Telemetry

Shishio Tsuchiya

shtsuchi@cisco.com

Agenda

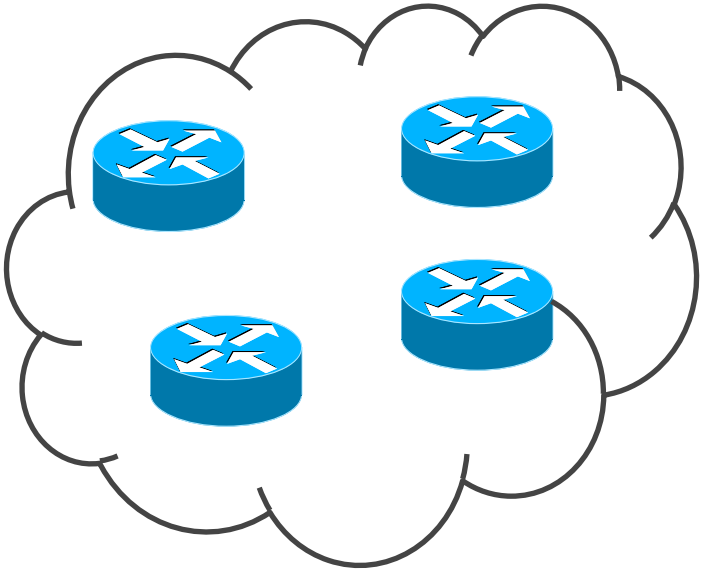
- **テレメトリー登場背景**
- **エンコード手法(誰が進めているのか)**
- 日本の運用者におけるテレメトリー例
- 実装例

Telemetry

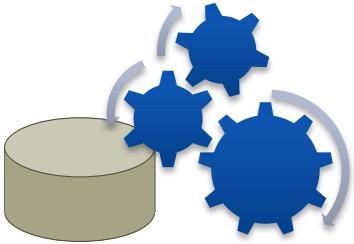
- 遠隔測定法（えんかくそくていほう）は、観測対象から離れた地点から様々な観測を行い、そのデータを取得する技術である。観測地点に常駐することが物理的・経済的あるいは安全上困難な場合や、観測対象が移動する場合に使用される。テレメトリー (telemetry) あるいはテレメタリング (telemetering) ということもある。 装置そのものは、テレメータ (telemeter) と呼ばれる。

