコマンド}`:`{SNMPコマンド}`
```

rrdtool graph

mitene internet

```
/usr/bin/rrdtool graph /home/rrdtool/image/hogehogebbn.png ¥
--imgformat=PNG ¥
--start='-86400' ¥
--end='-30' ¥
--title='タイトル' ¥
--rigid ¥
--base='1000' ¥
--height='120' ¥
--width='500' ¥
--alt-autoscale-max ¥
--lower-limit='0' ¥
--vertical-label='bits per second' ¥
--slope-mode ¥
--font TITLE:10: ¥
--font AXIS:7: ¥
--font LEGEND:8: ¥
--font UNIT:7: ¥
--color BACK#F7F7F7 ¥
--color FRAME#FFFFFF ¥
--color MGRID#FF9F8080 ¥
DEF:a='/home/rrdtool/rra/hogehogebbn.rrd ':'traffic_in':AVERAGE:start=end-2w ¥
DEF:b='/home/rrdtool/rra/hogehogebbn.rrd ':'traffic_out':AVERAGE ¥
CDEF:cdefa='a,8,*' ¥
CDEF:cdefc=cdefa ¥
CDEF:cdefe='b,8,*' ¥
SHIFT:cdefc:604800 ¥
AREA:cdefa#25BC8580 ¥
LINE:cdefa#25BC85:'inbound' ¥
GPRINT:cdefa:LAST:' Current¥:%8.2lf %s' ¥
GPRINT:cdefa:AVERAGE:'Average¥:%8.2lf %s' ¥
GPRINT:cdefa:MAX:'Maximum¥:%8.2lf %s¥n' ¥
AREA:cdefc#00000010 ¥
LINE2:cdefe#00ADDE:'Outbound' ¥
GPRINT:cdefe:LAST:'Current¥:%8.2lf %s' ¥
GPRINT:cdefe:AVERAGE:'Average¥:%8.2lf %s' ¥
GPRINT:cdefe:MAX:'Maximum¥:%8.2lf %s¥n' ¥
LINE:cdefc#00000030:'1 Week ago inbound¥n'
```



```
/usr/bin/rrdtool graph /home/rrdtool/image/hogehogebbn.png ¥
--imgformat=PNG ¥
--start='-86400' ¥
--end='-30' ¥
--title='タイトル' ¥
--rigid ¥
--base='1000' ¥
--height='120' ¥
--width='500' ¥
--alt-autoscale-max ¥
