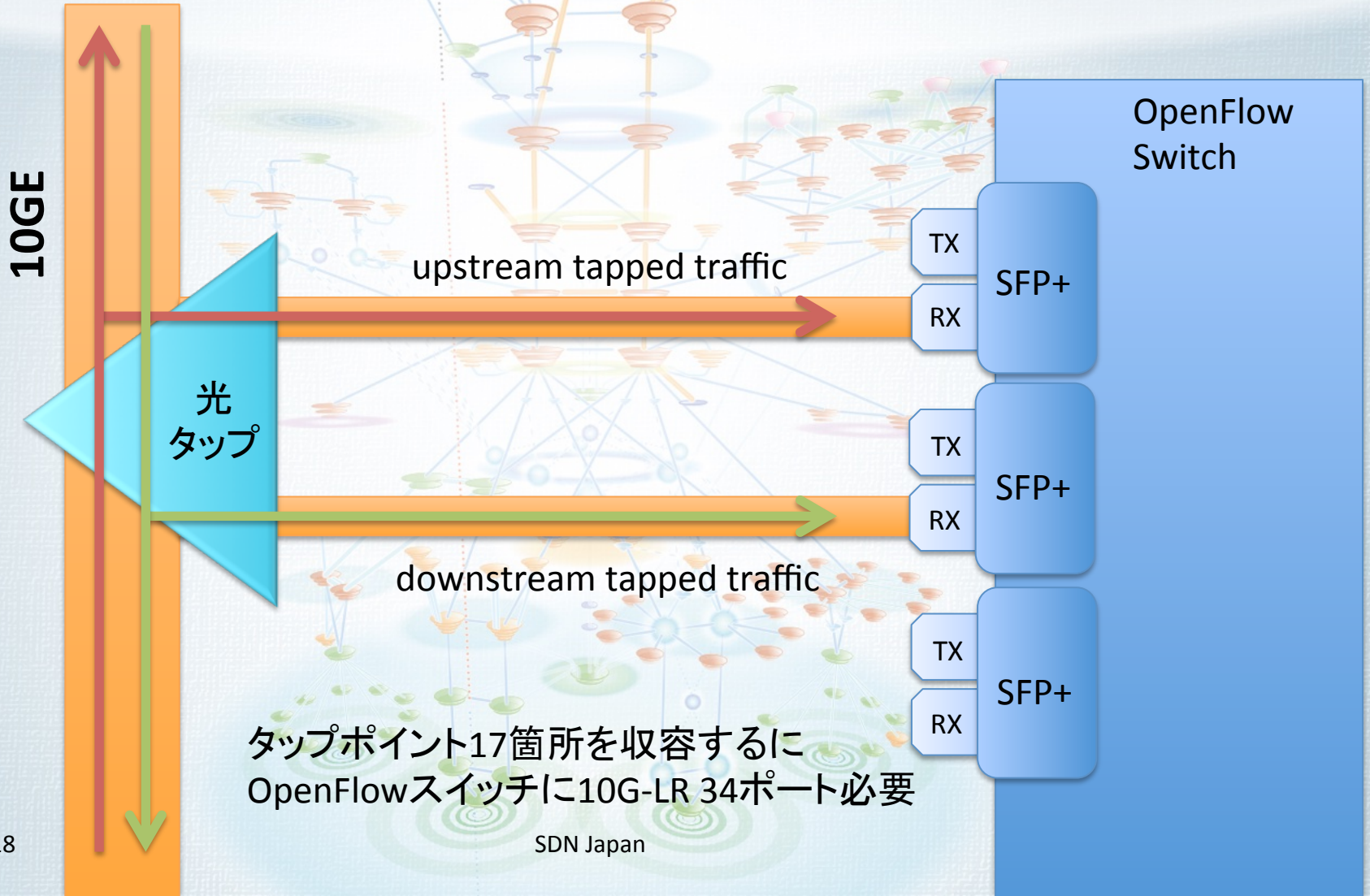
