YAMAHAルータの死活監視をAWSでやって見えたこと

株式会社ハウインターナショナル 岩男皓一朗 @JANOG 41

1枚10秒ペースなので

巻きで行きます

スピーカーについて

- 福岡から来ました 名強の中小 業
- ひとり情シスでネットワークエンジニア兼
- 一応ネットワークが専門の
- xrdpというOSSの
- IPv6対応 などを中心に業務とプライベートで開発
- 自社サービスのほぼ全て AWSで提供している

身の上話より本題の話
あとで懇親会で！

NWについては
ほぼユーザ側です

当初の目的

弊社の対外回線の死活監視

中から外への疎通を監視してもダウン時に通知する手段がない

いまさら社内に監視・通知サーバを置いてお守りはしたくない

ダウンしたらモバイル回線でSMS通知できると冗長性・利便性の面で最高！

フルマネージドなクラウドサービスでやりたい！

皆様、覚えていますか…？

読売新聞より引用（2016年11月8日午前7時52分、福岡市博多区で）＝山田伸彦撮影

博多駅前陥没事故

NTT西・QTnetのファイバもダメージを受ける → その節は大変お疲れ様でした

アクセス回線を冗長化していても、災害により両方が使えなくなることがある

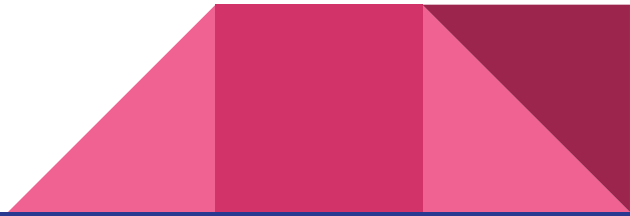
そもそも中小企業ではアクセス回線の冗長化までコストをかけられない

マルチホーミングなんて論外！　地方はアクセス回線の選択肢も少ない

フレッツと電力系でシングルホーム×2 がやっと？

大事なことなので

フルマネージドなクラウドサービスで死活監視したい！



監視の対象とするのは

中小企業御用達のYAMAHAルータ

RTX1210, RTX830

Amazon Route 53 Health Check

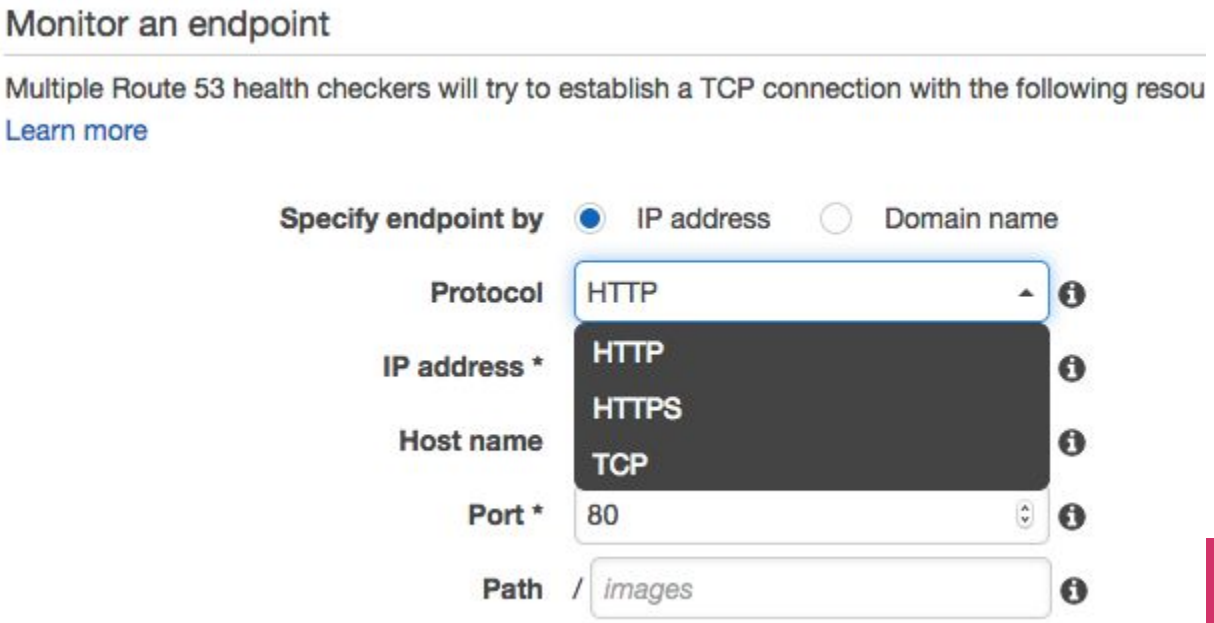
AWS外のエンドポイント1つにつき $0.75/month

レイテンシ計測のオプションを追加すると +$2.00/mo

日本の Route 53 は岡山駅付近から鳥取へ

JANOGの帰りに立ち寄ってみてください

Amazon Route 53 Health Check



NO ICMP!!!

