自動化、AI、その前に、**あなたのネットワーク見えてますか？**

2024年1月

今回の発表は企業の買収とは全く関係ありません

splunk> turn data into doing™

池永 隆次郎

Splunk Services Japan合同会社
インダストリーアドバイザー

経歴：初JANOG

シスコシステムズ合同会社

NTT、コナミデジタルエンターテインメント

本日皆さんと話したいこと

- Telemetryの活用
- ネットワークの可視化と
  マルチレイヤーデータ連携
- 現実的な運用を考えた場合、
  どこまで追求する？

本日の内容

メーカー視点でお話ししますが、是非運用に携わる方々の率直な意見やコメントをいただき、熱い議論ができることを楽しみにしております。

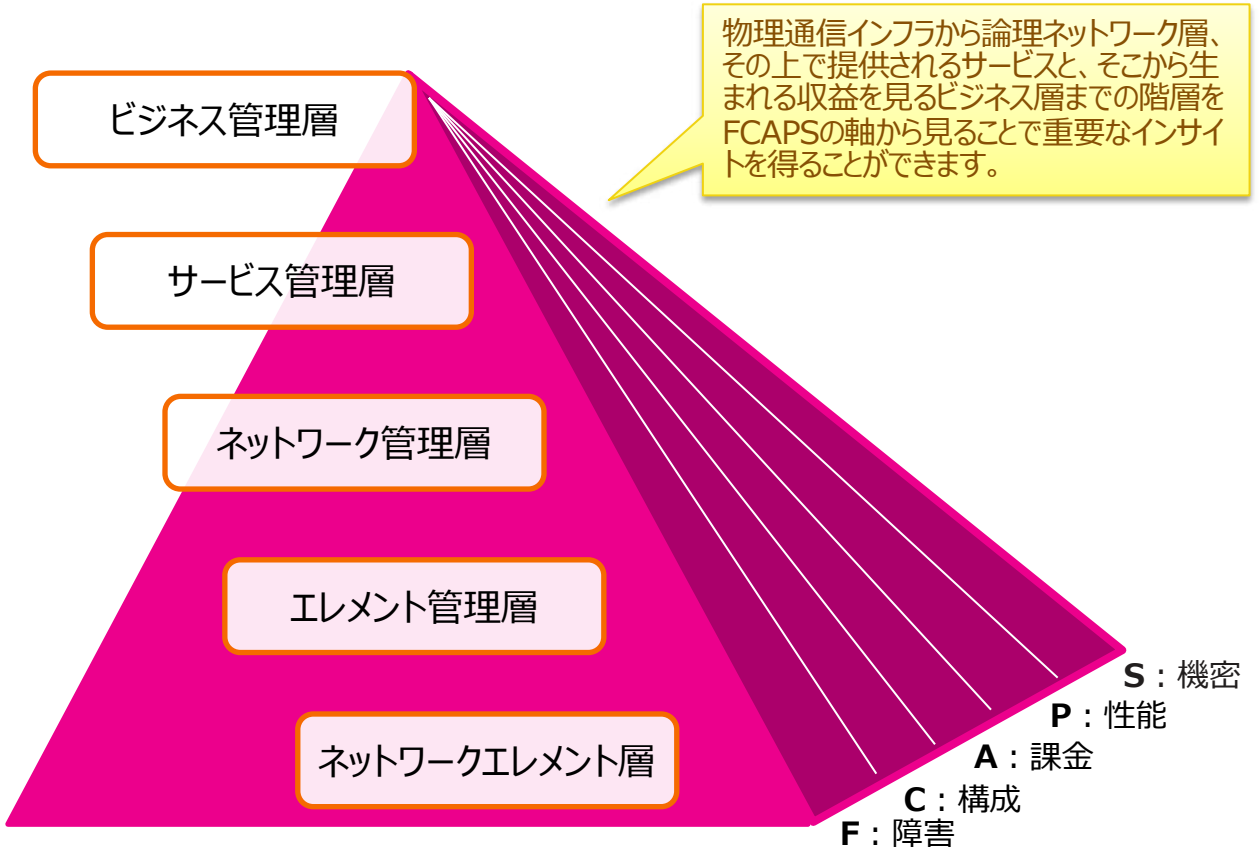
データを活用しましょう！	池永
ネットワークの可視化とは	鎌田
ネットワークデータの活用	池永
デモ	鎌田

データを活用しましょう！

通信ビジネスのスタック全体で価値を向上

- ✓ ネットワークエレメント層からビジネス管理層まで、通信ビジネスのスタック全体を通じてFCAPS (障害、構成、課金、性能、機密)に関するインサイトを獲得する
- ✓ 組織全体であらゆる問題解決と意思決定にデータを活用して行動につなげる
- ✓ データサイロを解消し、誰もがデータにアクセスして、その有用性と価値を引き出せるようにする

データの集約と民主化が実現のキーです。



FCAPS: **F**ault, **C**onfiguration, **A**ccounting, **P**erformance and **S**ecurity

求められる機能

通信サービスプロバイダーの変革を支援し、あらゆる側面で価値向上を後押しする、プラットフォーム

- レイヤーを跨ぐ複雑な通信インフラの管理
- 故障や顧客からのクレームと通信インフラの状況の紐付け
- 需要を予測して拡張工事を先行一括手配
- その他は通信キャリアに限らず価値のある点

広範で強力・柔軟なプラットフォーム

ニアリアルタイム: ニアリアルタイムでストリーミングデータ を処理して分析し、導出した分析結果をインタラクティブで使いやすいプラットフォームで共有

異なる層をまたいだ関連付けと自動化: 統合的な分析によってサービスのエクスペリエンスをネットワークインフラと関連付け、適切なアクションを自動的に実行

データへのアクセス: インサイトや分析結果を異なる部門間で共有して活用て

あらゆるデータ: あらゆるソースからもたらされる、さまざ まな構造のデータを取り込んで活用し、データサイロを統合

将来を見据えたプラットフォーム: 機械学習などのAI機能を新たに組み込み、さらなる高度化を行うことが可能

splunk> turn data into doing

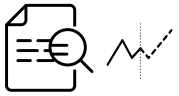
統合レジリエンス・プラットフォーム

Out of box

分析サービス


Ad hoc Search


Monitor and Alert


Reports/ Analyze


Custom Dashboards


Integrate Business Systems


NW担当者


情報セキュリティ担当者


ビジネスアナリスト


経営層


基幹システム担当者

```
sourcetype=network | stats avg(payment) by generation
```

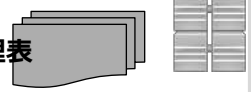
Search

統合プラットフォーム

splunk> Operational Intelligence Platform

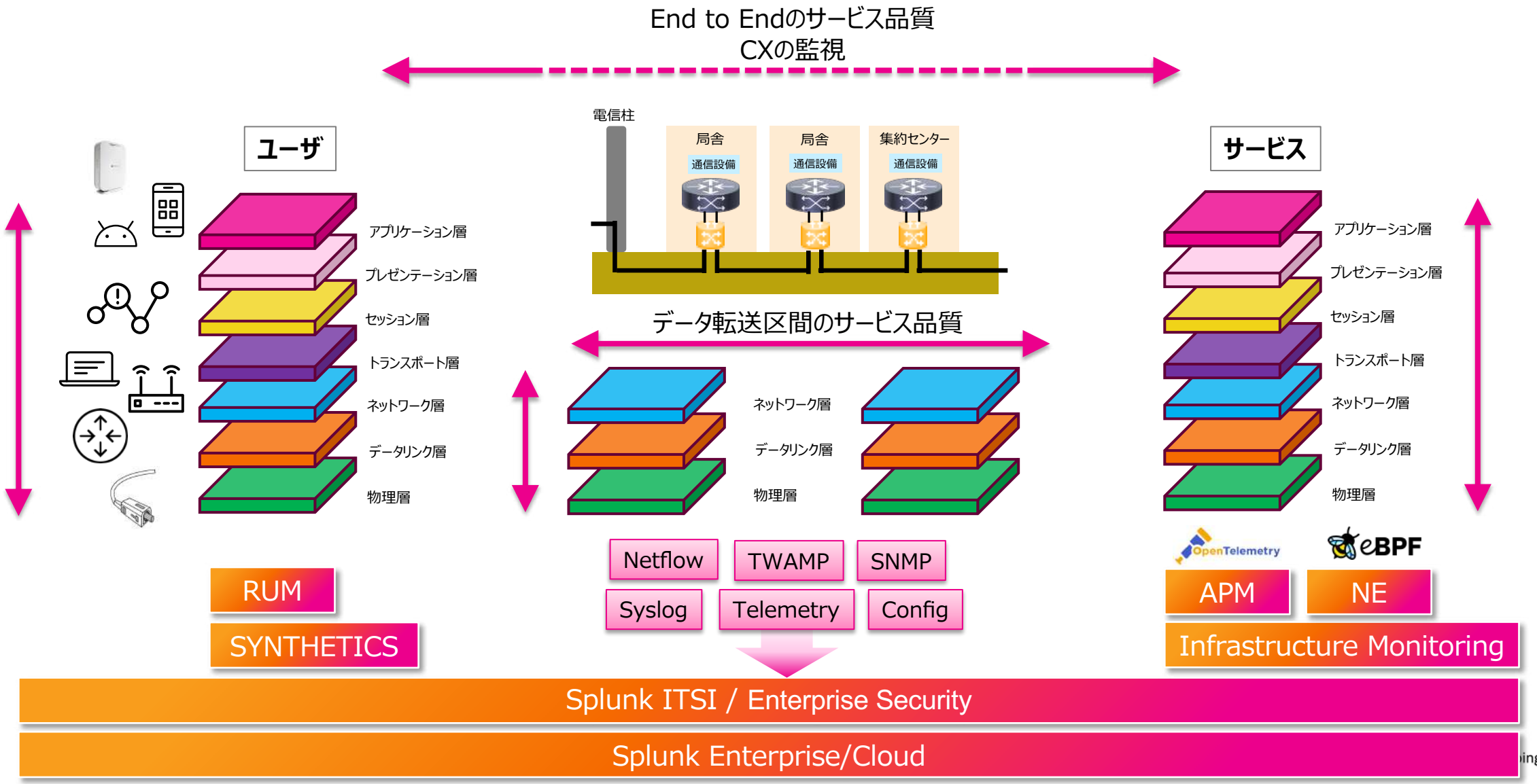
収集 | データストア | 検索インターフェイス | 機械学習 | アラート | 可視化 | 統合・連携

データサイロ

資産管理・構成情報	ネットワークデータ	クラウド・ホストデータ	アプリケーションデータ	ユーザ情報	外部情報
構成図  試算管理表 	SNMP シスログ Telemetry Netflow 	Telemetry シスログ SNMP  	 Telemetry Metric/Trace/ログ	ID情報 契約情報 コールセンター履歴  	  気象情報