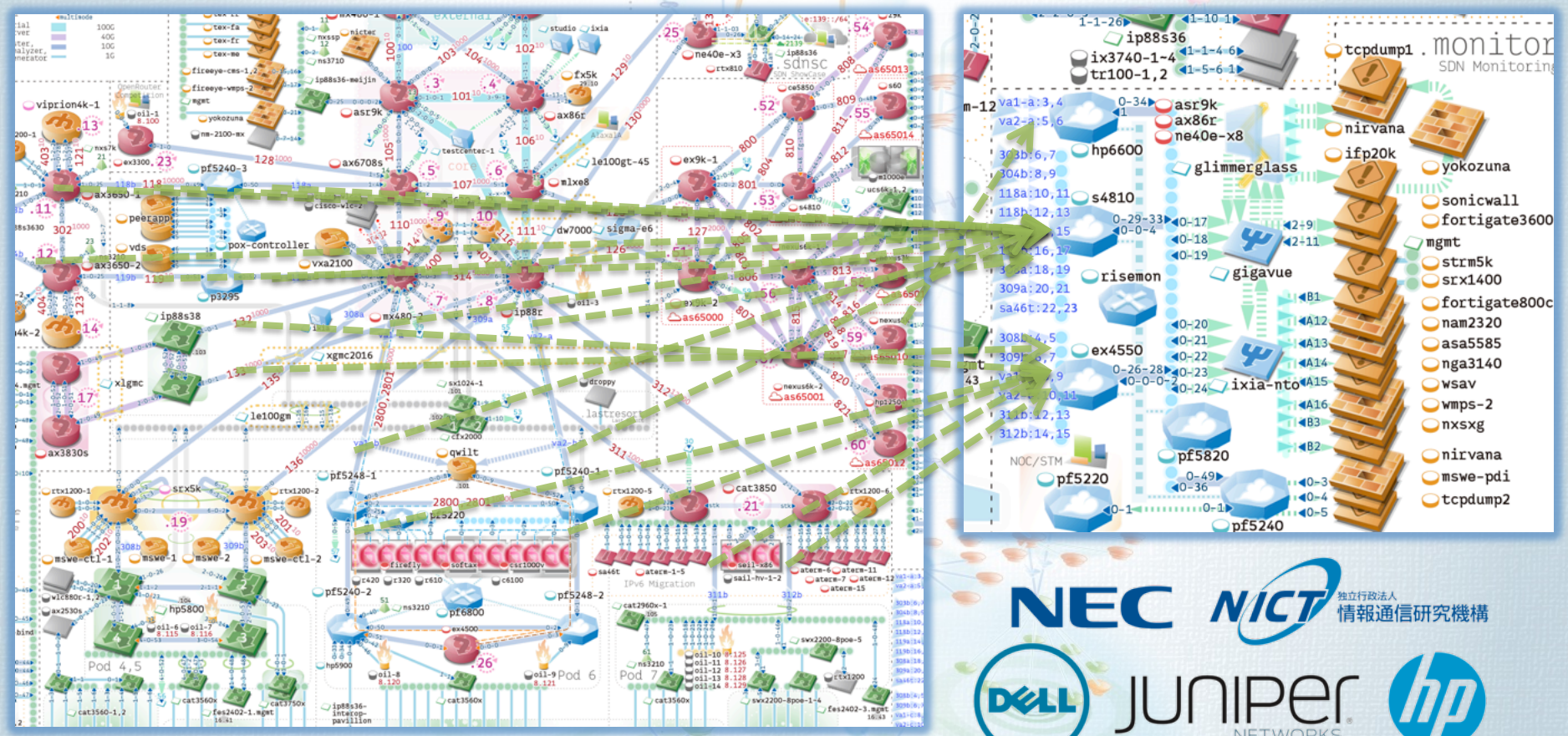
タップしたトラフィックをOpenFlowスイッチへ