困った

YAMAHAルータにこんな機能がある

Lua スクリプト機能

- 概要
- 対応機種とファームウェアリビジョン
- Lua スクリプト機能バージョンの変更履歴
- 用語の定義
- 詳細
- 注意事項
- コマンド
- 設定例
- SYSLOG メッセージ一覧
- 参考情報

Luaスクリプト機能

YAMAHAルータ*1 はLuaスクリプトでプログラマブル

USBポートにバーコードリーダを接続して読み取ったりすることもできる *2

これは使える！

*1　概ね2008年11月発売のRTX1200以降の機種

*2　https://www.slideshare.net/nvsofts/rtx-49061065

ICMPがないなら

TCPでやればいいじゃない

Luaスクリプトで
TCPサーバを実装して
ルータ上で動かす

できた

ソースコード
http://bit.ly/YamahaLuaTCPServer

health_check_server.lua

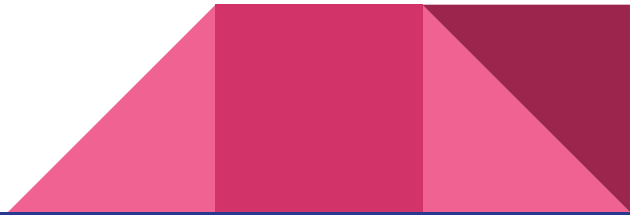
```lua
tcp = rt.socket.tcp()
tcp:setoption("reuseaddr", true)
res, err = tcp:bind("0.0.0.0", 3800)

if not res and err then
  print(err)
  os.exit(1)
end

res, err = tcp:listen()
if not res and err then
  print(err)
  os.exit(1)
end

while 1 do
  control = assert(tcp:accept())
  control:settimeout(10)
  sent, err = control:send("I'm alive.\n")
  control:close()
  rt.sleep(1)
end
```