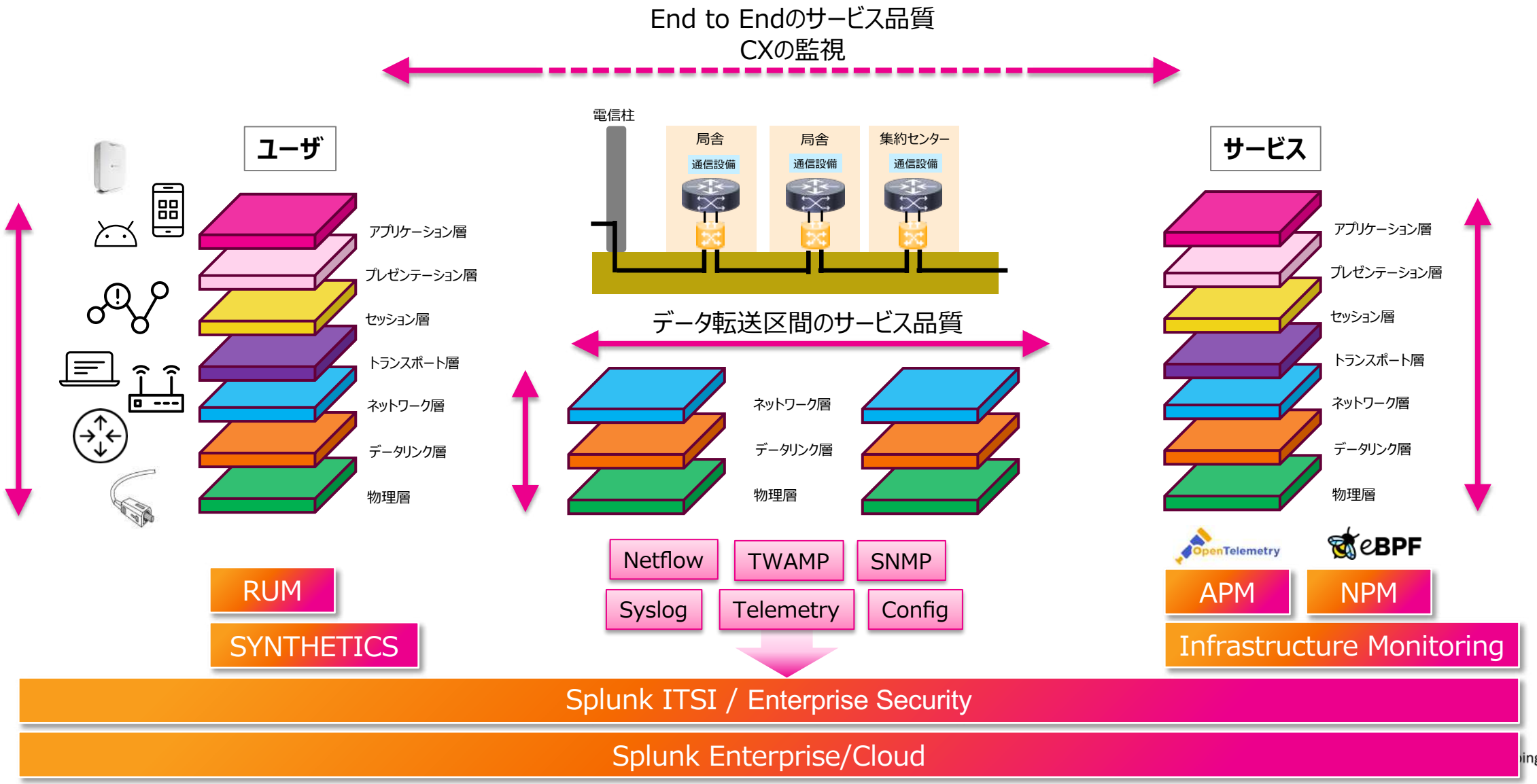
splunk> turn data into doing

Full Stack Observability
横(End-End)と縦(Multi-Layer)の可視化/調査/分析



ネットワークデータの活用

Full Stack Observability
横(End-End)と縦(Multi-Layer)の可視化/調査/分析



カスタマーエクペリエンスの向上・改善

エンドツーエンドのネットワークの状況を把握

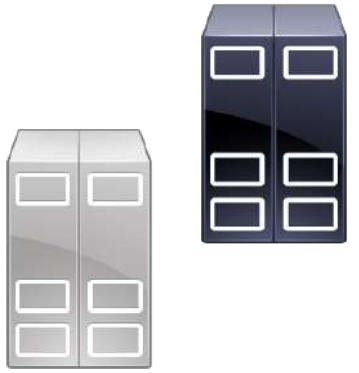
ユーザ

繋がらない

遅い
ブロックノイズ

再接続

サービス



VoD画面にノイズが出る
ECサイトでの買い物がエラーになる

何か把握できていない事象が発生しているかも・・・

エラーは出ていない
クライアント側の応答がない

カスタマーエクペリエンスの向上・改善

エンドツーエンドのネットワークの状況を把握

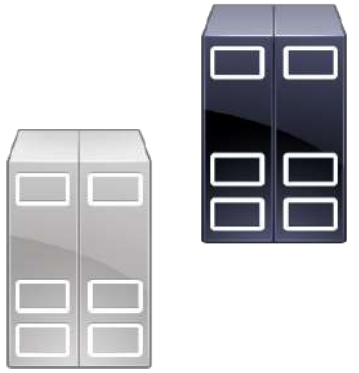
ユーザ

通信機器でのサイレント障害

バースト
パケットロス

再接続

サービス



何か把握できていない事象が発生しているかも・・・

Netflow　TWAMP　SNMP
Syslog　Telemetry　Config

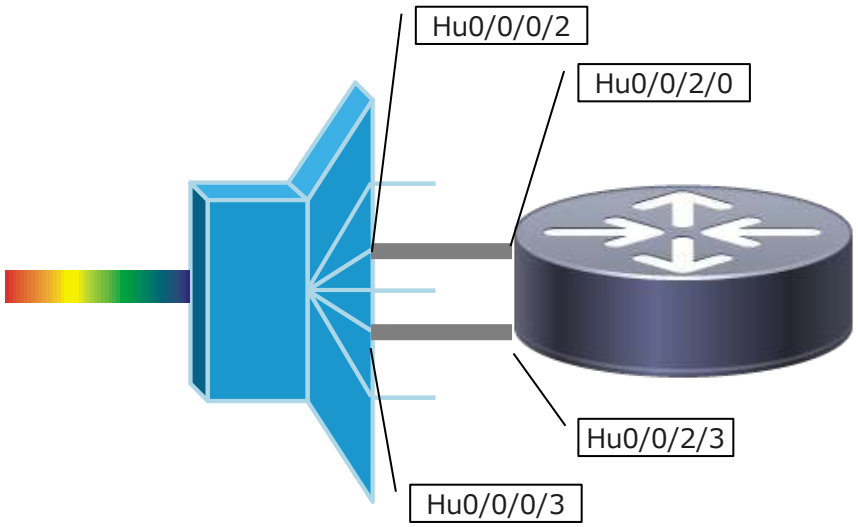
VoD画面にノイズが出る
ECサイトでの買い物がエラーになる

Observability

エラーは出ていない
クライアント側の応答がない

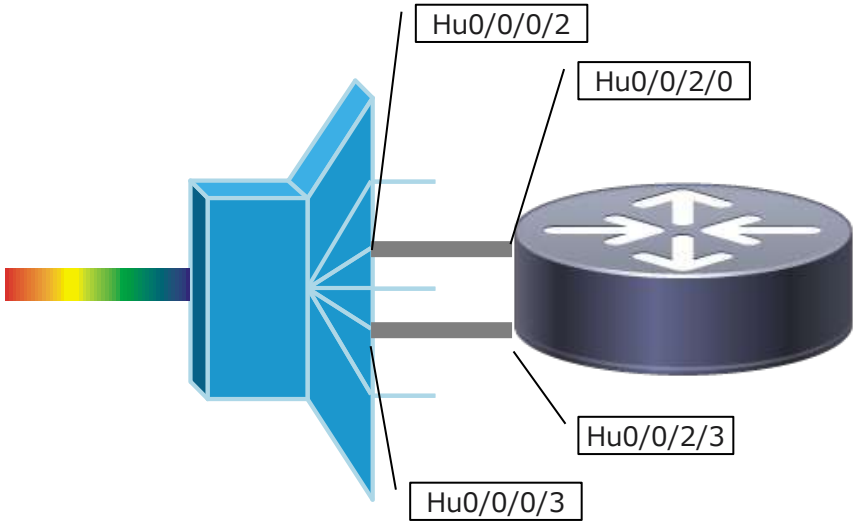
WDMとルータの接続確認

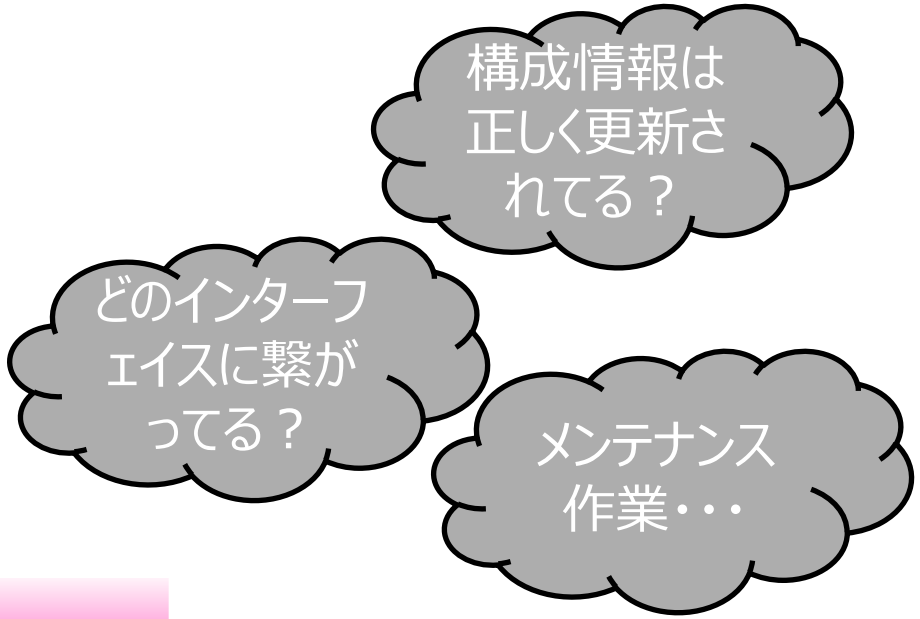
WDMとルータは透過的な接続のためリモートでの接続確認が難しい



WDMとルータの接続確認

WDMとルータは透過的な接続のためリモートでの接続確認が難しい





マルチレイヤー間の通信機器の接続情報を正確に管理
イベントや障害発生時の基本的な影響範囲を迅速に把握

ネットワークデータの活用

1. メトリックデータの相関性
2. インターフェイスのDOWN/UPのログをデータ化
3. スクリプトによるデータ取得対応

メトリックデータの相関性の有無

メトリックデータの相関分析

- CPU使用率などのメトリックデータ同士の相関係数を算出し、相関関係の有無を確認
- たとえば、とある項目 (e.g. 通信量、ルーティングテーブルサイズ) とCPU使用率には実は相関があり、その相関係数は0.9 (正の強い相関) である、など

カスタムサーチコマンド

Pearsonおよび相互相関関数 (xcorr) を利用可能に

相関係数rおよびp値の算出

CPU使用率と相関があり、相関度の強いKPIを特定

インタラクティブなダッシュボード

任意のノードおよびKPIを選択し、その他KPIとの相関係数を全出力可能

カスタムサーチコマンドの実装

Pearson

pearsonr: CUP使用率	p_value: KPI 1
0.957932734404477	0.0

- Pearson相関係数およびp値を算出可能
- scipy.stats.pearsonr を実装
- 複数の変数(KPI値)を指定可能

相互相関関数

100件/ページ ▾ フォーマット プレビュー ▾

Lag ≑	CUP使用率 vs_ KPI 1
-2	0.9396290247195543
-1	0.9537630768798065
0	0.9579327344044769
1	0.9498555682038221
2	0.9344362104027227

- ラグ毎の相関係数を算出可能
- matplotlib.pyplot.xcorr を実装
- 複数の変数(KPI値)を指定可能

splunk>

CPU使用率とその他KPIの相関分析結果 - 例

Pearson相関係数r

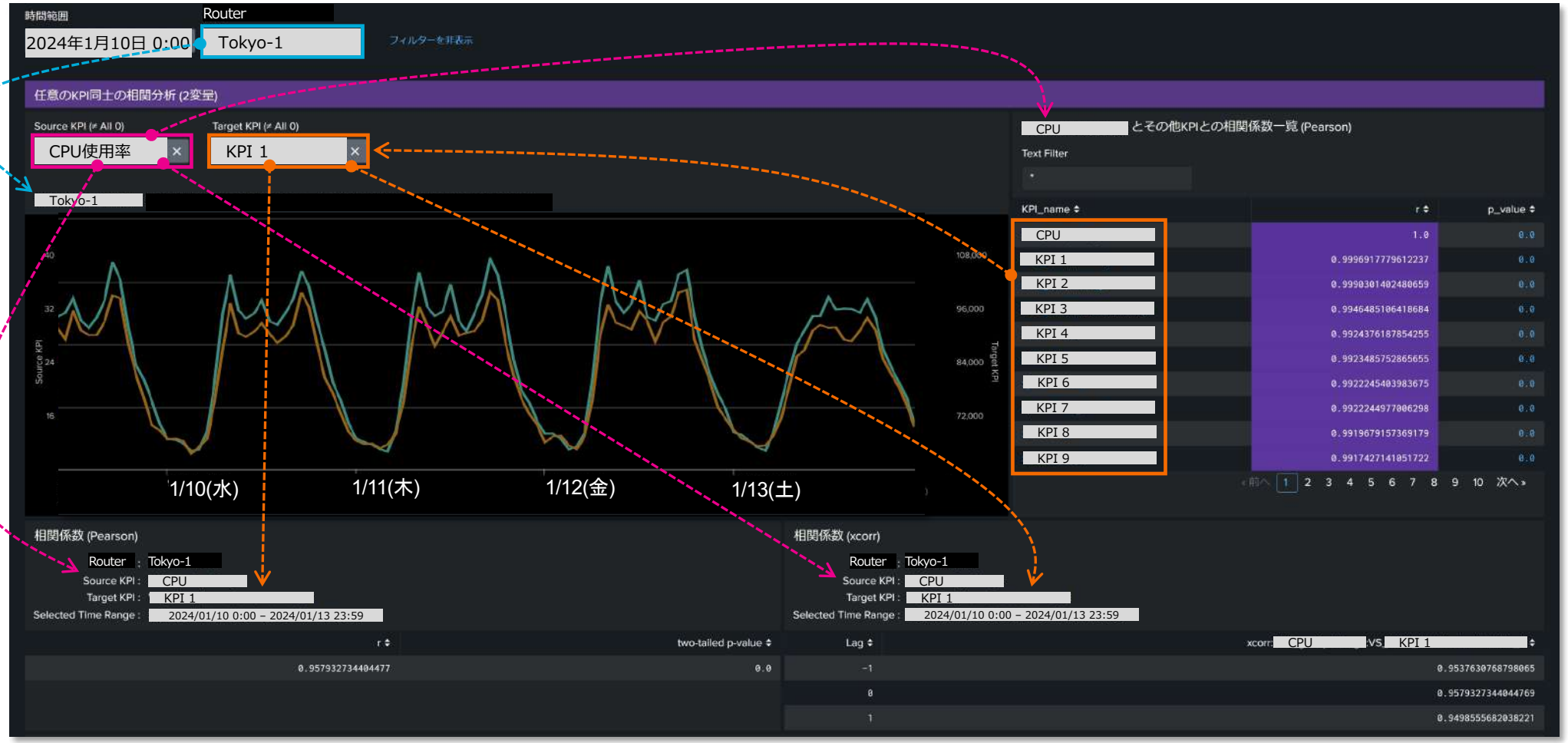
KPI_name	r	p_value
CPU使用率	1.0	0.0
KPI 1	0.9996917779612237	0.0
KPI 2	0.9990301402480659	0.0
KPI 3	0.9946485106418684	0.0
KPI 4	0.9924376187854255	0.0
KPI 5	0.9923485752865655	0.0
KPI 6	0.9922245403983675	0.0
KPI 7	0.9922244977006298	0.0
KPI 8	0.9919679157369179	0.0
KPI 9	0.9917427141051722	0.0
KPI 10	0.990824683882667	0.0
KPI 11	0.9906947580921798	0.0
KPI 12	0.9905531502430686	0.0