--lower-limit='0' ¥
--vertical-label='bits per s ¥
--slope-mode ¥
--font TITLE:10: ¥
--font AXIS:7: ¥
--font LEGEND:8: ¥
--font UNIT:7: ¥
--color BACK#F7F7F7 ¥
--color FRAME#FFFFFF ¥
--color MGRID#FF9F8080 ¥
DEF:a='/home/rrdtool/rra/hogehogebbn.rrd ':'traffic_in':AVERAGE:start=end-2w ¥
DEF:b='/home/rrdtool/rra/hogehogebbn.rrd ':'traffic_out':AVERAGE ¥
CDEF:cdefa='a,8,*' ¥
CDEF:cdefc=cdefa ¥
CDEF:cdefe='b,8,*' ¥
SHIFT:cdefc:604800 ¥
AREA:cdefa#25BC8580 ¥
LINE:cdefa#25BC85:'inbound' ¥
GPRINT:cdefa:LAST:' Current¥:%8.2lf %s' ¥
GPRINT:cdefa:AVERAGE:'Average¥:%8.2lf %s' ¥
GPRINT:cdefa:MAX:'Maximum¥:%8.2lf %s¥n' ¥
AREA:cdefc#00000010 ¥
LINE2:cdefe#00ADDE:'Outbound' ¥