タップポイント17箇所を収容するに
OpenFlowスイッチに10G-LR 34ポート必要

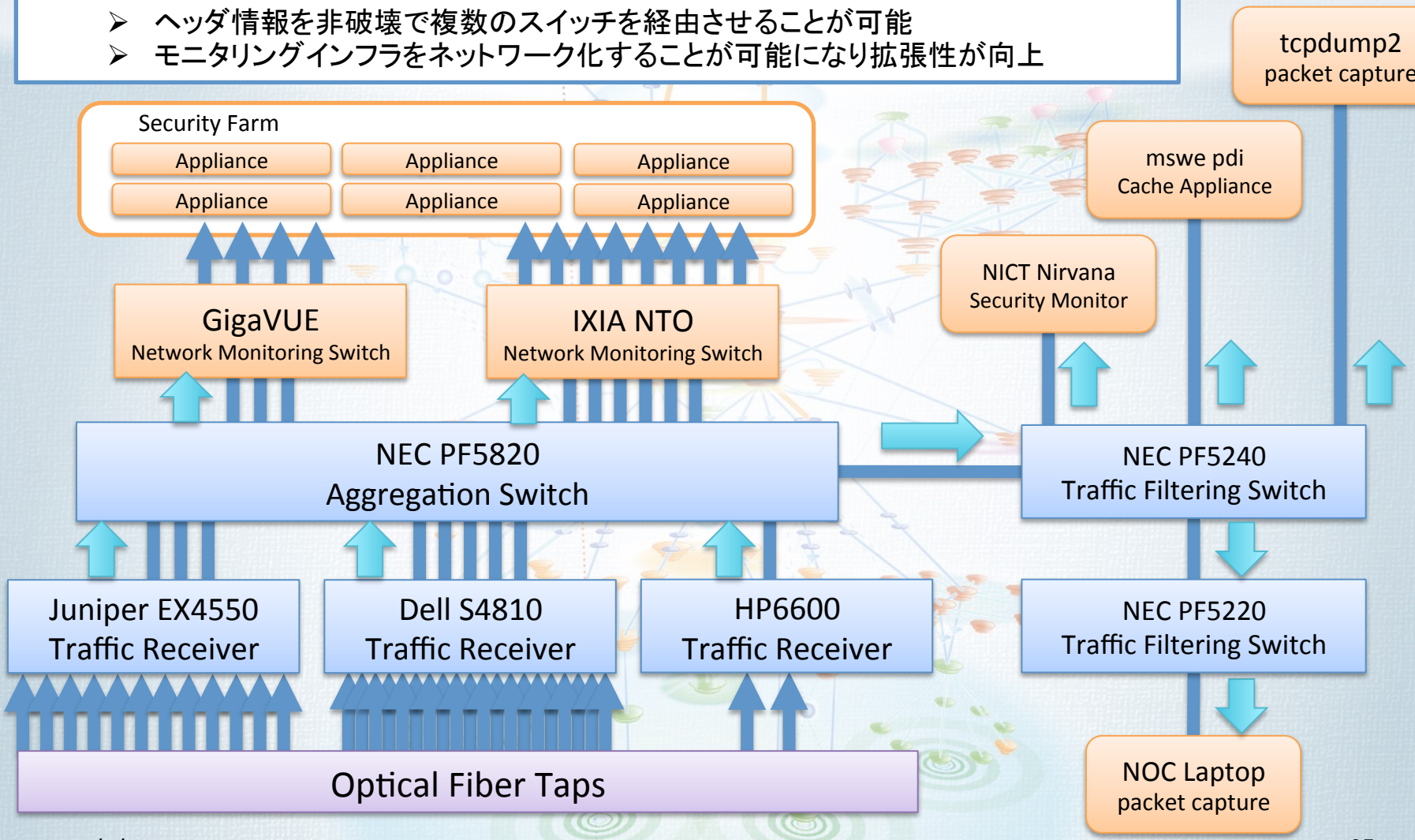
タップからの配線を収容してる様子

ShowNetの10Gbpsリンク17箇所に割り入れた光タップからのキャプチャーパケットを
OpenFlowスイッチ6台のネットワークでコントロールし解析装置などへ供給