- 相関係数r は-1～1の範囲の値をとり、1に近づくほど正の相関が強い（Aが増えればBが増える）、-1に近づくほど負の相関が強い（Aが増えればBが減る）
- p値は統計学的に有意かどうか判断するための値であり、慣習的に0.05 (5%) 未満であれば有意差 (つまり、相関) があるとされる

※ 値がすべて0やnull、もしくは常に値が一定のKPIの場合は相関係数はnan表示となります

splunk>

KPI値 - 相関分析ダッシュボード



インターフェイスのDOWN/UPのログをデータ化

レイヤーを跨ぐ装置のインターフェイスステイタスのログを集計する

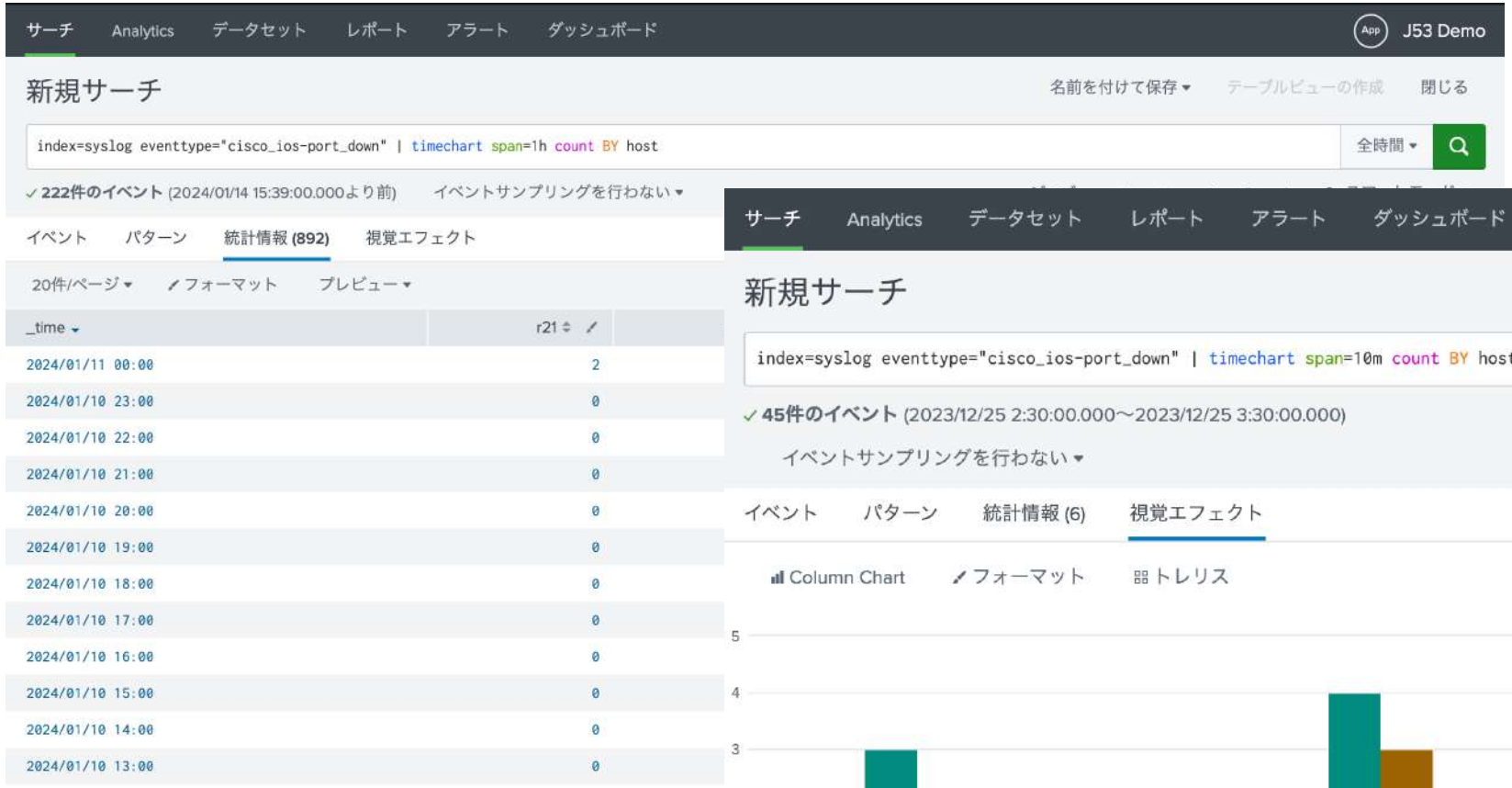
短時間でこんなたくさんのログが出ます。論理・バンドル・サブインターフェイス、ルーティングやトラシーバーのログを含めるともっとたくさんある。大変だ・・・

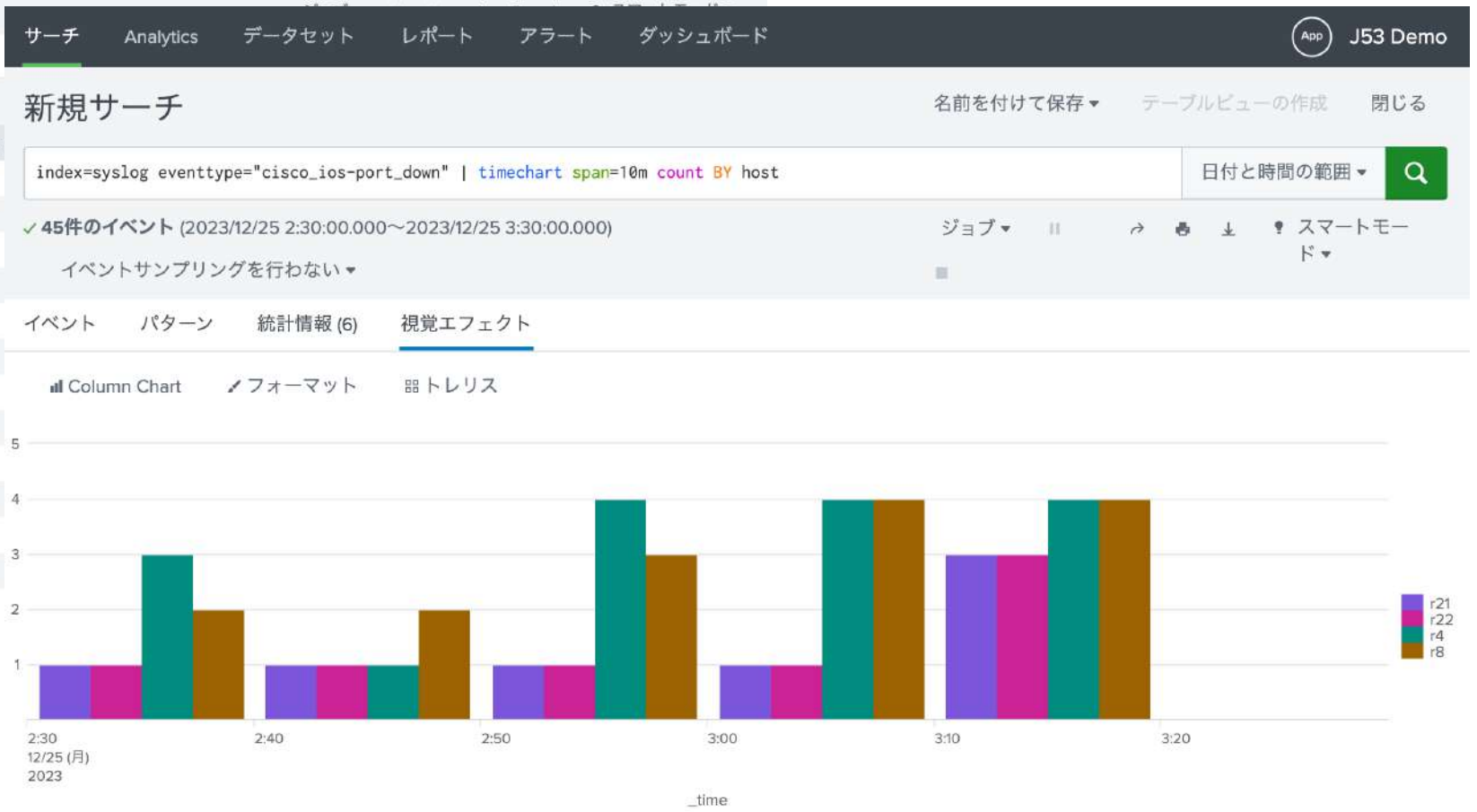
```
R4 - Jan 5 06:48:51 r4.5gsc.lab 1578774: LC/0/0/CPU0:Jan 5 15:49:04.005 JST: ifmgr[181]: %PKT_INFRA-LINK-3-UPDOWN : Interface FortyGigE0/0/2/3, changed state to Down
R4 - Jan 5 06:48:51 r4.5gsc.lab 1578776: LC/0/0/CPU0:Jan 5 15:49:04.005 JST: ifmgr[181]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface FortyGigE0/0/2/3, changed state to Down
R21 - Jan 5 06:54:14 172.20.0.21 682: RP/0/RP0/CPU0:Jan 5 15:54:27.811 JST: osa_driver[346]: %PKT_INFRA-FM-3-FAULT_MAJOR : ALARM_MAJOR :SIGLOSS :DECLARE :HundredGigECtrlr0/0/0/3:
R22 - Jan 5 06:54:35 172.20.0.22 584: RP/0/RP0/CPU0:Jan 5 15:54:48.768 JST: osa_driver[182]: %PKT_INFRA-FM-3-FAULT_MAJOR : ALARM_MAJOR :SIGLOSS :DECLARE :HundredGigECtrlr0/0/0/3:
R4 - Jan 5 06:54:15 r4.5gsc.lab 1578794: LC/0/0/CPU0:Jan 5 15:54:28.137 JST: ifmgr[181]: %PKT_INFRA-LINK-3-UPDOWN : Interface HundredGigE0/0/2/3, changed state to Down
R4 - Jan 5 06:54:15 r4.5gsc.lab 1578796: LC/0/0/CPU0:Jan 5 15:54:28.137 JST: ifmgr[181]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface HundredGigE0/0/2/3, changed state to Down
R22 - Jan 5 06:54:35 172.20.0.22 584: RP/0/RP0/CPU0:Jan 5 15:54:48.768 JST: osa_driver[182]: %PKT_INFRA-FM-3-FAULT_MAJOR : ALARM_MAJOR :SIGLOSS :DECLARE :HundredGigECtrlr0/0/0/3:
R5 - Jan 5 06:59:08 r5.5gsc.lab 1605058: LC/0/0/CPU0:Jan 5 15:59:21.887 JST: ifmgr[134]: %PKT_INFRA-LINK-3-UPDOWN : Interface HundredGigE0/0/2/3, changed state to Down
R5 - Jan 5 06:59:08 r5.5gsc.lab 1605060: LC/0/0/CPU0:Jan 5 15:59:21.887 JST: ifmgr[134]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface HundredGigE0/0/2/3, changed state to Down
R4 - Jan 5 06:59:08 r4.5gsc.lab 1578862: LC/0/0/CPU0:Jan 5 15:59:21.887 JST: ifmgr[181]: %PKT_INFRA-LINK-3-UPDOWN : Interface HundredGigE0/0/2/3, changed state to Down
R4 - Jan 5 06:59:08 r4.5gsc.lab 1578864: LC/0/0/CPU0:Jan 5 15:59:21.887 JST: ifmgr[181]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface HundredGigE0/0/2/3, changed state to Down
R22 - Jan 5 06:59:13 172.20.0.22 606: RP/0/RP0/CPU0:Jan 5 15:59:26.772 JST: osa_driver[182]: %PKT_INFRA-FM-6-FAULT_INFO : OPUK-CSF :DECLARE :ODU40/0/0/0/2:
R21 - Jan 5 06:59:13 172.20.0.21 708: RP/0/RP0/CPU0:Jan 5 15:59:26.812 JST: osa_driver[346]: %PKT_INFRA-FM-3-FAULT_MAJOR : ALARM_MAJOR :SIGLOSS :DECLARE :HundredGigECtrlr0/0/0/3:
R4 - Jan 5 06:59:30 r4.5gsc.lab 1578874: LC/0/0/CPU0:Jan 5 15:59:43.781 JST: ifmgr[181]: %PKT_INFRA-LINK-3-UPDOWN : Interface HundredGigE0/0/2/0, changed state to Down
R4 - Jan 5 06:59:30 r4.5gsc.lab 1578876: LC/0/0/CPU0:Jan 5 15:59:43.781 JST: ifmgr[181]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface HundredGigE0/0/2/0, changed state to Down
R5 - Jan 5 06:59:30 r5.5gsc.lab 1605070: LC/0/0/CPU0:Jan 5 15:59:43.783 JST: ifmgr[134]: %PKT_INFRA-LINK-3-UPDOWN : Interface HundredGigE0/0/2/3, changed state to Down
R5 - Jan 5 06:59:30 r5.5gsc.lab 1605072: LC/0/0/CPU0:Jan 5 15:59:43.783 JST: ifmgr[134]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface HundredGigE0/0/2/3, changed state to Down
R8 - Jan 5 06:59:30 r8.5gsc.lab 18068: RP/0/RP0/CPU0:Jan 5 15:59:43.785 JST: ifmgr[298]: %PKT_INFRA-LINK-3-UPDOWN : Interface HundredGigE0/0/0/29, changed state to Down
R8 - Jan 5 06:59:30 r8.5gsc.lab 18070: RP/0/RP0/CPU0:Jan 5 15:59:43.785 JST: ifmgr[298]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface HundredGigE0/0/0/29, changed state to Down
R4 - Jan 5 06:59:30 r4.5gsc.lab 1578880: LC/0/0/CPU0:Jan 5 15:59:43.788 JST: ifmgr[181]: %PKT_INFRA-LINK-3-UPDOWN : Interface HundredGigE0/0/2/3, changed state to Down
R4 - Jan 5 06:59:30 r4.5gsc.lab 1578882: LC/0/0/CPU0:Jan 5 15:59:43.788 JST: ifmgr[181]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface HundredGigE0/0/2/3, changed state to Down
```