GPRINT:cdefe:LAST:'Current¥:%8.2lf %s' ¥
GPRINT:cdefe:AVERAGE:'Average¥:%8.2lf %s' ¥
GPRINT:cdefe:MAX:'Maximum¥:%8.2lf %s¥n' ¥
LINE:cdefc#00000030:'1 Week ago inbound¥n'
```

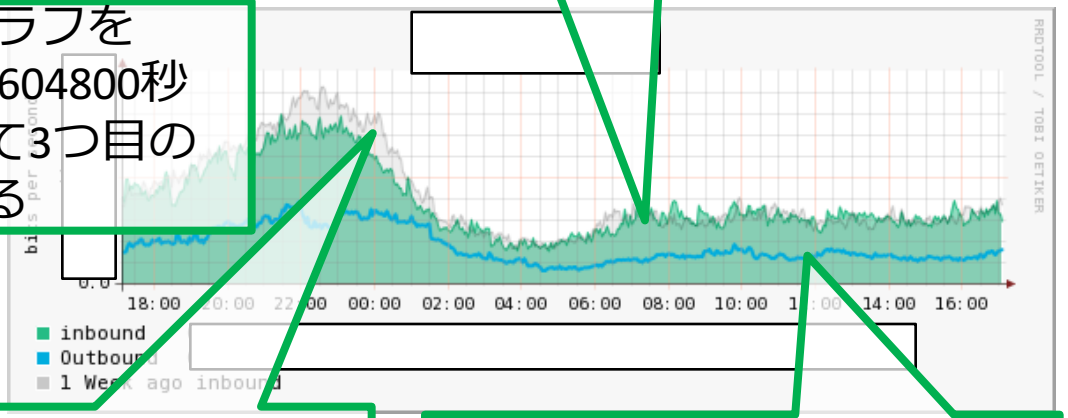
滑らかなグラフに

2週間分のデータを取得

Inboud(1day)
枠線：25BC85
塗枠：25BC85(透過率20%)

inboundのグラフを
60x60x24x7=604800秒
前にずらして3つ目の
グラフにする

線を太く



inbound(1週間前)
枠線：#000000(透過率70%)
塗枠：#000000(透過率90%)

outbonud(1日)
枠線：#00ADDE

https://www.peeringdb.com/api/net?asn={AS番号}
でASの詳細、irr_as_set、maxprefix等が取得できます。
この検索で一番欲しい内容は“id”

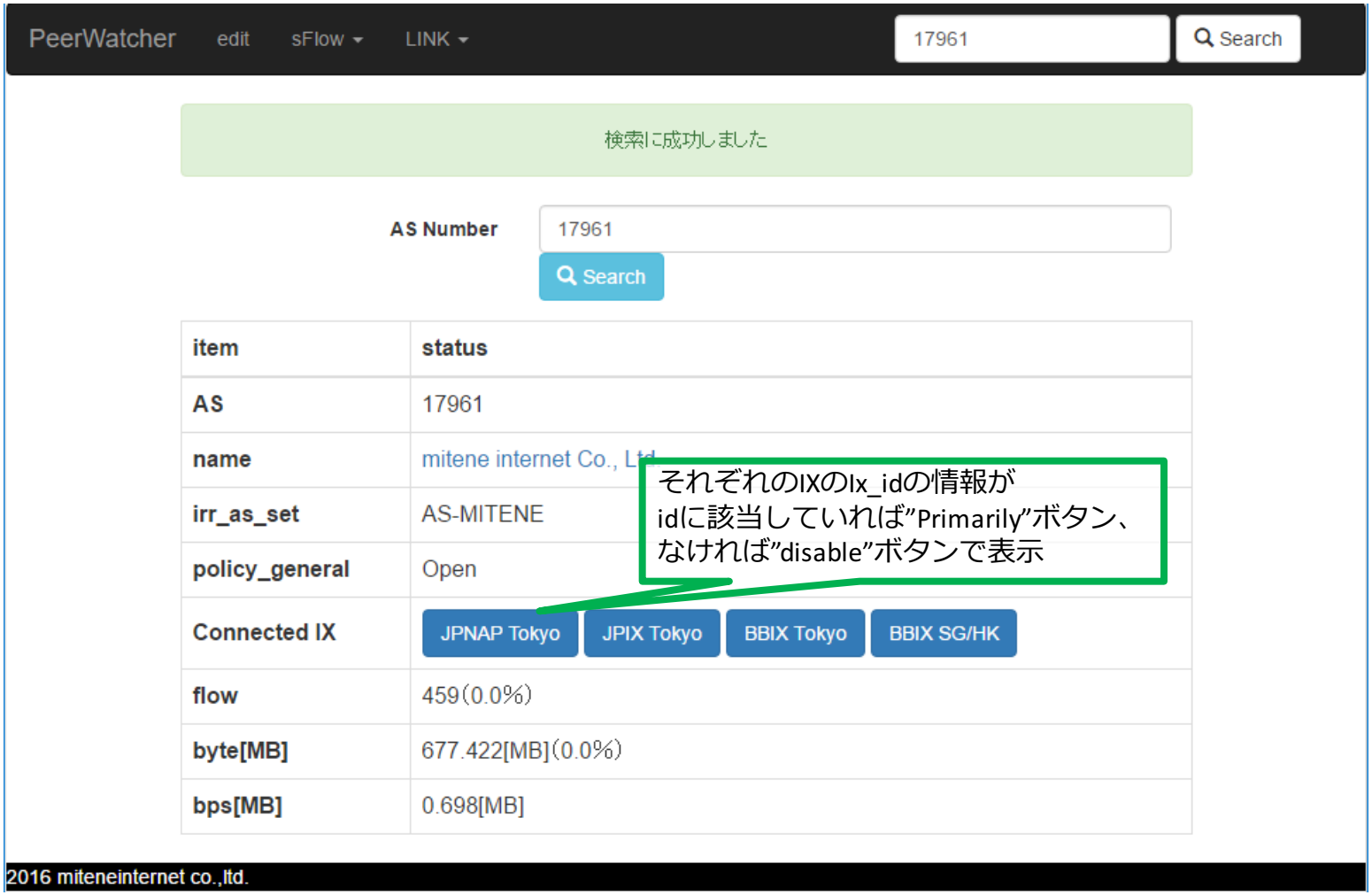
PeeringDB内でファシリティやIXの情報を引き出す際にはidで検索する必要があります。miteneの場合は”2806”

https://www.peeringdb.com/api/net/2806