2017年7月13日 運用開始

実際に運用してみて

AWSのマネジメントコンソールから

レイテンシのログを見ることができる



AWS CloudWatch

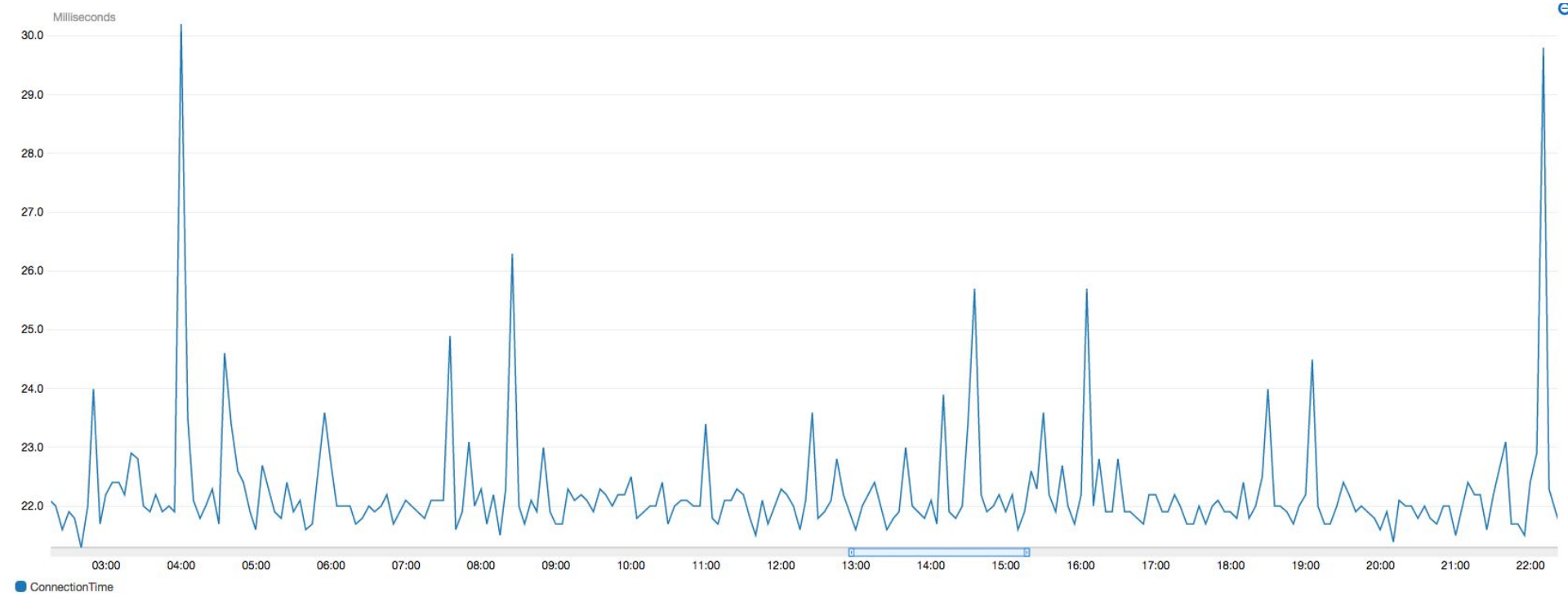
データの保存期間は最大15ヶ月

表示期間を絞ったり、対象データを絞ったり、マウスでぐりぐり

古いものほどデータが間引かれていき、1分平均、5分平均、15分平均と
だんだん粗いデータしか参照できなくなる

詳細なAWS CloudWatchの公式資料で！

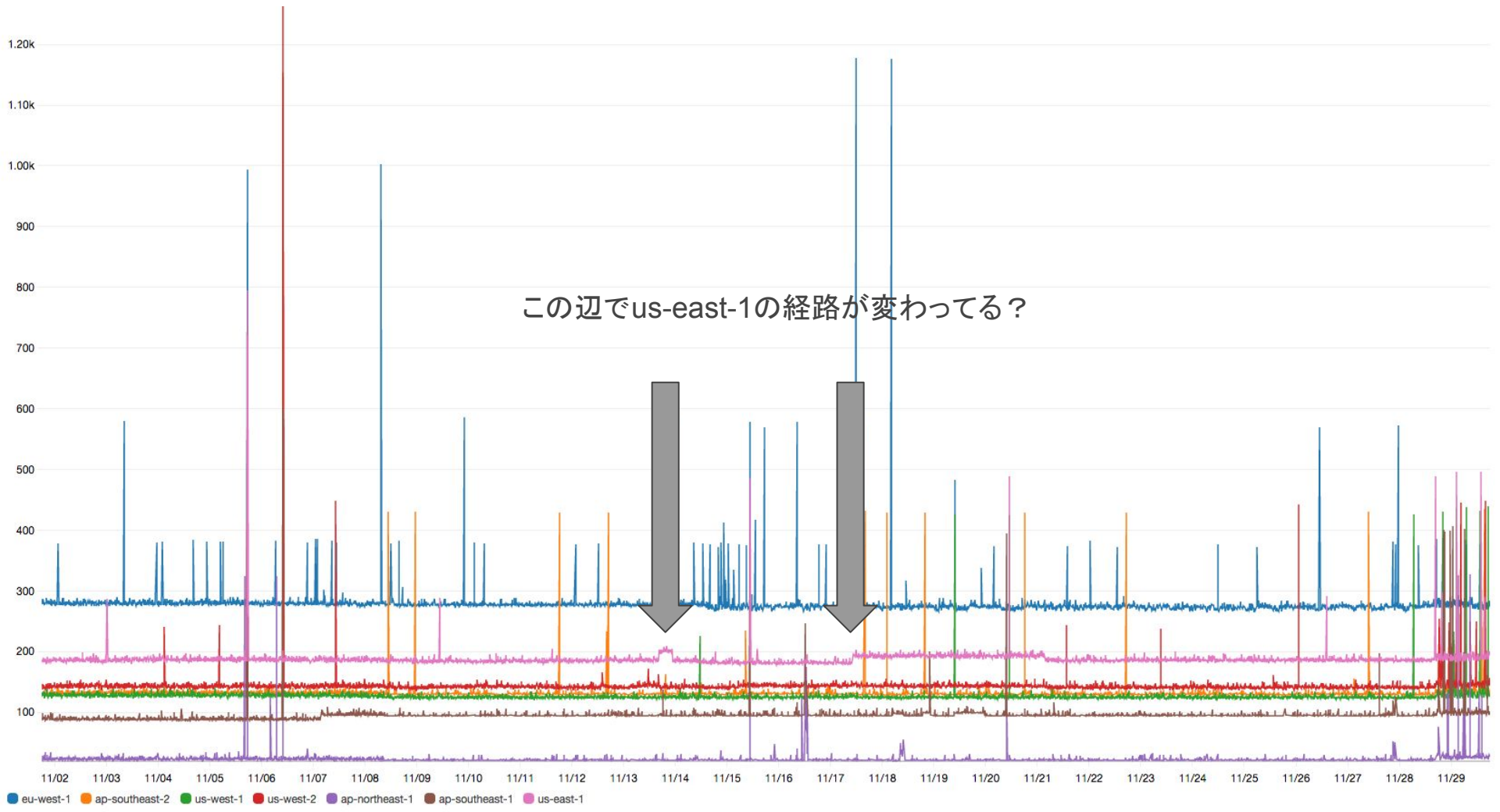
ap-northeast-1 から福岡 (1min avg)