とりあえず、データを生ログのまま取り込んで、あれやこれやとサーチしながら意味にあるデータにしてみる！

インターフェイスのDOWN/UPのログをデータ化

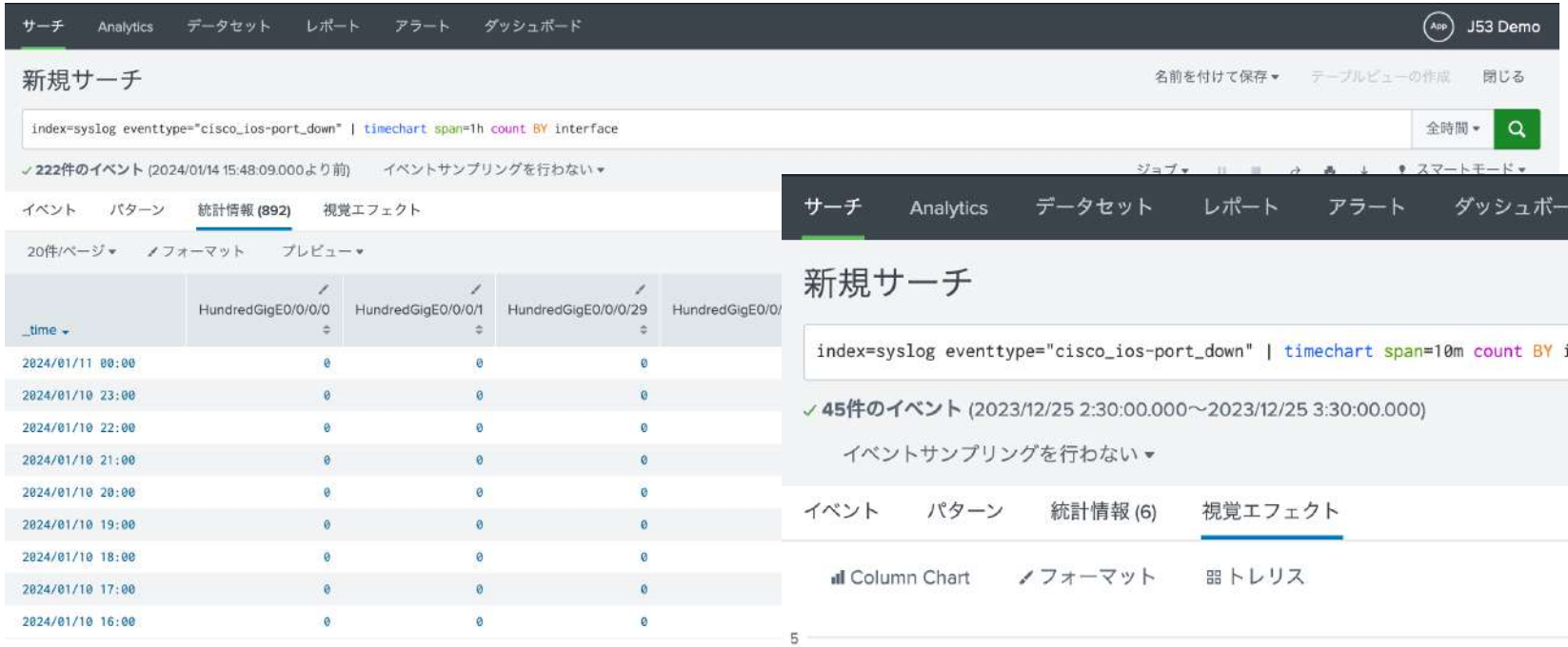
王道のリンクダウンのログをポートダウンのイベントとしてまとめてノード毎にDownイベントを集計してみた。