NEC NICT 独立行政法人 情報通信研究機構
DELL JUNIPER NETWORKS hp

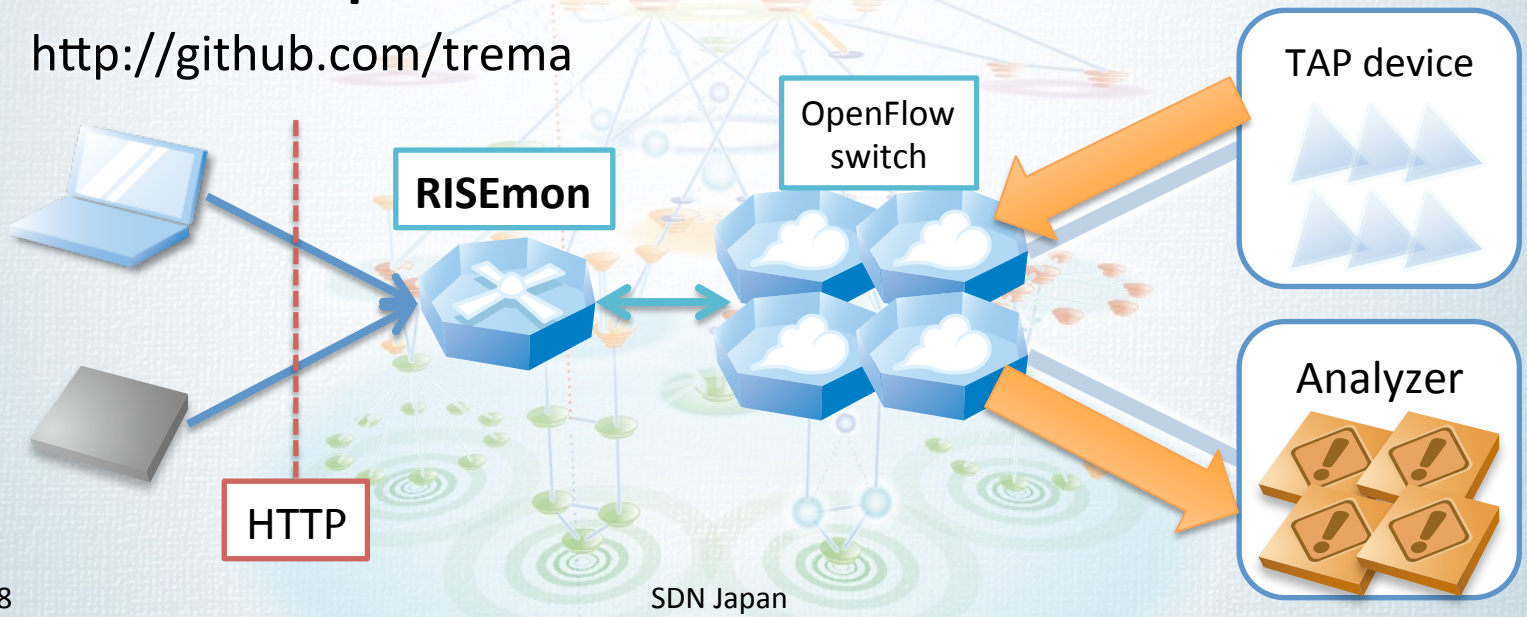
- 従来はポートミラーリングやL1スイッチで行なっていたものをOpenFlowで実現
  - 集中制御と外部との動的連携が可能になった
  - ヘッダ情報を非破壊で複数のスイッチを経由させることが可能
  - モニタリングインフラをネットワーク化することが可能になり拡張性が向上



デバッグ中の風景

RISEmon

- モニタリングインフラの集中制御コントローラ
  - Web UIによるオペーレータ制御
  - REST APIによる外部アプリケーションとの連携
- **Trema based OpenFlow controller**
  - http://github.com/trema



モニタリングしたい
トラフィックフローを
Match条件で指定

RISE mon Web UI

Back

macsrc:
macdst:
mactype:
ipsrc:
ipdst:
ipproto:
tpsrc:
tpdst:
out_port:
Add Monitor Flow

Monitor flow(s)

Match	Priority	Output Port	Updated at	Stats
In Port 49 VLAN * MAC src * MAC dst * MAC Type 0x800 IP src 130.129.254.204/32 IP dst 0.0.0.0/0 IP proto * TP src * TP dst *	50000	3 delete 3 append: Append Port	Fri Jun 14 15:15:45 +0900 2013	Stats at Fri Jun 14 16:29:44 +0900 2013 duration_sec 4439 duration_nsec 0 packet_count 0 byte_count 0
In Port 49 VLAN * MAC src * MAC dst * MAC Type 0x800 IP src 130.129.187.14/32 IP dst 0.0.0.0/0 IP proto * TP src * TP dst *	50000	3 delete 3 append: Append Port	Fri Jun 14 08:28:41 +0900 2013	Stats at Fri Jun 14 16:29:44 +0900 2013 duration_sec 28863 duration_nsec 0 packet_count 2856628 byte_count 393557809
In Port 49 VLAN * MAC src * MAC dst * MAC Type 0x800 IP src 10.255.8.105/32 IP dst 0.0.0.0/0 IP proto * TP src * TP dst *	50000	3 delete 3 append: Append Port	Fri Jun 14 13:22:50 +0900 2013	Stats at Fri Jun 14 16:29:44 +0900 2013 duration_sec 11214 duration_nsec 0 packet_count 0 byte_count 0