https://ja.wikipedia.org/wiki/遠隔測定法

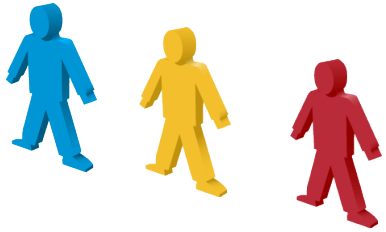
どういう事か？



ネットワーク

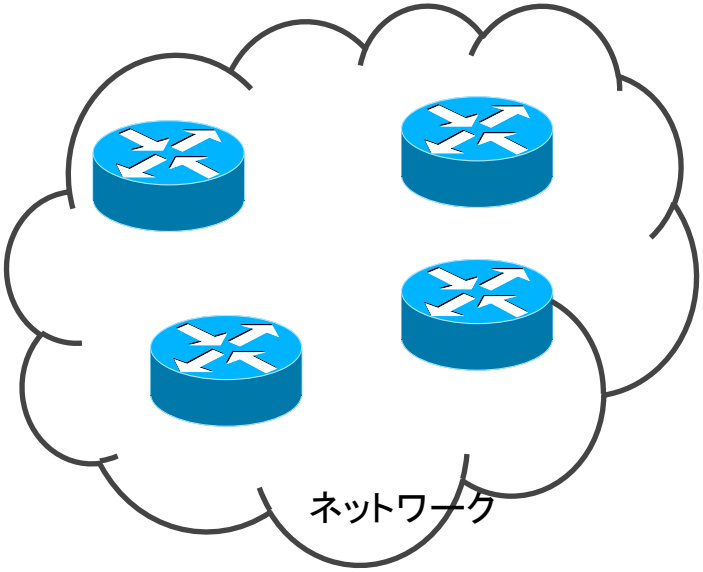


ストレージ/解析



- テレメーターはネットワーク・ネットワーク機器となる

ネットワークでの遠隔診断の問題点

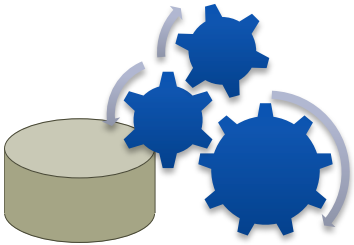


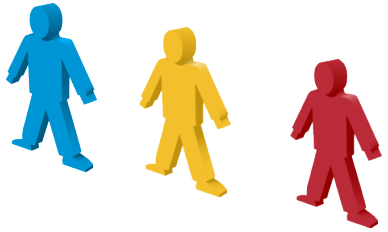
SNMP

Syslog

xFlow

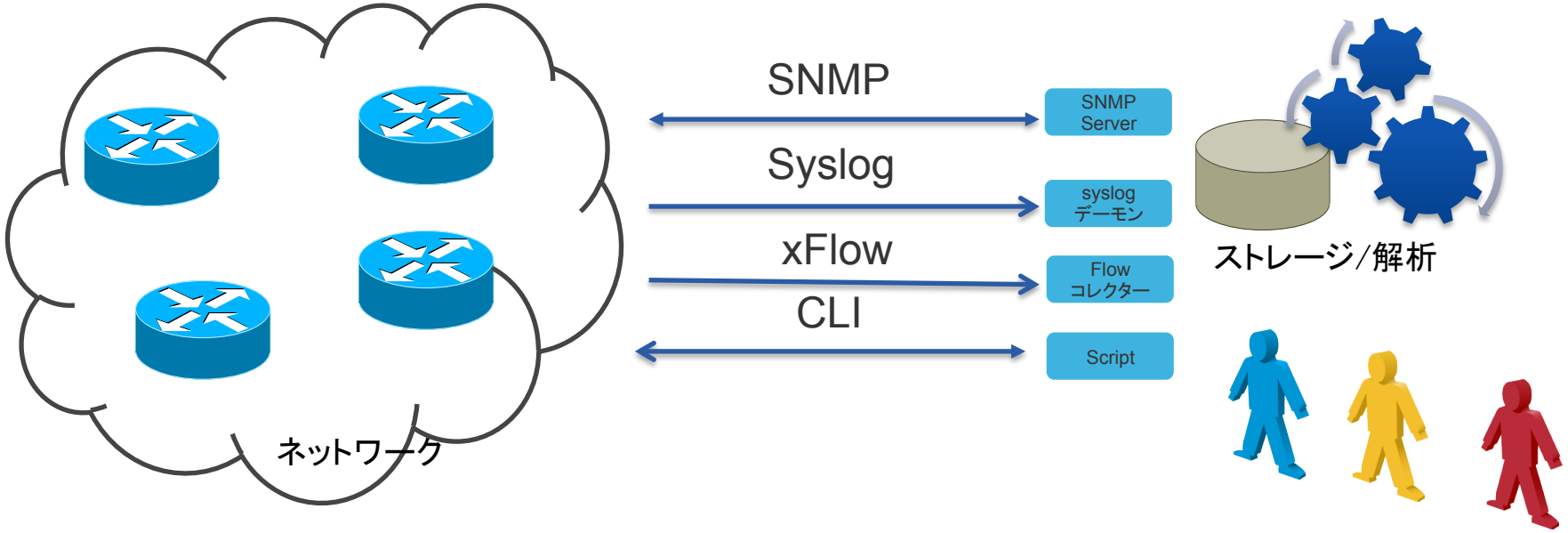
CLI





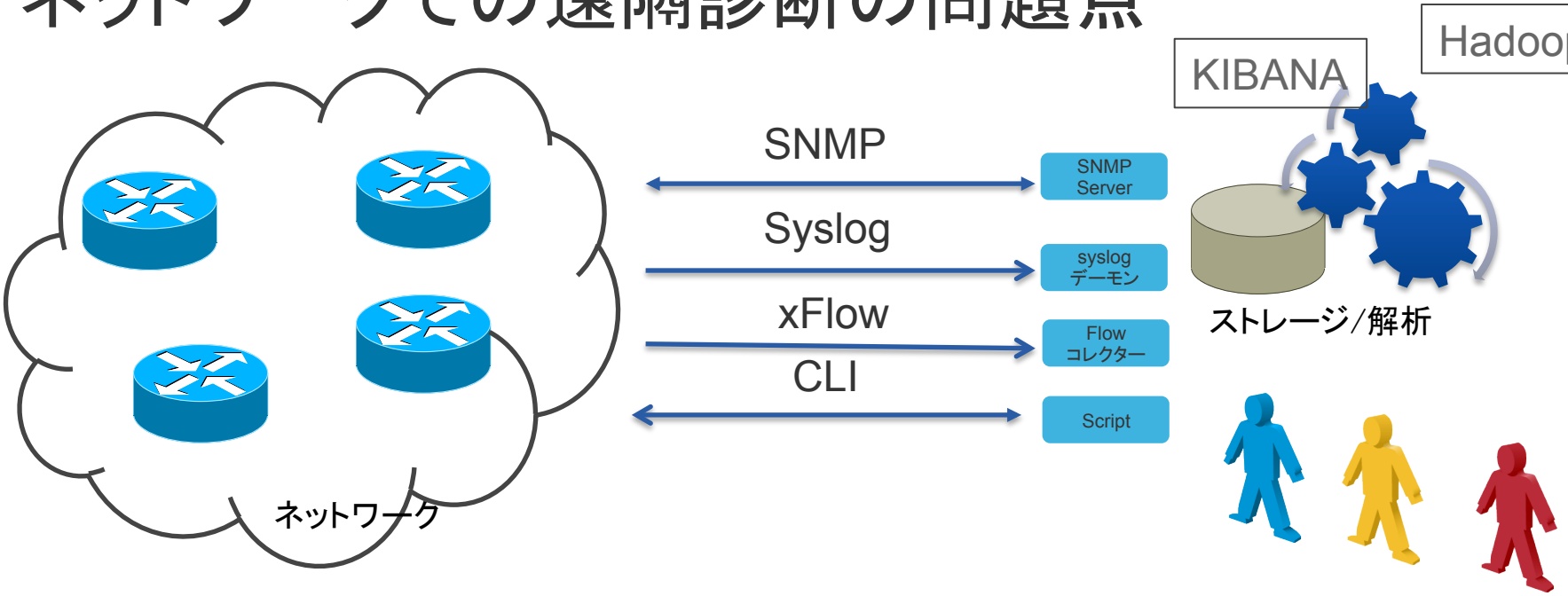
- スケール問題：SNMP/CLI
- 標準的では無いフォーマット：Syslog/SNMP