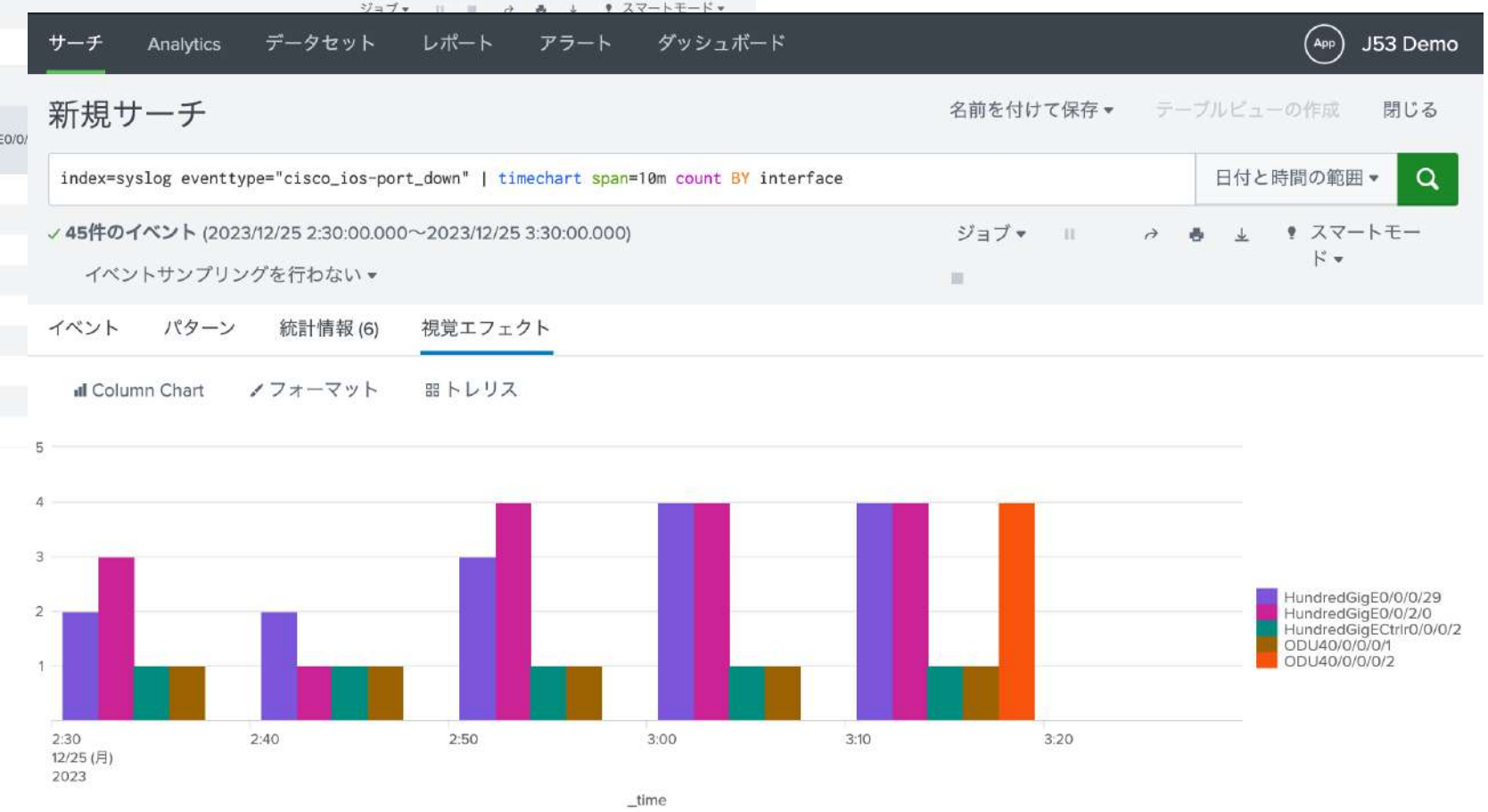




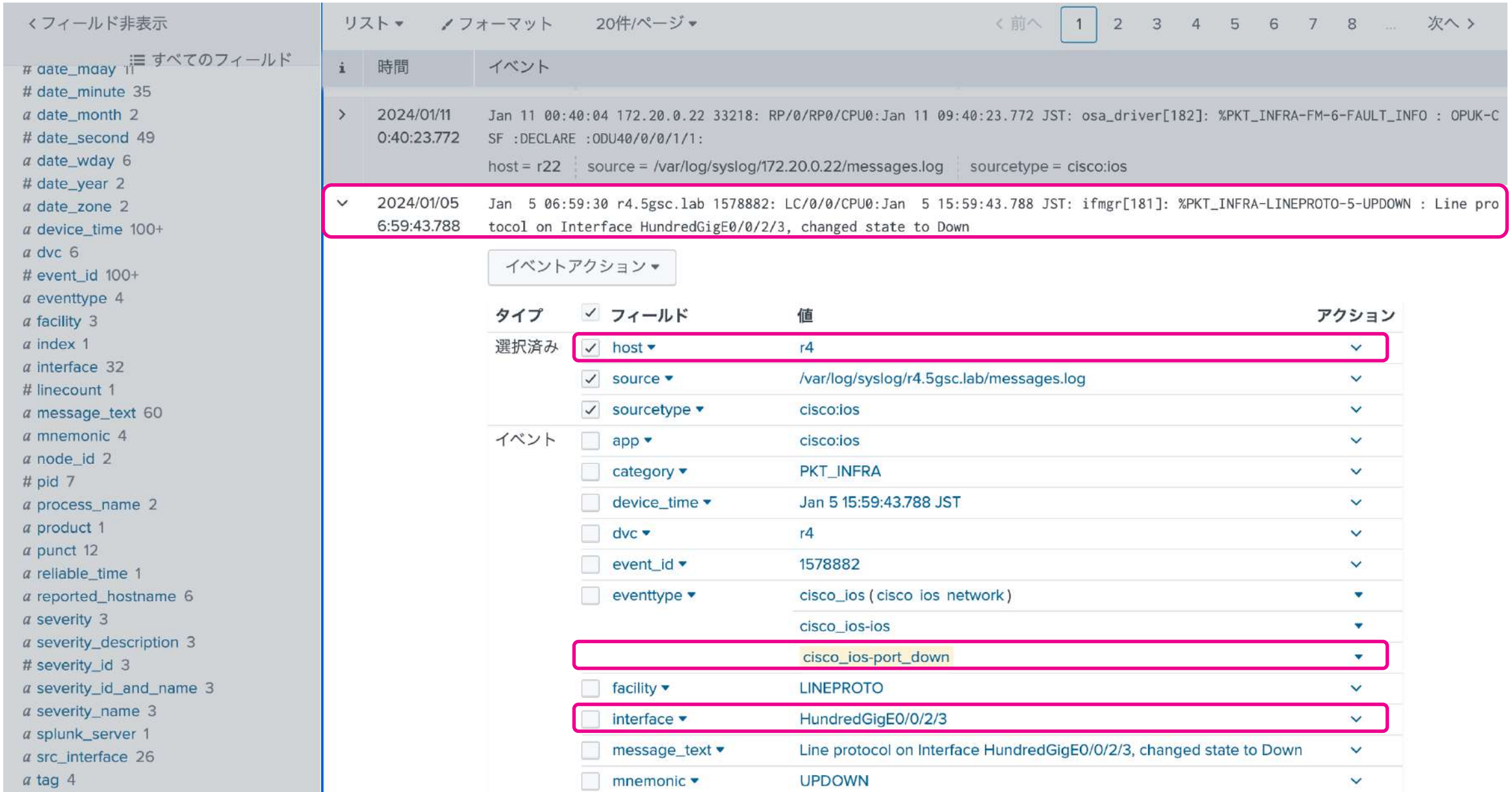
インターフェイスのDOWN/UPのログをデータ化

リンクダウンのログをポートダウンのイベントとしてまとめてインターフェイス毎にDownイベントを集計してみた。

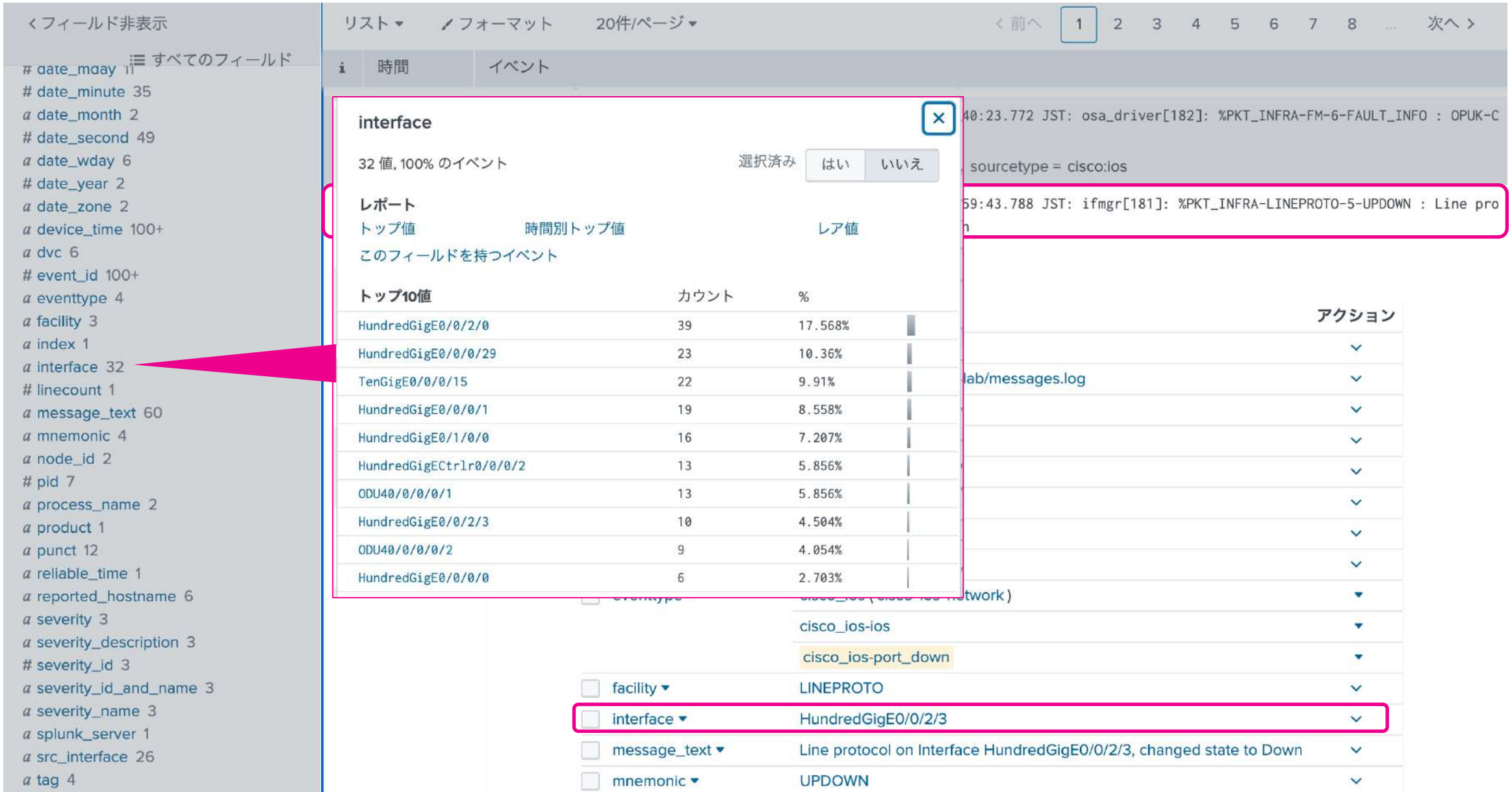




インターフェイスのDOWN/UPのログをデータ化



インターフェイスのDOWN/UPのログをデータ化



インターフェイスのDOWN/UPのログをデータ化

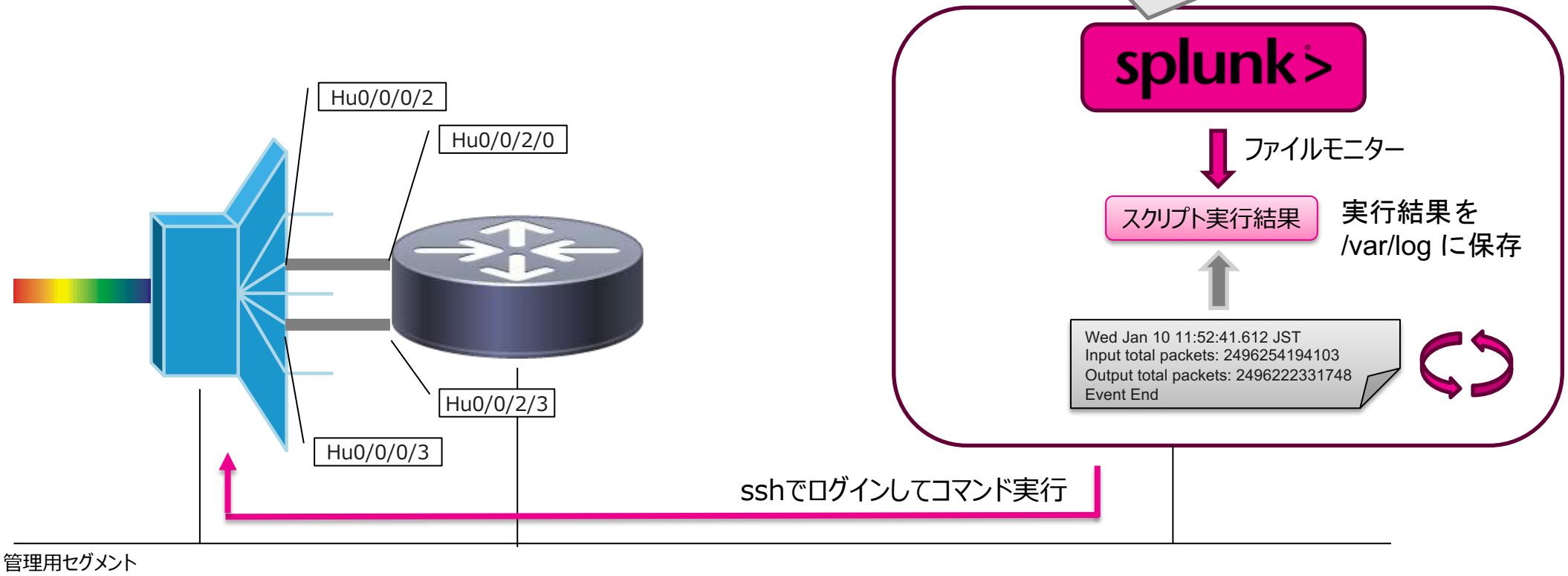
伝送装置(WDM)のDOWN関連ログをポートダウンのイベントとして追加してみた

R4 - Jan 5 15:59:21.887 JST: ifmgr[181]: %PKT_INFRA-LINK-3-UPDOWN : Interface HundredGigE0/0/2/3, changed state to Down
R4 - Jan 5 15:59:21.887 JST: ifmgr[181]: %PKT_INFRA-LINEPROTO-5-UPDOWN : Line protocol on Interface HundredGigE0/0/2/3, changed state to Down
R22 - Jan 5 15:59:26.772 JST: osa_driver[182]: %PKT_INFRA-FM-6-FAULT_INFO : OPUK-CSF :DECLARE :ODU40/0/0/0/2:
R21 - Jan 5 15:59:26.812 JST: osa_driver[346]: %PKT_INFRA-FM-3-FAULT_MAJOR : ALARM_MAJOR :SIGLOSS :DECLARE :HundredGigECtrlr0/0/0/3:

困った時のスクリプト対応

あれ、SNMP MIB / Telemetryでデータが取れない・・・

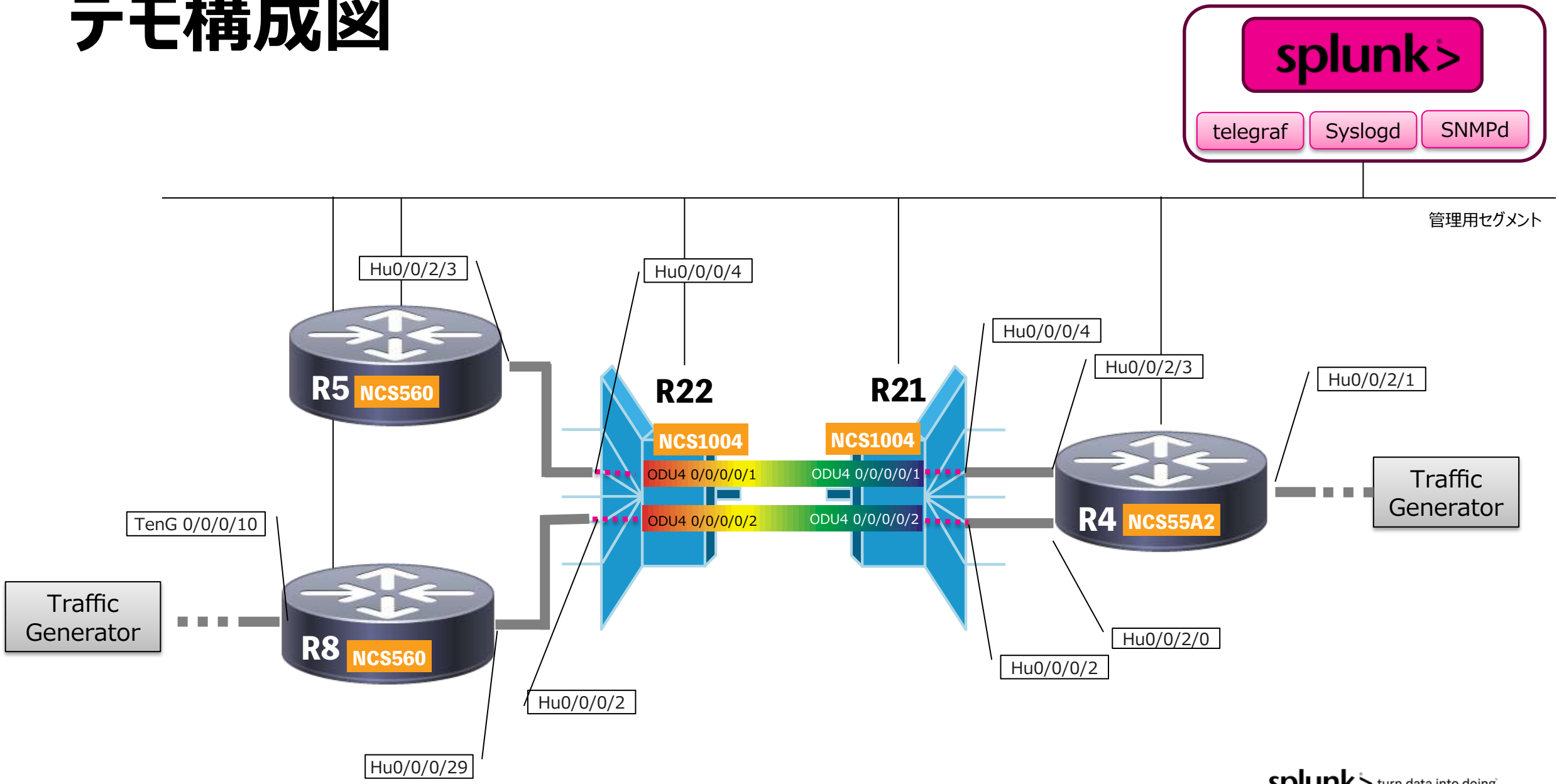
イベント改行：Event End([¥r¥n]+)
タイムスタンプ形式：%a %b %d %H:%M:%S.%3N %Z
例) フィールド抽出
Input total
packets:¥s+(?P<input_packets_counter>¥d+)¥s+Output
total packets:¥s+(?P<output_packet_counter>¥d+)