こんな風に使えました

冗長回線のトラフィックを10G 1ポートで解析装置に欲しい

- 複数回線でPrimacy/Backupしてるとルーティングの切り替わりで流れる回線が変わるが、解析側ではその影響を考慮したくない
- 複数ポートからのinputを選択して単一ポートへ出力
- 昨年まではポートミラーリングを使っていたが集中制御で圧倒的に運用が楽に

NW全体から特定アプリケーションに関わるトラフィックだけ欲しい

- メールセキュリティ製品やWebキャッシュ製品に多いパターン
- 特定TCPポート番号のフローを抽出して出口ポートに出力
- 同じことをOpenFlow無しでやるならPolicy Based Routingが必要

そのほか細かい条件など来ても条件の組み合わせで実現可能

やってみて解ったこと

- **動的な集中制御はやっぱり楽だった**
  - 既存手法は静的／手動なものが多い
    - 動的な仕組みが無いところにSDNを持ち込むのは有効
- **やっぱり私達はネットワーク技術者だ**
  - 解析の専門家はどこをどう取ればいいか分からない
  - モニタリング網構築にこれまでの知見が活かせる
- **事前に作ることが重要だ**
  - モニタリングの要望が来てからNWを止めたくない
  - 後付だとどうしても時間がかかる

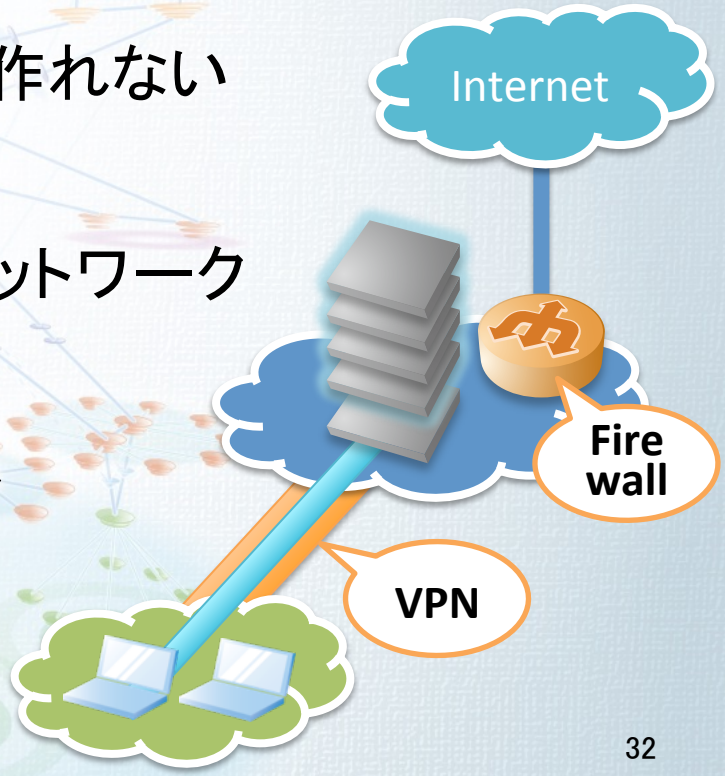
show net
> Go to the next decade

INTEROP
TOKYO | 12-14 JUNE, 2013

クラウドアプライアンスネットワーク

クラウドのネットワーク

- **サービスは、VMだけではつくれない**
  - VMは仮想化された**“サーバ資源”**
  - サーバ資源だけではサービスは作れない
- **仮想ネットワーク**
  - IaaSクラウドにおけるユーザのネットワーク
  - **“ネットワークの機能”**の必要性
  - ex.) VPNによるクラウドへの接続、
    Firewall、NAT