- 単体では解析には使えない:SNMP/Syslog

ネットワークでの遠隔診断の問題点



- フォーマットがすべて異なる
- 異なるトランスポート
- タイムスタンプ情報

ネットワークでの遠隔診断の問題点



- 異なるフォーマットからの解析ツールへの変換

Agenda

- テレメトリー登場背景
- **エンコード手法(誰が進めているのか)**
- 日本の運用者におけるテレメトリー例
- 実装例

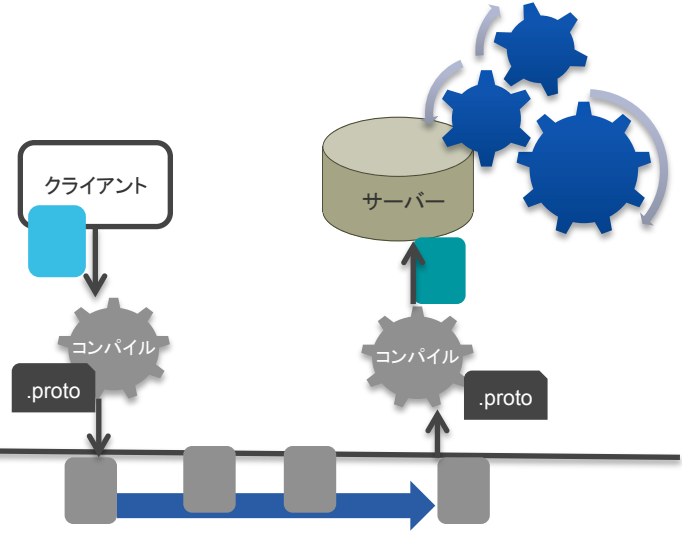
Google Protocol Buffers

https://developers.google.com/protocol-buffers/

- 2001年 Googleにより開発
- 2008年にBSDライセンスでオープンソース化
- 他プログラム言語間のシリアライゼーションを定義
- .protoファイルでメッセージフォーマットを作成
- サポート言語：