を引っ張ってくることで接続IX情報を引っ張ってくる事ができます。

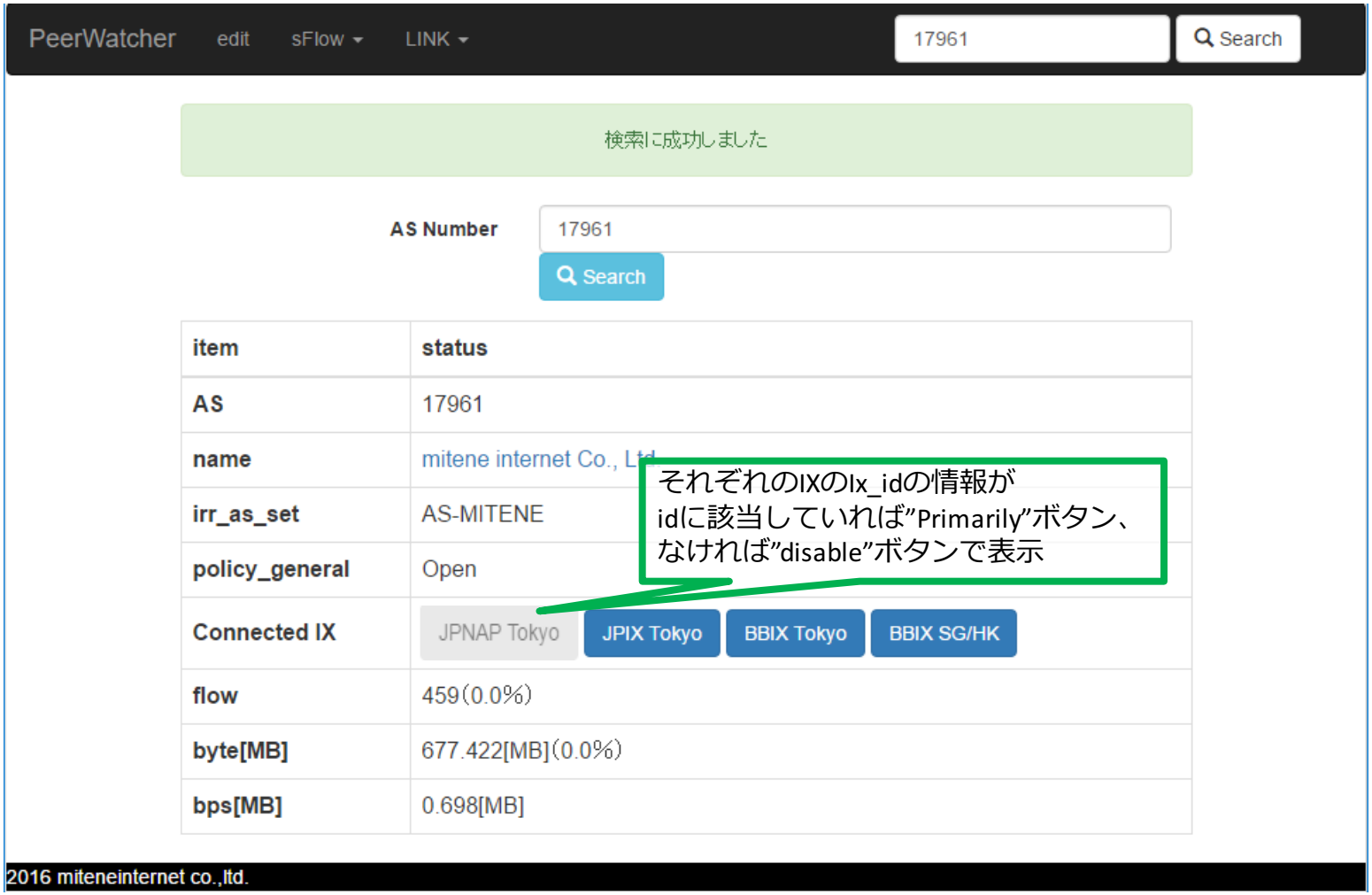
IX毎にもローカルで持っているIDがあります。(ix_id)
ex)
ix_id=30 JPIX Tokyo
ix_id=95 JPNAP Tokyo I
Ix_id=126 BBIX Tokyo

AS番号から（PeeringDB内のローカル）idを呼び出して、ix_id内の情報を引き出すという作業ができます。

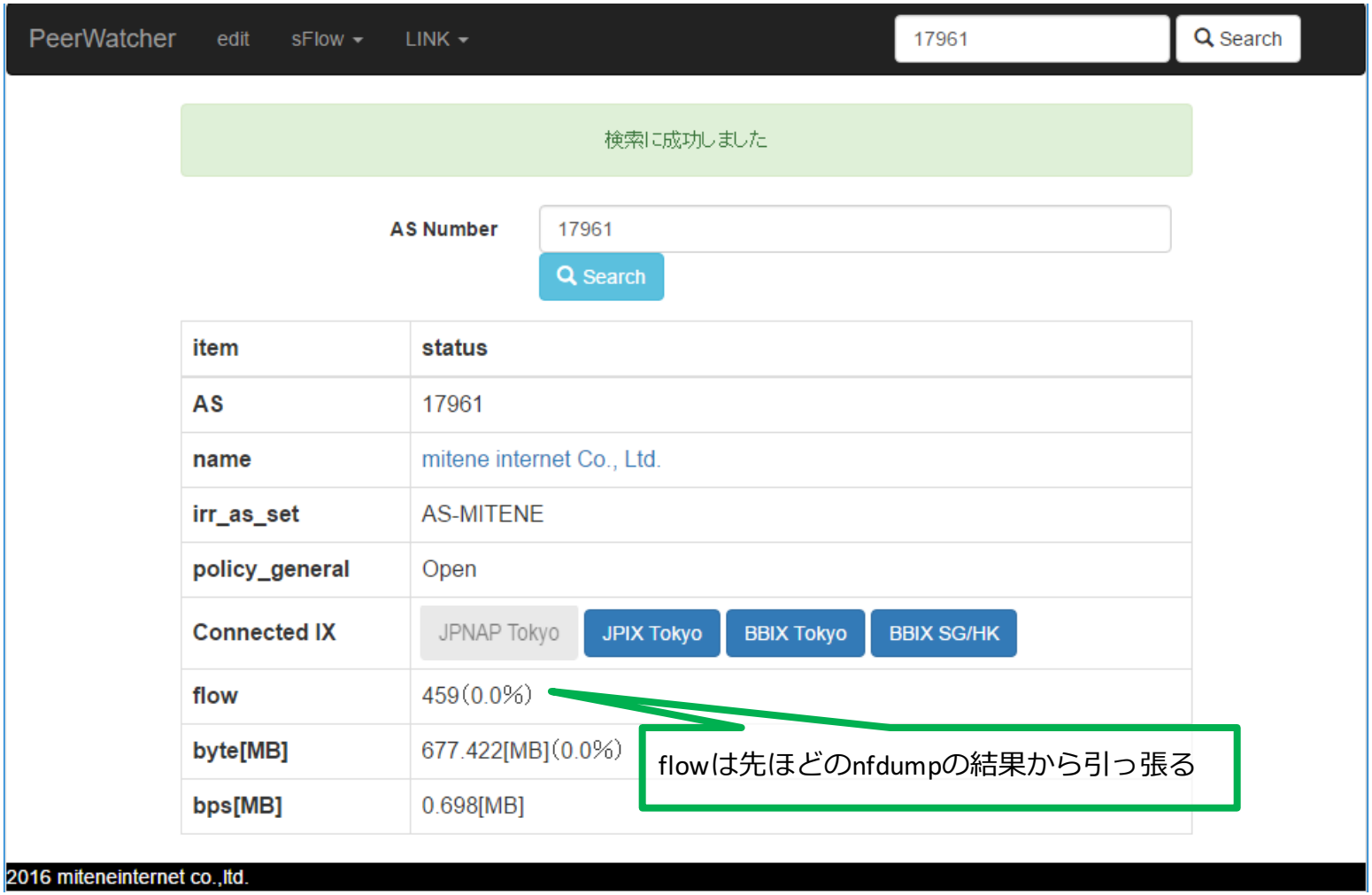
AS検索


PeerWatcher
edit
sFlow
LINK
17961
Search
検索に成功しました
AS Number
17961
Search
item
status
AS
17961
name
mitene internet Co., Ltd
irr_as_set
AS-MITENE
policy_general
Open
Connected IX
JPNAP Tokyo
JPIX Tokyo
BBIX Tokyo
BBIX SG/HK
flow
459(0.0%)
byte[MB]
677.422[MB](0.0%)
bps[MB]
0.698[MB]
それぞれのIXのIx_idの情報が
idに該当していれば”Primarily”ボタン、
なければ”disable”ボタンで表示
2016 miteneinternet co.,ltd.