AWS東京リージョンから福岡

だいたい25ms切るくらい


Milliseconds
579
550
500
450
400
350
300
250
200
150
100
50.0
2017年10月25日
東京とシンガポールリージョンで
RTTの変化がシンクロしている
2017-10-25 06:50 UTC
1. eu-west-1 278
2. us-east-1 189
3. us-west-2 139
4. ap-southeast-2 133
5. us-west-1 123
6. ap-southeast-1 87.2
7. ap-northeast-1 23.9
00:00
03:00
10-25 06:49
09:00
12:00
15:00
18:00
21:00
00:00
03:00
06:00
09:00
12:00
15:00
18:00
21:00
ap-northeast-1
eu-west-1
ap-southeast-2
us-west-1
us-west-2
ap-southeast-1
us-east-1


この辺でus-east-1の経路が変わってる？
1.20k
1.10k
1.00k
900
800
700
600
500
400
300
200
100
11/02
11/03
11/04
11/05
11/06
11/07
11/08
11/09
11/10
11/11
11/12
11/13
11/14
11/15
11/16
11/17
11/18
11/19
11/20
11/21
11/22
11/23
11/24
11/25
11/26
11/27
11/28
11/29
eu-west-1
ap-southeast-2
us-west-1
us-west-2
ap-northeast-1
ap-southeast-1
us-east-1

あれ？

これSmokePingっぽくない？

想定していなかった副次的効果

1エンドポイントあたりわずか $2.75/month で

回線の死活監視とSmokePing“っぽい”ことができてしまった

(RTTの幅を表現してくれないのであくまで“っぽい”)

SmokePingっぽい

死活を監視したところでそんなに死なないのが現実

というより、死なせないのが仕事

運用開始後、一度もアラートは飛んできていない

こちらの副次的に得られた機能の方が重宝している

普段からRTTを記録しておくと経路が乱れても安心！

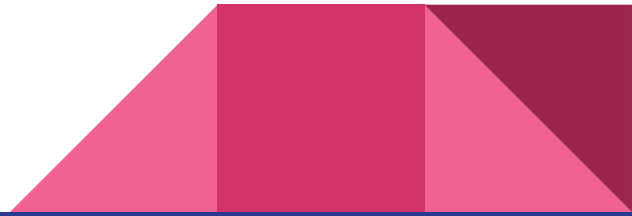
SmokePingとの違い

計測しているのはICMPではなくTCPでのRTT

行きと戻りの経路は同じとは限らない

実際には戻りの経路ですらなくAWSからの行きの経路

計れるのはAWSの各リージョンからエンドポイントへのRTT



まとめ

Amazon Route 53 Health Check を使って弊社対外回線の死活監視を実現

R53HCはICMPをサポートしていないが、YAMAHAルータ上にTCPサーバを
実装することでTCPによる死活監視・レイテンシの計測を実現 (他社製品は？)

当初想定していなかったSmokePing的な機能が得られた

SmokePingの安価な代用としては必要十分以上 ($2.75/month)

安く・早く・楽かつランニング人件費もわずか