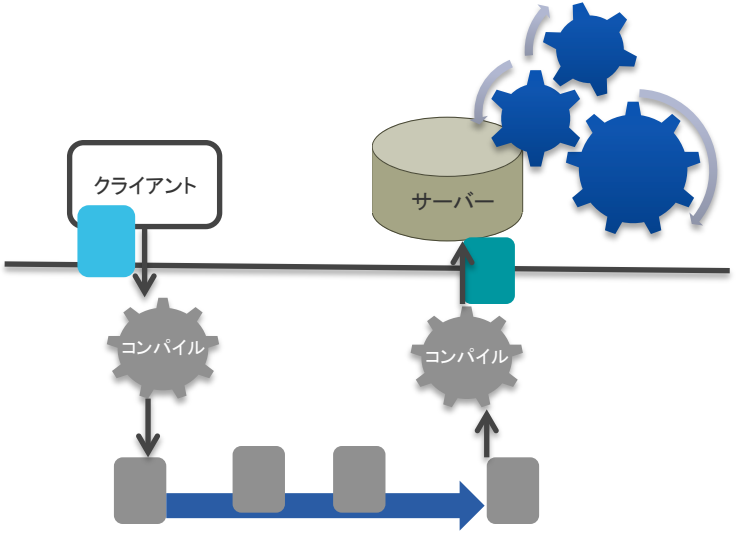
  Java, Python,C++

  Go, JavaNano, Ruby,C# (proto3)
- RPCは未定義



Apache Thrift(スリフト)

https://thrift.apache.org/

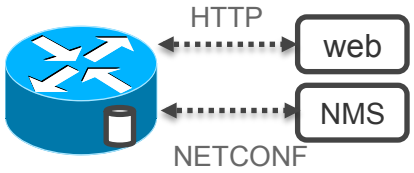


- 2007年 Facebookにより開発
- Apacheライセンス
- 他プログラム言語間のフレームワーク
- コード生成からRPCプロトコルまで規定
- サポート言語:

  Java, C++, Python, C#, Cocoa, Erlang, Haskell, OCaml, Perl, PHP, Ruby, Smalltalk
- トランスポートはTCPを使用する

RESTCONF

draft-ietf-netconf-restconf



Mgmt info (definition)	YANG modules	
Mgmt info (encoding)	XML-encoded content	XML/ JSON
Mgmt Services	NETCONF operations	RESTCONF
Remote Operations	XML RPC	HTTP
Transport	TLS, SSH	TCP

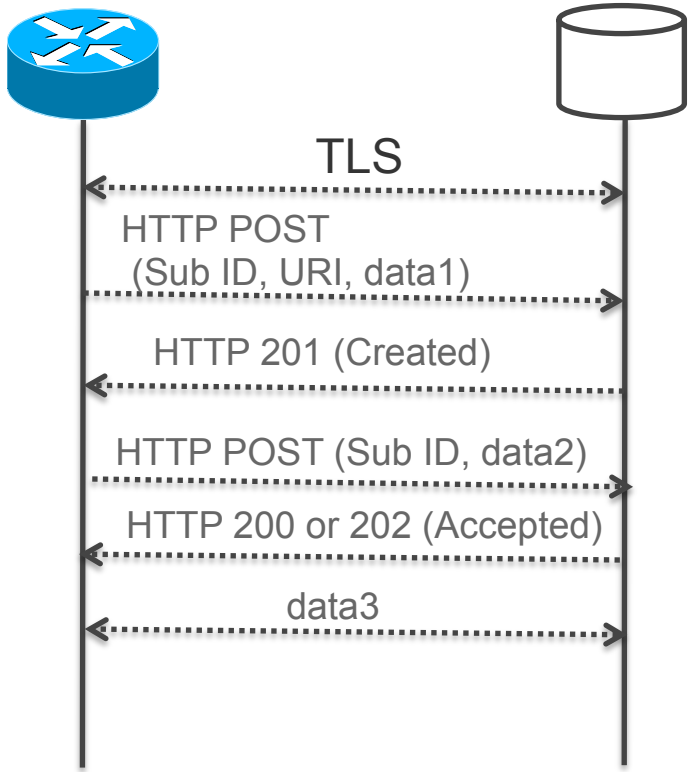
Subscribing to YANG datastore push updates

Periodic Subscriptions

On-Change Subscriptions

- I2RS YANG Subscription Requirements
- **Subscriber**:YANGデータ・モデルを要求する Subscriptionモデルのオーナ
- **Publisher**: SubscribeされたYANGデータ・モデルを各Subscriberに配布する。データモデルのオーナ