デモ

1. Telemetryを活用したサイレント障害検知
2. Telemetryを活用したバーストトラフィック検知
3. WDMとルータ間の接続確認

デモ構成図



シナリオ１

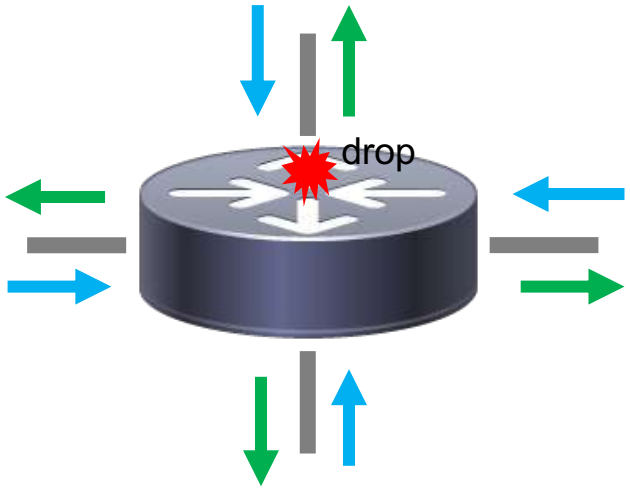
Telemetry dataを使ったサイレント障害検知

こんな時？
- メンテナンス作業における迂回作業後
- 故障機器の交換後
- 設定変更後
- 周辺装置のメンテナンスにようりMax-metricを受信した時

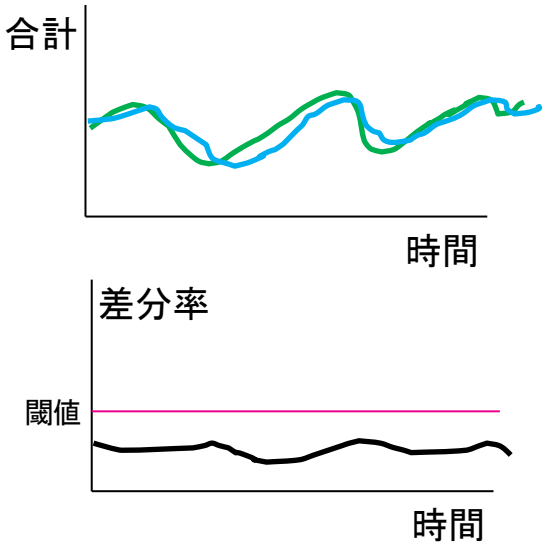
各装置（今回の構成ではR8）のインプットとアウトプットの転送データ量の差分をモニター

通常はxx%程度の差分率

この差分率がxx%以上であれば異常と判断しアラートを通知（もしくは、動的閾値として標準偏差からの外れ値や機械学習によるベースラインからの外れ値として検知）

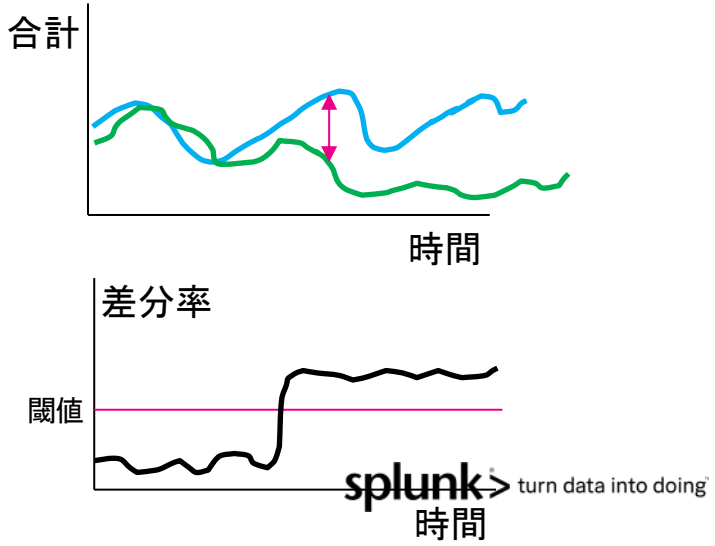


通常時



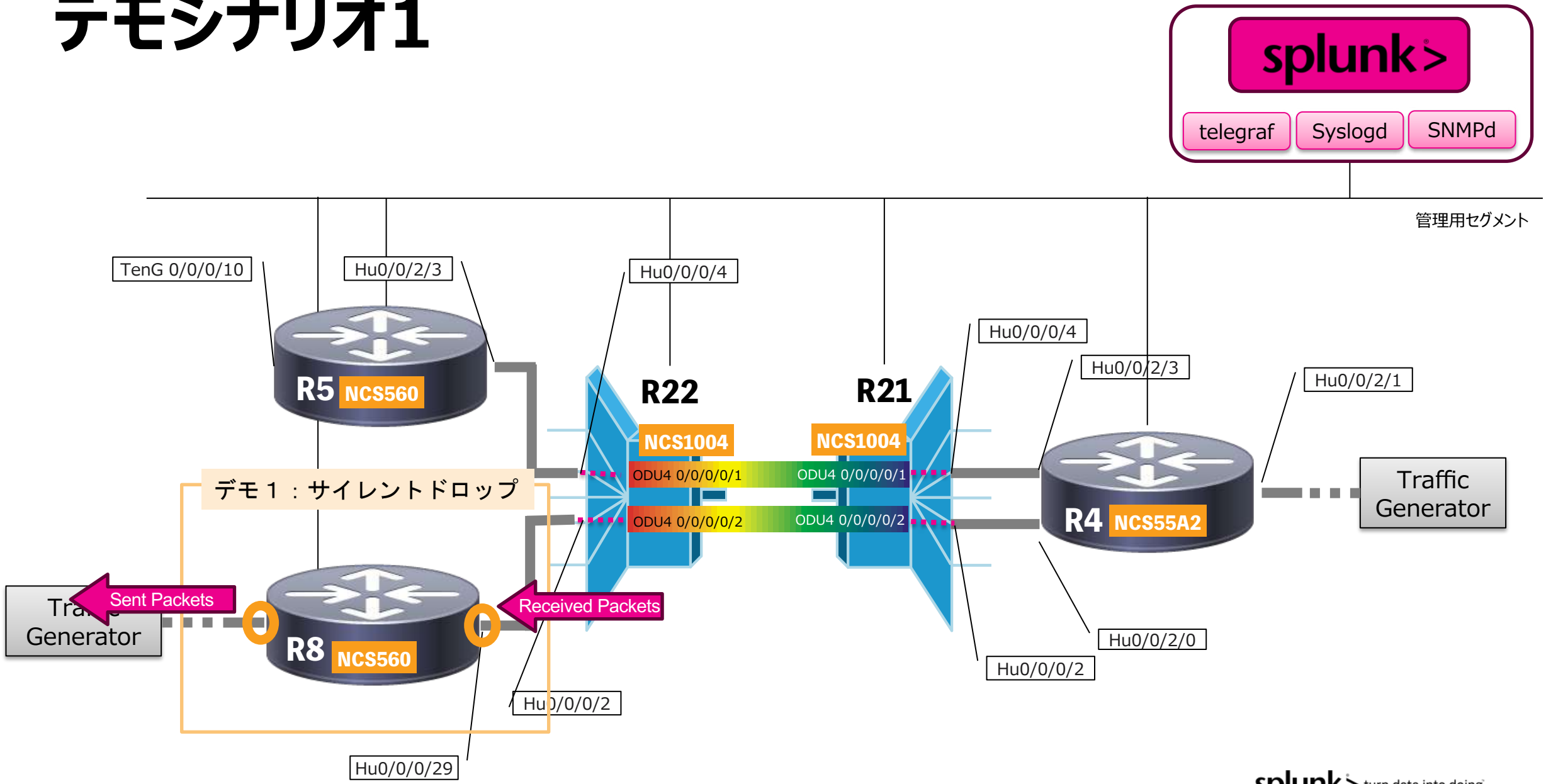
サイレント障害の疑似発生
アウトプットデータをアクセスリストでドロップさせることでインプットとアウトプットのデータ量に差分が発生する

異常時



splunk> turn data into doing

デモシナリオ1



splunk>enterprise App ▾ Administrator ▾ メッセージ ▾ 設定 ▾ アクティビティ ▾ ヘルプ ▾ 検索

サーチ Analytics データセット レポート アラート ダッシュボード App

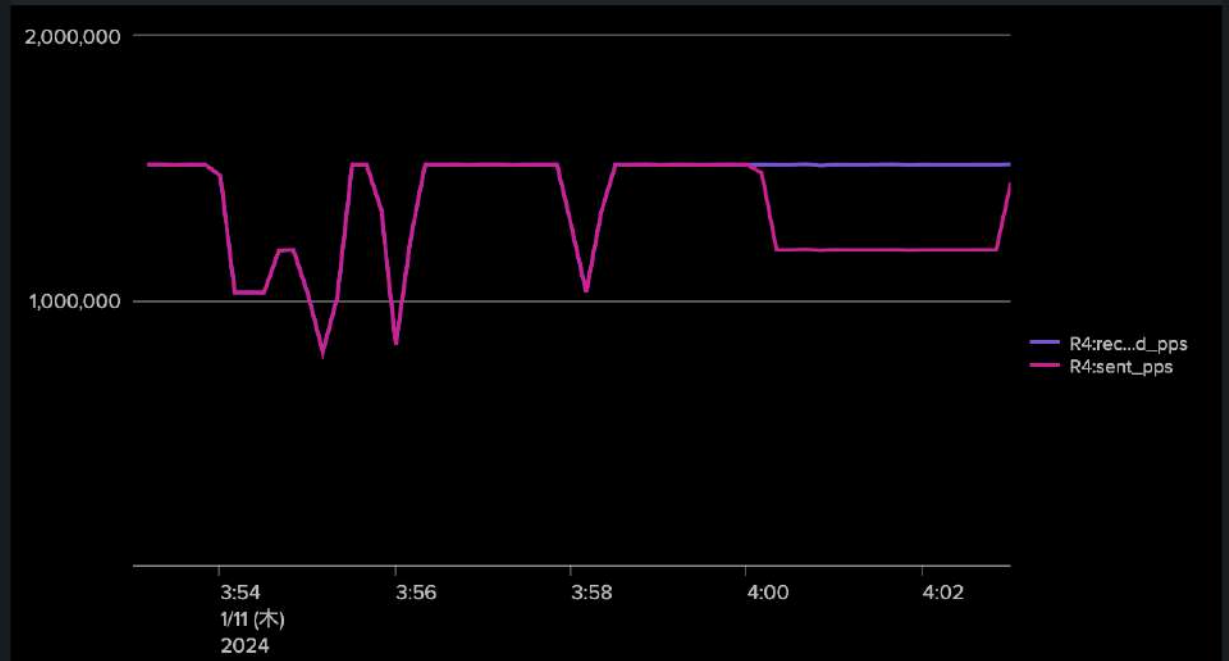
Telemetry dataを使ったサイレント障害検知

編集 エクスポート ▾ ...

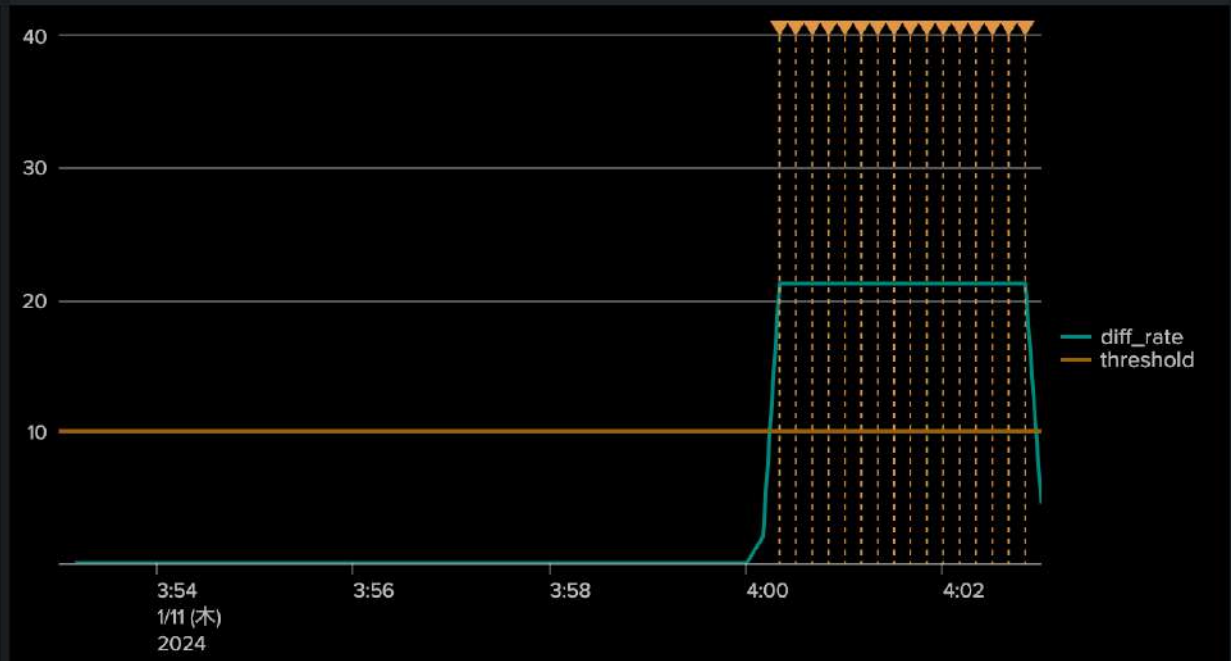
ホスト

R8-NCS540 ▾ フィルターを非表示

転送データ量の合計



差分率 (%)