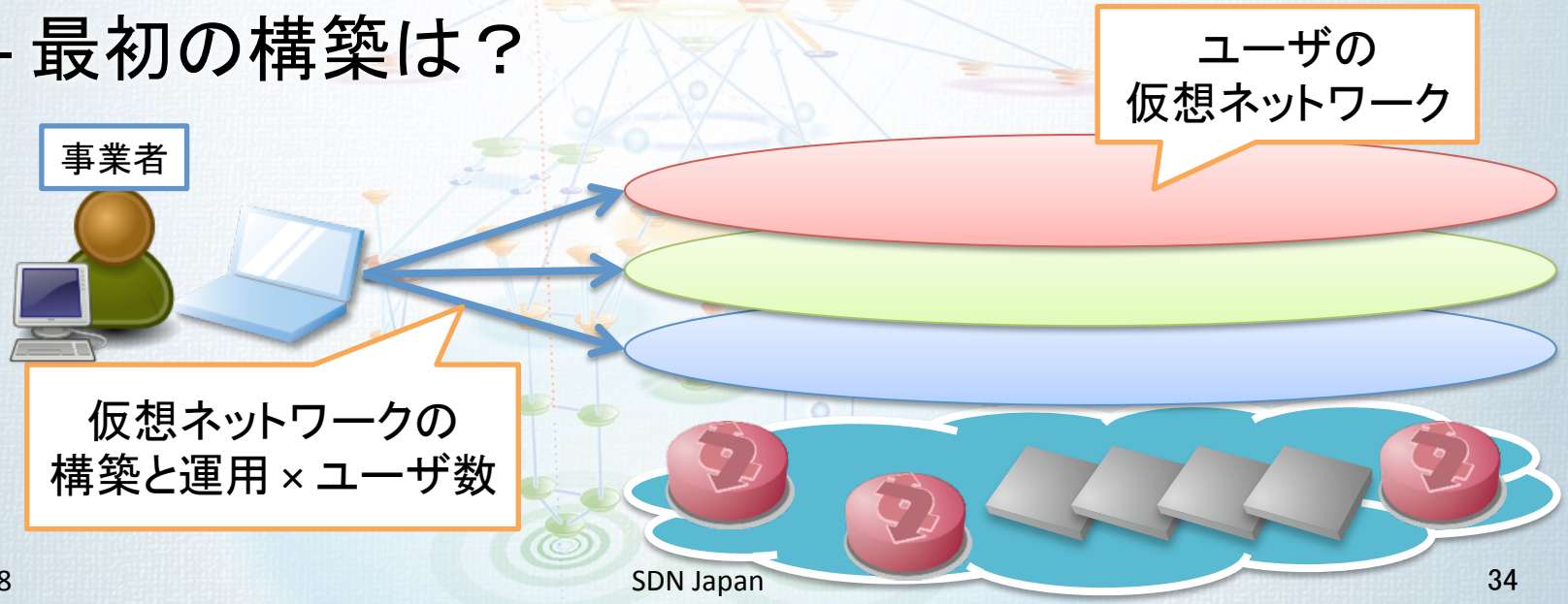
でも、欲しい機能は様々

- 必要な機能はユーザによって色々
  - VPN、L2/L3？P2P/マルチポイント？暗号化は？encap formatは？
  - NAT、VIP、Protocol Translation、HTTP Proxy、Socks
  - Firewall、Policy based filter、Screening、IDS