Restconf subscription and HTTP push for YANG datastores
draft-voit-netconf-restconf-yang-push



- HTTP/2の活用
  - ✓ 多重化（multiplex）
  - ✓ ストリームのPrioritization
  - ✓ フローコントロール
  - ✓ ヘッダー圧縮

Agenda

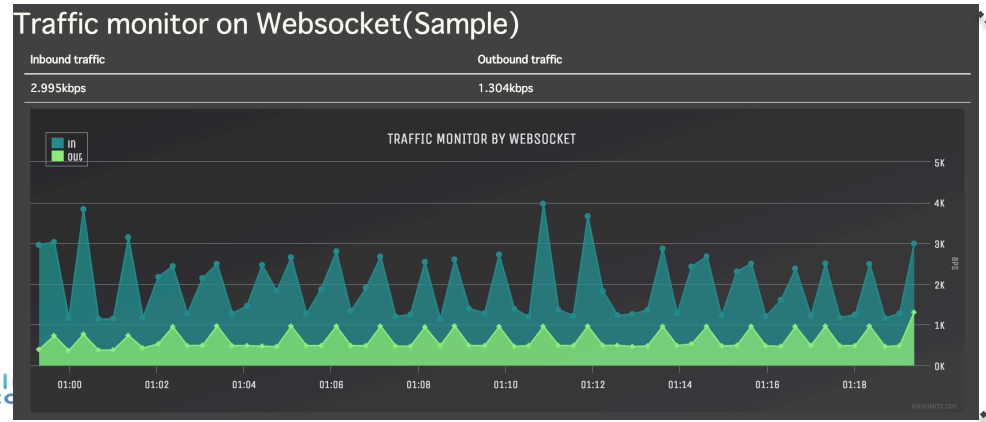
- テレメトリー登場背景
- エンコード手法(誰が進めているのか)
- 日本の運用者におけるテレメトリー例
- 実装例

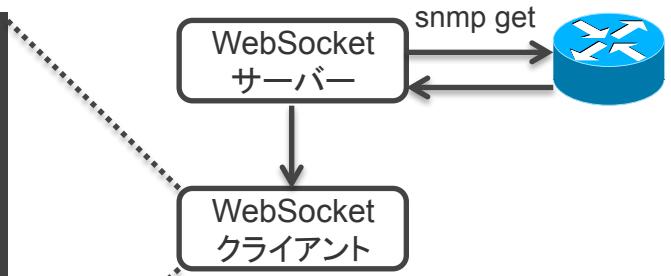
Microsoftにおけるネットワーク自動化とそれを支えるソフトウェア群について
http://www.slideshare.net/netopscoding/microsoft-54939067

- マイクロソフトにおけるコンフィグ作成の自動化
- 使用帯域を元にしたLSPの自動調整機能とルータ側Script
- P23　Monitoring Intelligence以降を参照
- SNMP/Syslog/CLI/xFlowなどのデータ元にAzure HDInsightなどで解析
- 相関関係により、障害の切り分けや問題リンクの特定などを実施

トラフィックをWebSocketでリアルタイムに描写してみた
http://qiita.com/Mabuchin/items/135a9edff34a3db00744

- 新しい回線開通やトラフィック調整作業時にはCLIでレートをモニター
- 複数箇所の状態変化に気づきずらい
- WebSocketサーバーにて対象機器のSNMPをポーリング
- WebSocketクライアントでそれらの情報を元にグラフを描画