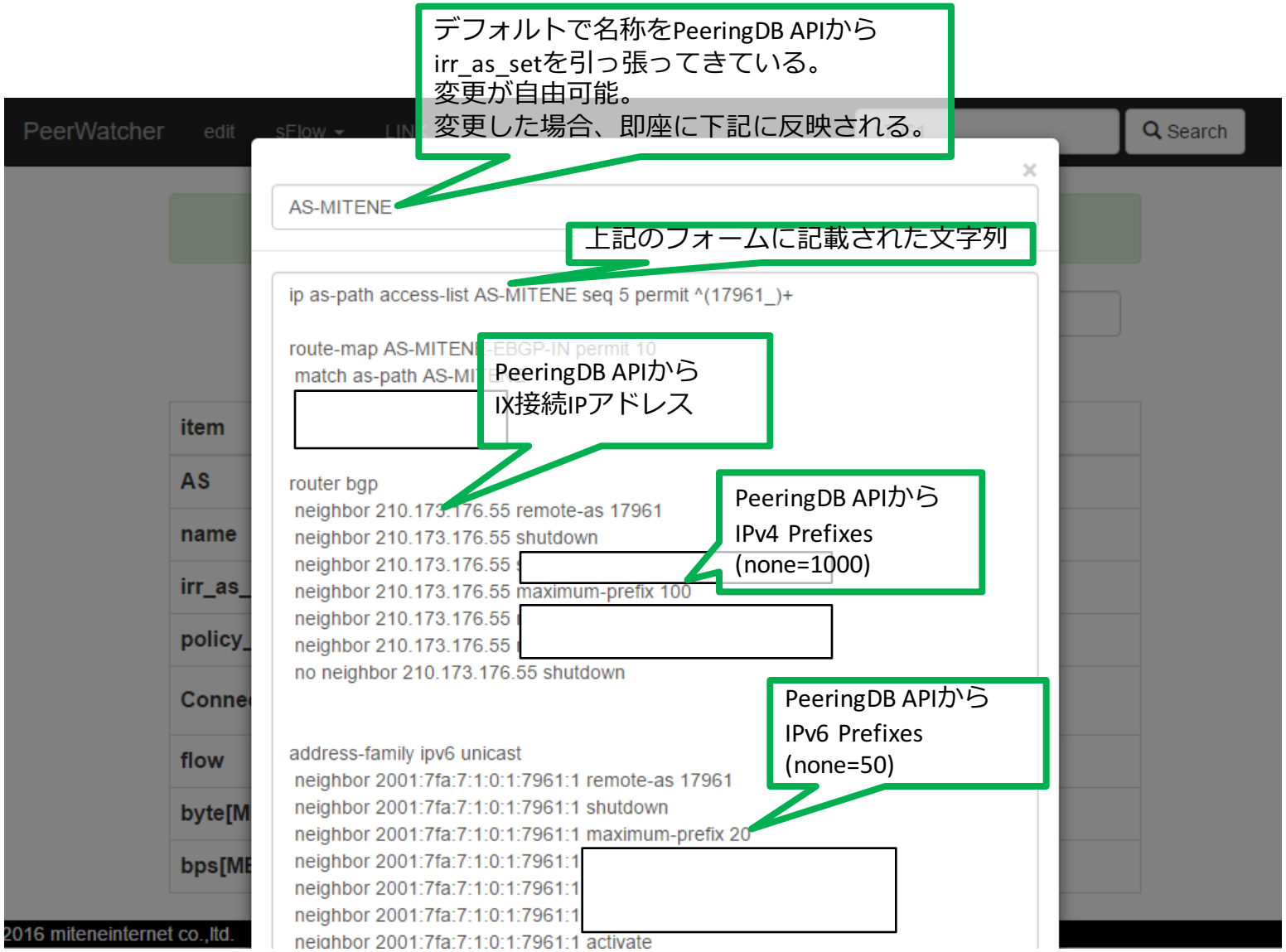
AS検索


PeerWatcher
edit
sFlow
LINK
17961
Search
検索に成功しました
AS Number
17961
Search
item
status
AS
17961
name
mitene internet Co., Ltd
irr_as_set
AS-MITENE
policy_general
Open
Connected IX
JPNAP Tokyo
JPIX Tokyo
BBIX Tokyo
BBIX SG/HK
flow
459(0.0%)
byte[MB]
677.422[MB](0.0%)
bps[MB]
0.698[MB]
それぞれのIXのIx_idの情報が
idに該当していれば”Primarily”ボタン、
なければ”disable”ボタンで表示
2016 miteneinternet co.,ltd.