これらの機能を全て
各IaaSクラウドが実装するのは非現実的

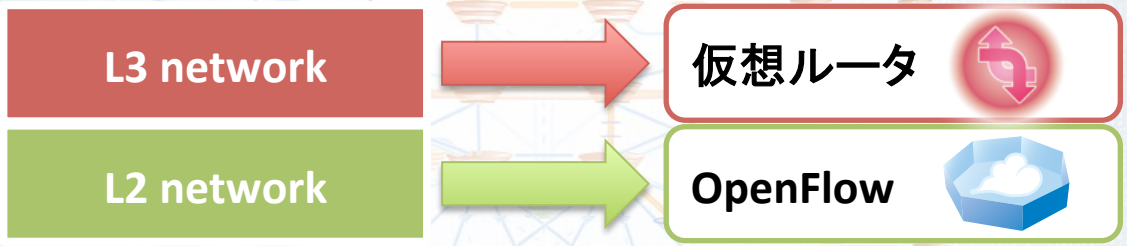
さらに、構築は自動化したい

- 誰が構築し、運用するのか
  - ユーザごとに要求の異なる仮想ネットワークをIaaSクラウド事業者が運用するのは無理
  - 最初の構築は？



そこでSDNですよ！！

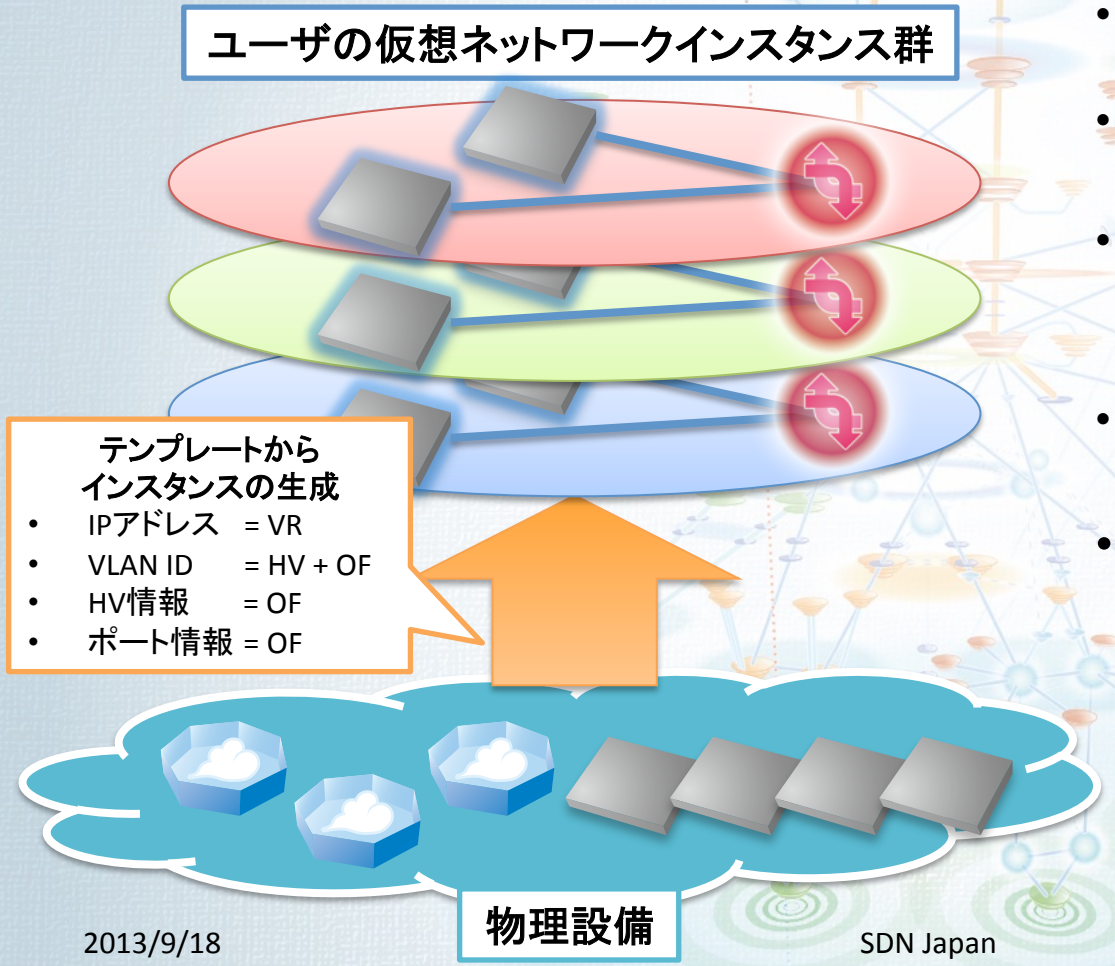
- ネットワークの機能をソフトウェアで定義



- テンプレートによる動的な生成
  - ユーザのネットワークをテンプレートから生成
  - 仮想ルータの設置、OpenFlowによるL2の設定

動的なプロビジョニングを実現