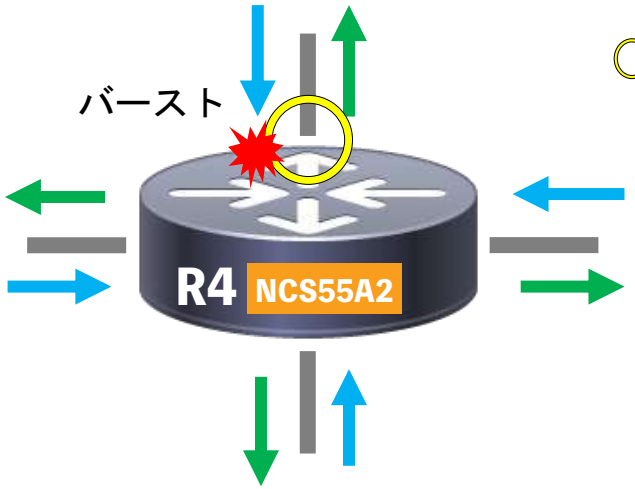


splunk>

シナリオ２

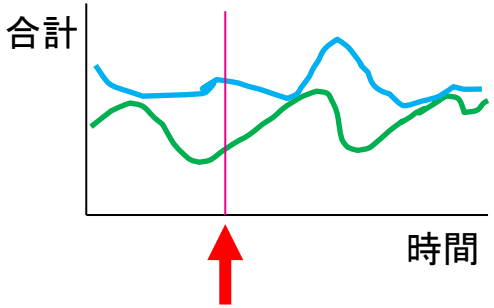
Telemetry dataを使った瞬間的なバーストトラフィックの検知

各装置（今回の構成ではR8）のHu0/0/2/0の転送データ量をTelemetryでモニター
最初は集計間隔をSNMPと同じ5分(300sec)で設定　（もしくはSNMP MIBをそのまま使う）
アクセスリストで特定フローの通信で1-2秒間バーストを発生させる
SNMP(5分平均の集計)だと１－2秒のバーストでは通信量の変化としてとらるのが難しい
Telemetryによる1秒(～30秒)のデータ集計だと、バーストを視覚的に捉えることができる



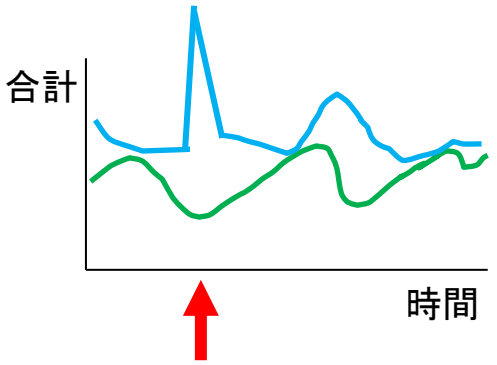
SNMP

監視対象のインターフェイスの通信量



瞬間的なDropの擬似発生
インプットデータをアクセスリストで1-2秒間ドロップさせる
SNMP MIBだとDropを視覚的に把握し辛いがTelemetryだと瞬間的なDropも視覚的に確認できる

Telemetry



splunk> turn data into doing

splunk>enterprise App ▾ Administrator ▾ メッセージ ▾ 設定 ▾ アクティビティ ▾ ヘルプ ▾ 検索

サーチ Analytics データセット レポート アラート ダッシュボード App

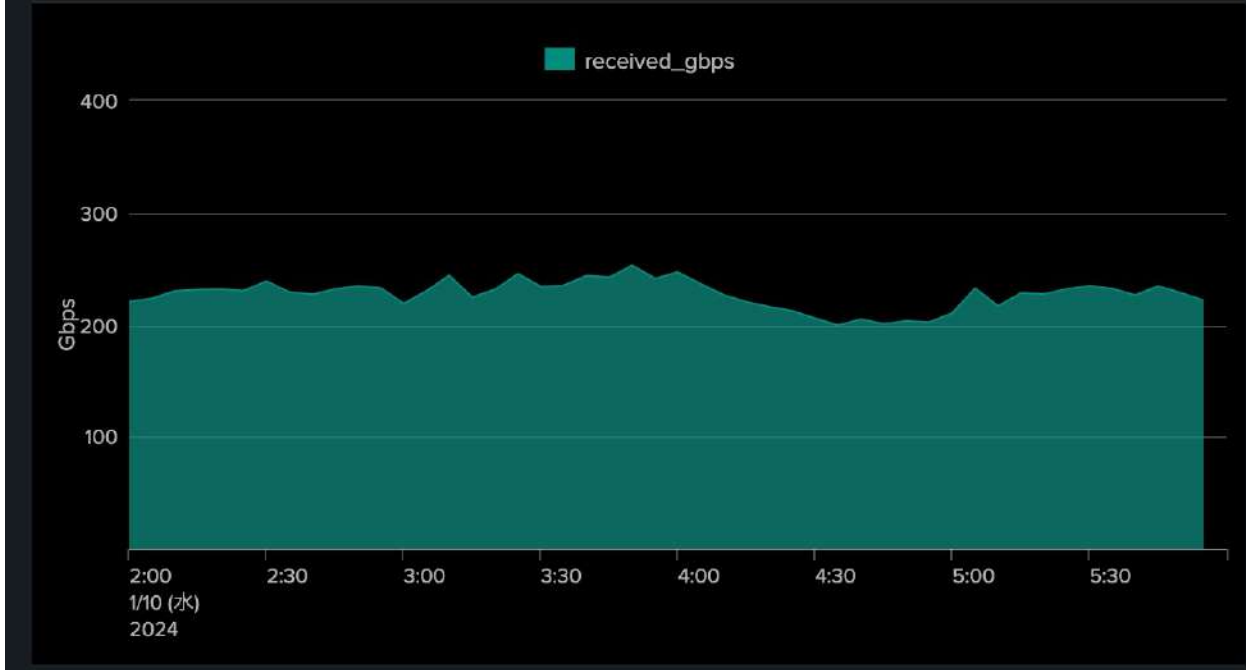
Telemetry dataを使った瞬間的なバーストトラフィックの検知

編集 エクスポート ▾ ...

02:00 – 06:00, 2024年1月1... ▾ フィルターを非表示

SNMP（5分平均の集計）

Received traffic (Gbps)

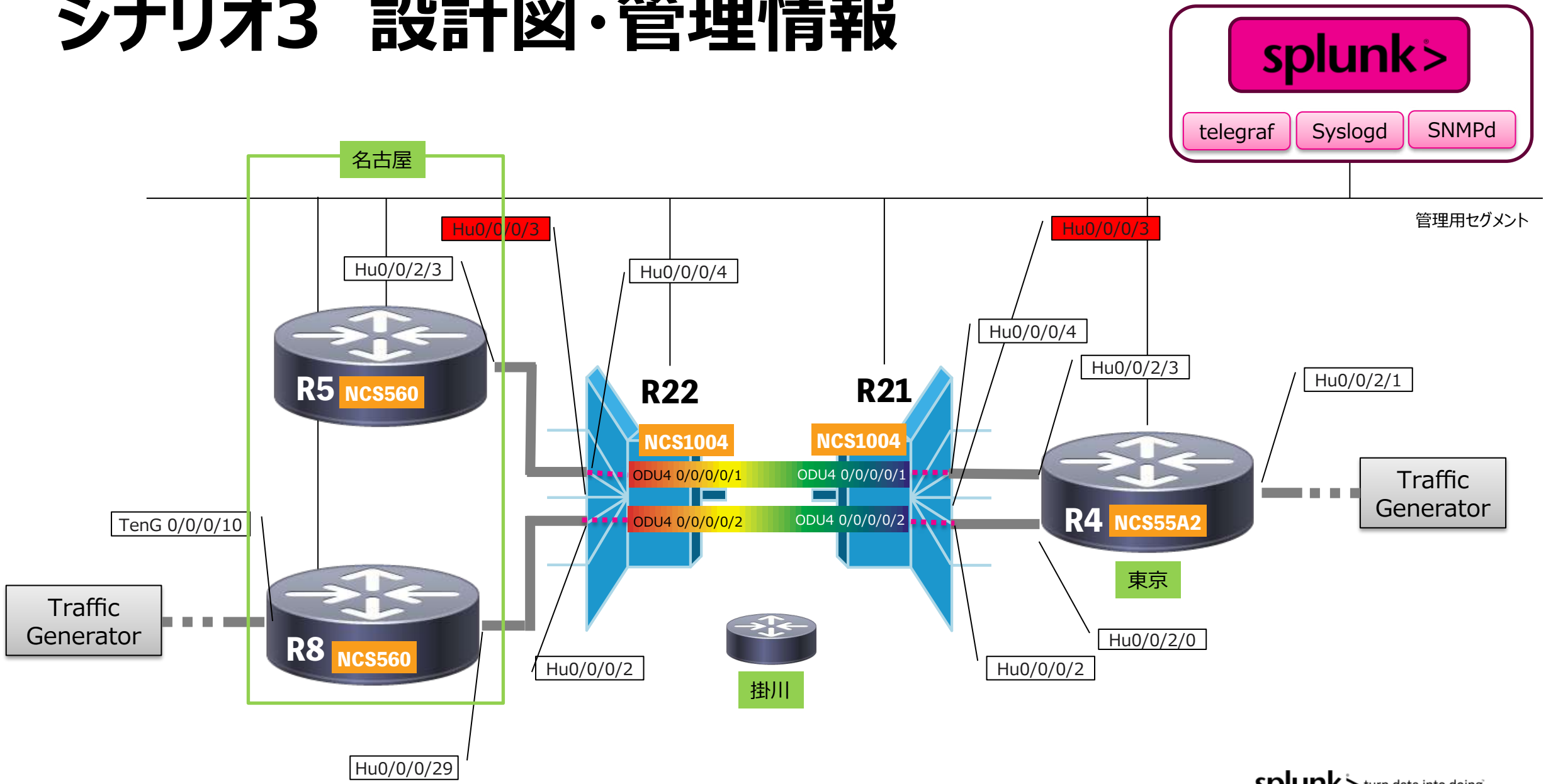


Telemetry（30秒平均の集計）

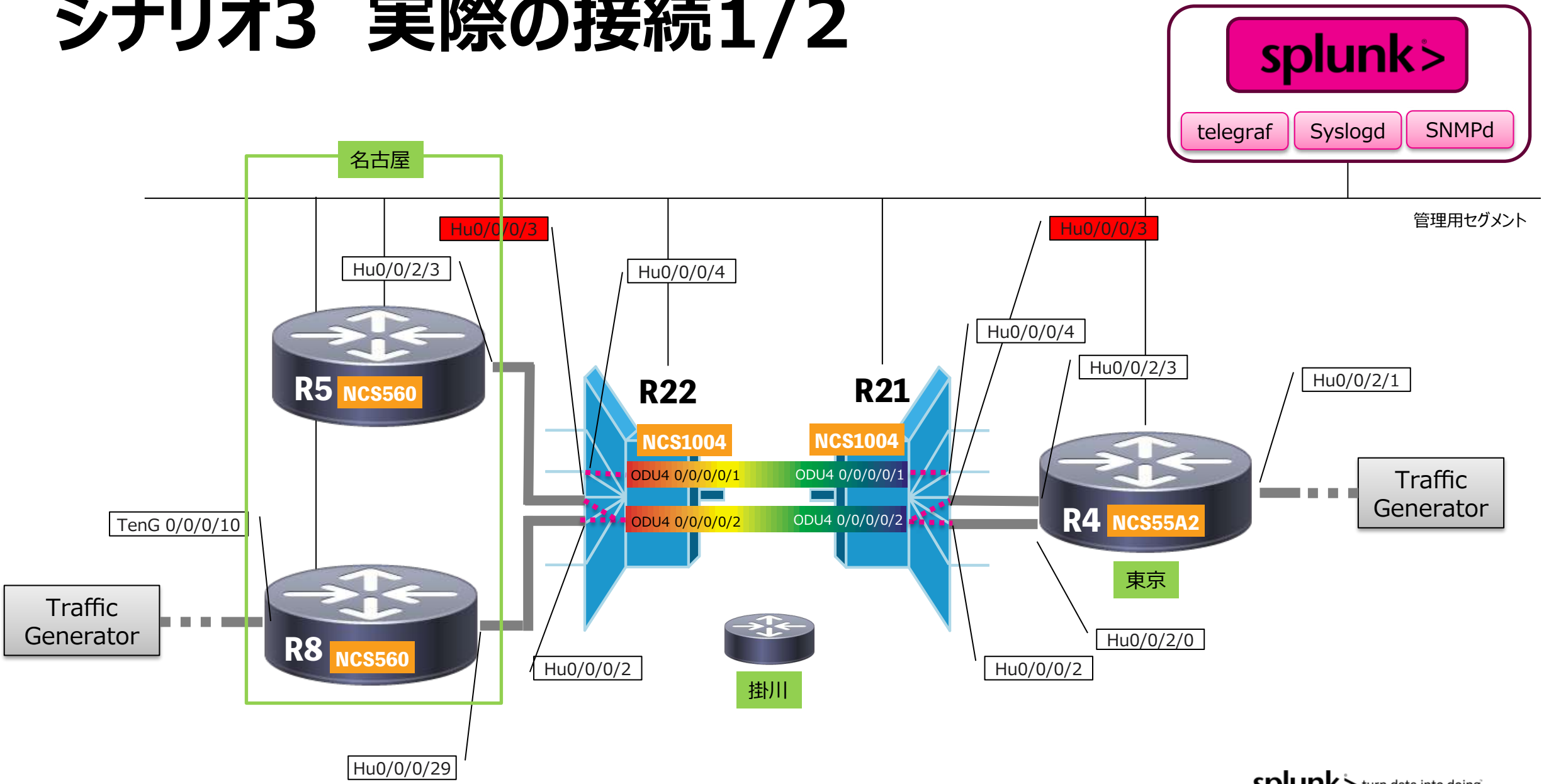
Received traffic (Gbps)