xFlowデータの活用

- docker + fluentd + Elasticsearch + kibana4 で構築するお手軽NetFlowアナライザ

  http://qiita.com/skjune12/items/d88a8eb32794865afcd3

- フロー見てみた

  http://www.janog.gr.jp/meeting/janog34/doc/janog34-lt4bg-yoshino-1.pdf

  fluentd+Norikra+GrowthForecast

- オープンソースでキメるDDoSトラヒック分析

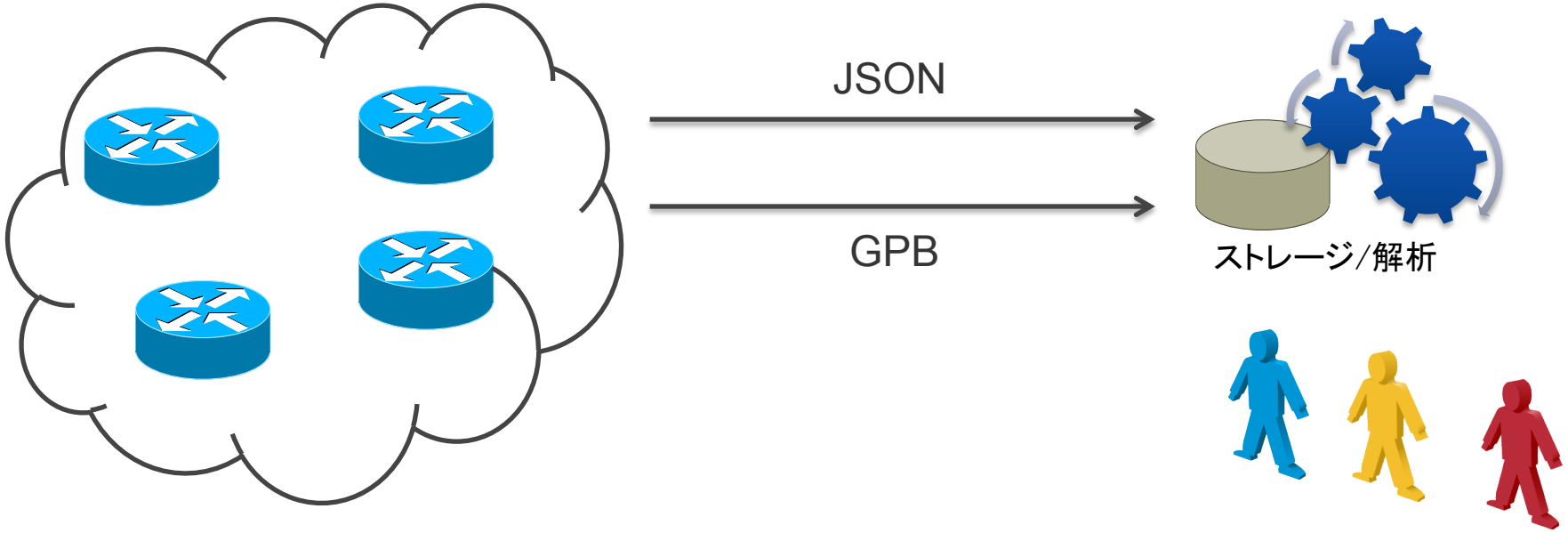
  http://www.janog.gr.jp/meeting/janog34/program/lt_tanal.html

  Flowデータ->MySQL->Google Chart

Agenda

- テレメトリー登場背景
- エンコード手法(誰が進めているのか)
- 日本の運用者におけるテレメトリー例
- 実装例

IOS-XR Telemetry



- ルータが保持するデータスキマーよりエンコードを作成し、JSON/GPB フォーマットでの送信が可能

Cisco IOS XR Telemetry

http://www.cisco.com/c/en/us/td/docs/iosxr/Telemetry/Telemetry-Config-Guide.html

```
RP/0/RP0/CPU0:XRv9K#show running-config telemetry
Sat Jan 16 06:26:59.525 UTC
telemetry
 encoder gpb
  policy group GPB
   policy BGP
   policy Traffic
   policy Inventory
   destination ipv4 10.71.134.83 port 2103
  !
  policy group TrafficENG
   policy Traffic
   destination ipv4 10.71.134.85 port 2103
  !
 !
 encoder json
  policy group JSON
   policy BGP
   policy Traffic
   policy Inventory
   destination ipv4 10.71.134.83 port 2103
   destination ipv4 172.16.1.253 port 6000
  !
```

- 複数のポリシーをグループとしてまとめる
- XR6.0ではGPB(UDP)/JSON(TCP)エンコードをサポート
- 異なるポリシーグループで送信先を分ける
- 同じポリシーを複数の宛先に送信可能

policyファイル

```
RP/0/RP0/CPU0:XRv9K#run cat  /telemetry/policies/Traffic.policy
Sat Jan 16 06:43:15.865 UTC
{
"Name": "Traffic",
"Metadata": {
    "Version": 1,
    "Description": "Traffic Monitor Sample",
    "Comment": "JANOG37 Sample",
    "Identifier": "<data that may be sent by the encoder to the mgmt stn>"},
"CollectionGroups": {
    "FirstGroup": {
      "Period": 30,
      "Paths": [
        "RootOper.InfraStatistics.Interface(*).Latest.DataRate",
        "RootOper.InfraStatistics.Interface(*).Latest.GenericCounters"
      ]
  }
}
}
```