AS検索


PeerWatcher
edit
sFlow
LINK
17961
Search
検索に成功しました
AS Number
17961
Search
item
status
AS
17961
name
mitene internet Co., Ltd.
irr_as_set
AS-MITENE
policy_general
Open
Connected IX
JPNAP Tokyo
JPIX Tokyo
BBIX Tokyo
BBIX SG/HK
flow
459(0.0%)
byte[MB]
677.422[MB](0.0%)
bps[MB]
0.698[MB]
flowは先ほどのnfdumpの結果から引っ張る
2016 miteneinternet co.,ltd.

AS検索


デフォルトで名称をPeeringDB APIから
irr_as_setを引っ張ってきている。
変更が自由可能。
変更した場合、即座に下記に反映される。
PeerWatcher
edit
sFlow
Search
AS-MITENE
上記のフォームに記載された文字列
ip as-path access-list AS-MITENE seq 5 permit ^(17961_)+
route-map AS-MITENE-EBGP-IN permit 10
match as-path AS-MITENE
PeeringDB APIから
IX接続IPアドレス
router bgp
neighbor 210.173.176.55 remote-as 17961
neighbor 210.173.176.55 shutdown
neighbor 210.173.176.55
neighbor 210.173.176.55 maximum-prefix 100
neighbor 210.173.176.55
neighbor 210.173.176.55
no neighbor 210.173.176.55 shutdown
PeeringDB APIから
IPv4 Prefixes
(none=1000)
PeeringDB APIから
IPv6 Prefixes
(none=50)
address-family ipv6 unicast
neighbor 2001:7fa:7:1:0:1:7961:1 remote-as 17961
neighbor 2001:7fa:7:1:0:1:7961:1 shutdown
neighbor 2001:7fa:7:1:0:1:7961:1 maximum-prefix 20
neighbor 2001:7fa:7:1:0:1:7961:1
neighbor 2001:7fa:7:1:0:1:7961:1
neighbor 2001:7fa:7:1:0:1:7961:1
neighbor 2001:7fa:7:1:0:1:7961:1 activate
item
AS
name
irr_as_
policy_
Conne
flow
byte[M
bps[MB
2016 miteneinternet co.,ltd.

今後の改善したい点

mitene internet

・Peer先の直ASでしかトラフィック量が表示されない
　　-sflowでASからinsrcmacを求めて、そのinsrcmacでnfdumpから
　　トラフィックを合算して求めればできるかもしれない。
　　それができれば残りの値でRouteServer経由のトラフィックも測れる。

・5分毎のFlow量を調べて、極端な増減があった際にアラートを出す。
　　-DDoSの早期発見に繋がる？しきい値の調整が大変そう。
　　-増減が大ければsrc/dstのIPアドレスのTOPとか表示できれば幸せ？

・ほぼリアルタイムなFlow結果も出したい。
　　-負荷が気になる・・・

・もっといろんなデータをrrdtoolで取りたい。
　　-いろんな方面から見える化できるように。

・モバイルでもちゃんとみられるようなレイアウトに
　　-今はぐちゃぐちゃ。

Thanks

mitene internet

・作るきっかけになったJPNAP APIハンズオンの講師
杉本さん（インターネットマルチフィード株式会社）
川上さん（インターネットマルチフィード株式会社）

・SNMPの情報や、sflowの解析などのサポートを頂いた
篠宮さん（株式会社FORNEXT）

その他いろんな相談に乗っていただいた皆様
ありがとうございました！！

ミテネインターネット株式会社

mitene internet co.,ltd

http://www.mitene.co.jp