splunk>

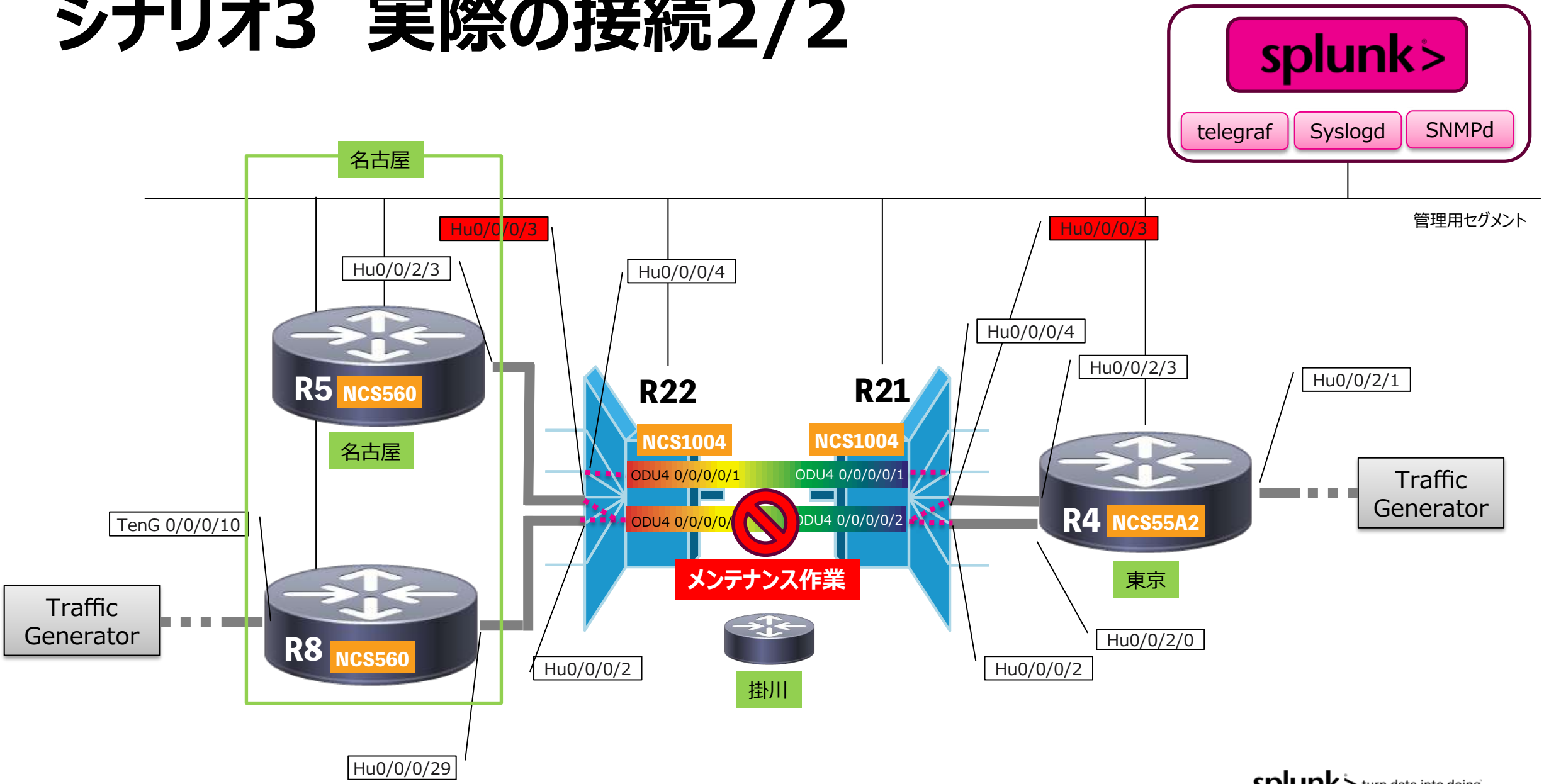
シナリオ3　設計図・管理情報



シナリオ3　実際の接続1/2



シナリオ3　実際の接続2/2



splunk>enterprise App ▾ Administrator ▾ メッセージ ▾ 設定 ▾ アクティビティ ▾ ヘルプ ▾ 検索

サーチ Analytics データセット レポート アラート ダッシュボード App J53 Demo

サーチ

ルータとWDM間の接続情報の確認

Actions ▾ 編集

Time Range: 12:00 – 13:00, 2024年1月1...

Router1: R8

Router1 Interface: HundredGigE0/0/0/29

Router1 metric: received_pps

Router2: R22

Router2 Interface: HundredGigE0/0/0/2

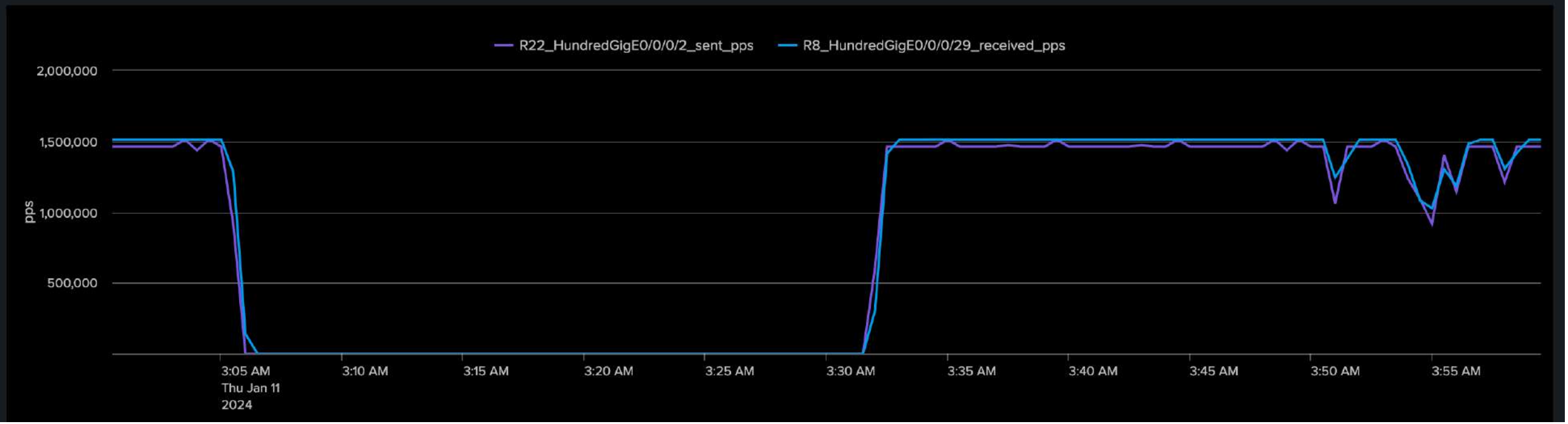
Router2 metric: sent_pps

接続確認

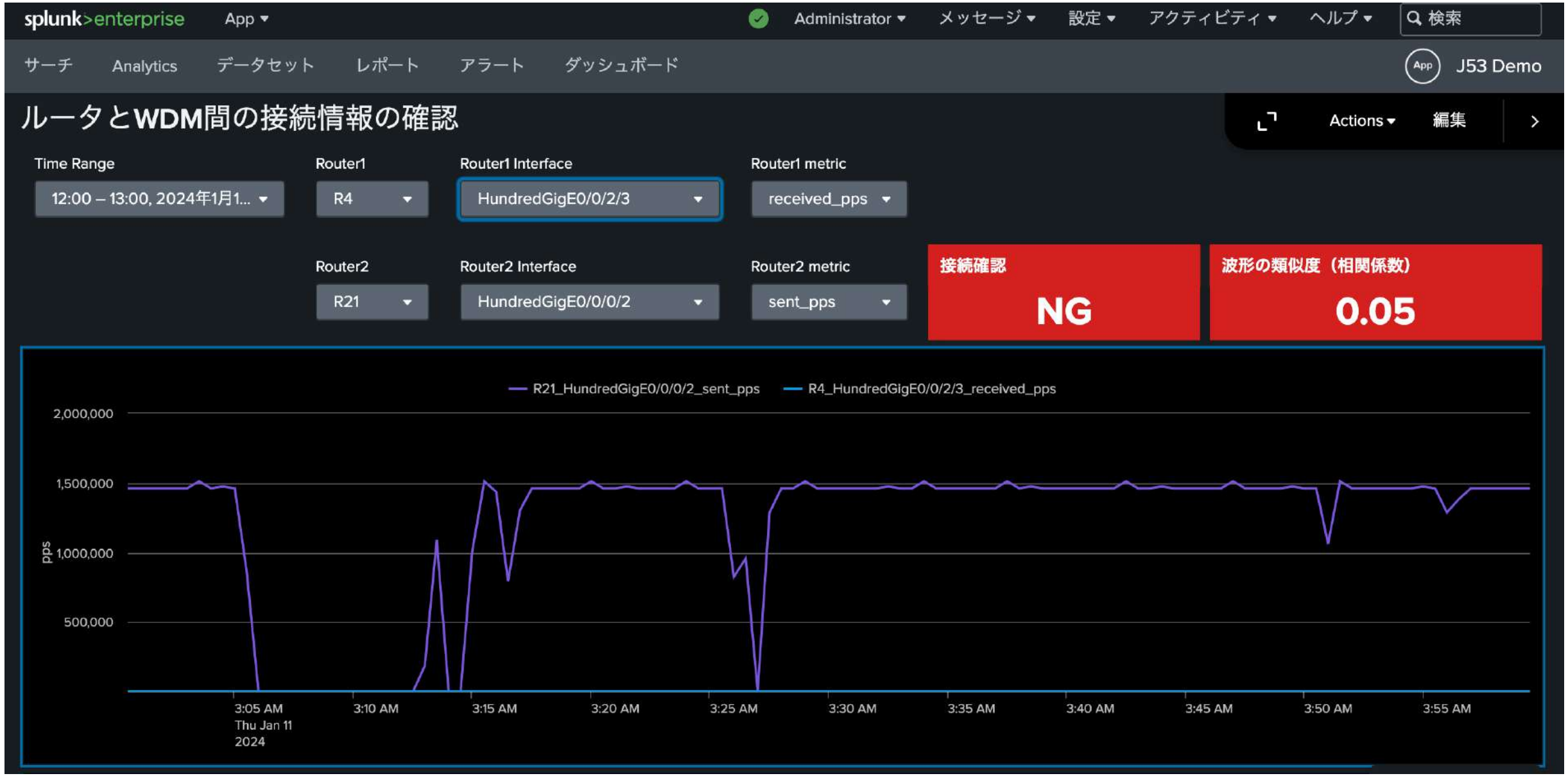
OK

波形の類似度（相関係数）

1.00

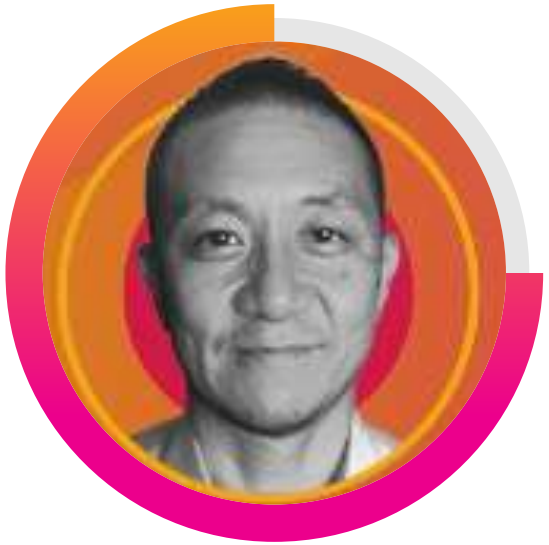


splunk>



最後に

Special Thanks



村田 達宣

Splunk プリンシパルアーキテクト

近藤 洋平

Splunk セールスエンジニア

議論したいこと

TelemetryはStandardになっているのか?

- なりきれていないのだとしたら理由を議論してみたい
- HWアーキテクチャの状態のTelemetryの情報も取得できるけど、ここまで見る？

Topologyの可視化についても技術進化はしている

- L3 Topologyの可視化は動的に行っていますか?
- 他のLayerとの連携はどうしていますか? (特にL1は気になります)
- 絵を起こした後、データとして扱うことはできていますか?

ProbeをだしてのNW状態監視したデータを上記と組み合わせて活用できていますか?

- 色々情報取れるけど、どこまで見る必要がある？

コストと品質のバランス

- どこまで追求してやりますか？

Thank You!

splunk> turn data into doing