- /telemetry/policies/以下に*.policyで定義ファイルを作成
- サポートされている全てのXMLスキマーが参照可能

Receiver

```
Message length = 825
unpacking TLV
TLV type = 2
Got message
Message keys:
    Policy:"Traffic"
    End Time:1452925081946
    Identifier:"<data that may be sent by the encoder to the mgmt stn>"
    Data:{
    "RootOper": {
        "InfraStatistics": {
…
            "GigabitEthernet0/0/0/1": {
                "Latest": {
                    "DataRate": {
                        "PeakInputPacketRate": 0,
                        "PeakOutputDataRate": 0,
                        "InputLoad": 0,
                        "InputPacketRate": 2742,
                        "LoadInterval": 0,
                        "OutputDataRate": 1233222,
                        "PeakOutputPacketRate": 0,
                        "PeakInputDataRate": 0,
                        "OutputPacketRate": 103123,
                        "OutputLoad": 255,
                    ....
                      "LastDiscontinuityTime": 1452889495,
                        "BytesSent": 1829305882789
                    }
                }
            }
```

- .policyで定義されたJSONフォーマットで受信されてる事を確認

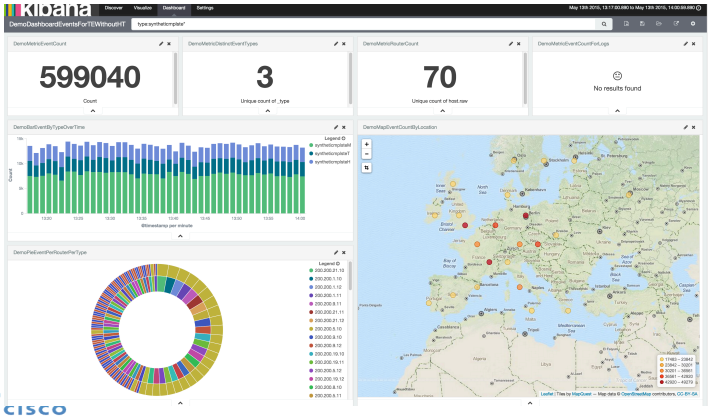
Streaming Telemetry Collector Stacks

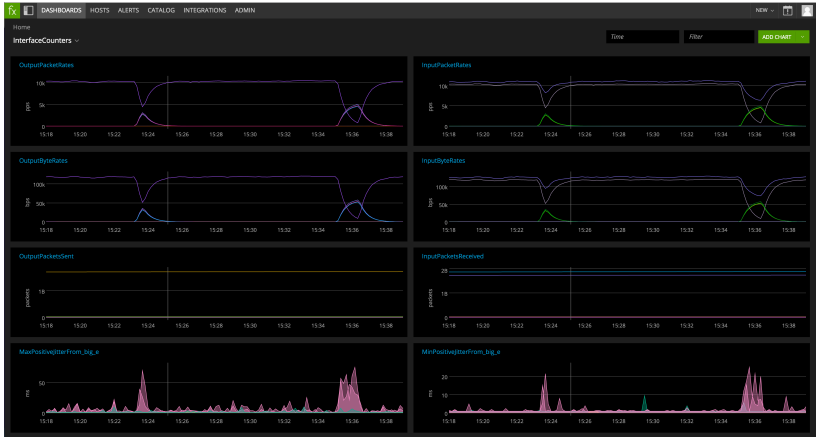
https://github.com/cisco/bigmuddy-network-telemetry-stacks

- それぞのメッセージフォーマットは記載があるがサンプルとしてgit公開

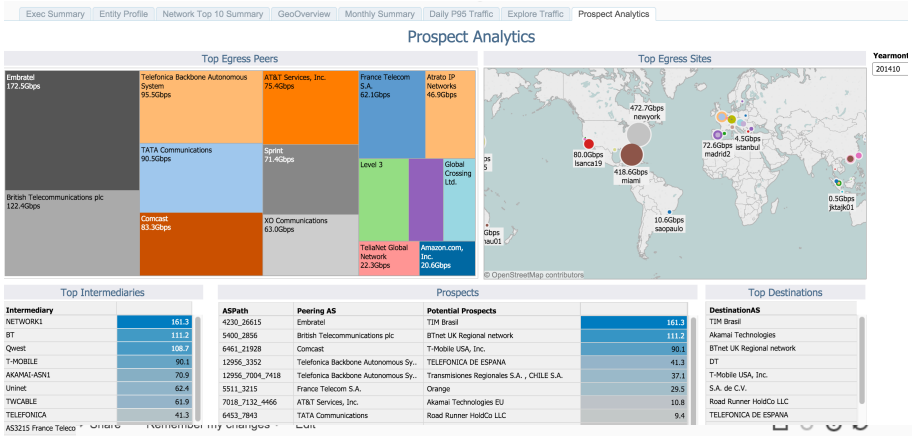
3つのdockerが起動

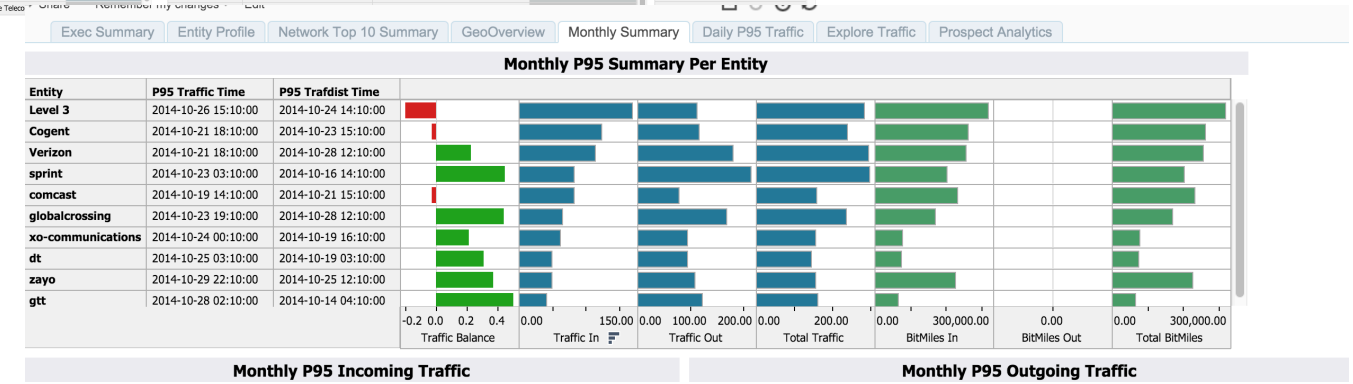
- stack_elk …elaticsearch/logstash/kibana
- stack_prometheus…logstash/prometheus/pushgateway/promdash
- stack_signalfx…signal fx





Cisco Network Business Intelligence





- xFlowデータ/トポロジー情報/回線コストなどを合わせて、トラフィックボリュームからのOPEXなどを計算
- pmacctおよびhadoop/TABLEAUを使用

スキーマーパス設定例

Interface Operational data	RootOper.Interfaces.Interface(*)
Packet/byte counters	RootOper.InfraStatistics.Interface(*).Latest.GenericCounters
Packet/byte rates	RootOper.InfraStatistics.Interface(*).Latest.DataRate
IPv4 packet/byte counters	RootOper.InfraStatistics.Interface(*).Latest.Protocol(['IPV4_UNICAST'])
MPLS stats	RootOper.MPLS_TE.Tunnels.TunnelAutoBandwidth RootOper.MPLS_TE.P2P_P2MPTunnel.TunnelHead RootOper.MPLS_TE.SignallingCounters.HeadSignallingCounters
QOS Stats	RootOper.QOS.Interface(*).Input.Statistics RootOper.QOS.Interface(*).Output.Statistic
BGP Data	RootOper.BGP.Instance({'InstanceName': 'default'}).InstanceActive.DefaultVRF.Neighbor([*])

まとめ

- Hadoop/Kibanaなどビックデータ解析ツールが台頭
- ネットワーク機器より解析ツールに直接データをプッシュが可能になる
- 上手に使う事で障害解析ツール・キャパシティプランニングにも使える

議論のポイント

- 何を見たい?　or 見てる？
- エンコード方式は?(GPB/JSON/Thrift)
- 今の運用・監視で辛い所(CLIで無ければ見れない所)あったら教えて!!

- などなど

TOMORROW starts here.