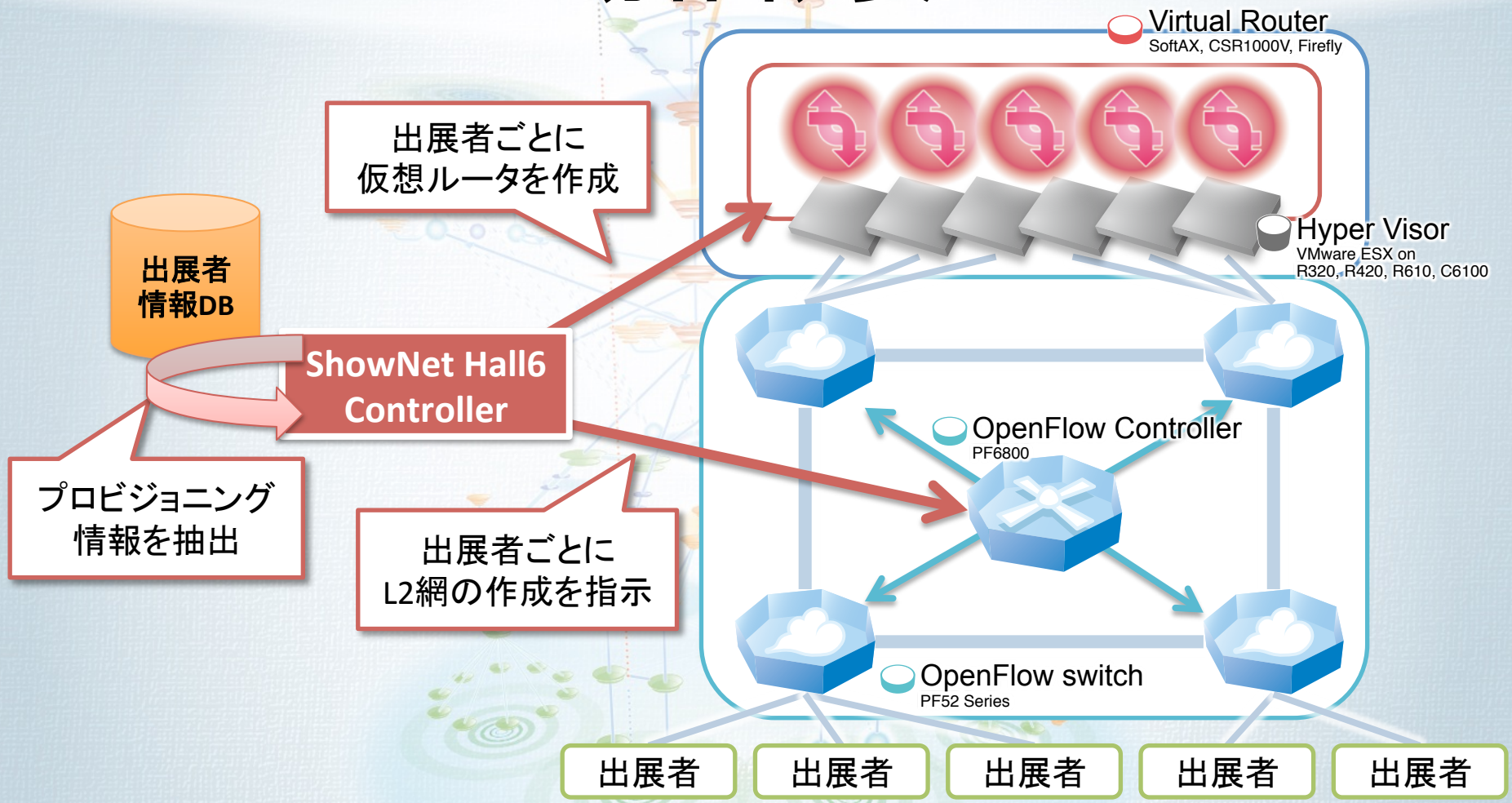
SoftwareでNetworkをDefineしよう！



- 各変数は、それぞれのレイヤの機能で定義される
- Layer 1 (物理ポート情報)
  - OpenFlow
- Layer 2
  - OpenFlow
  - HV内のソフトウェアスイッチ
- Layer 3
  - 仮想ルータ
- 全ての機能は既にソフトウェアで制御可能

仮想ネットワークのインスタンス化をソフトウェアで全自動で行う

動作概要


Virtual Router
SoftAX, CSR1000V, Firefly
Hyper Visor
VMware ESX on
R320, R420, R610, C6100
出展者ごとに
仮想ルータを作成
出展者
情報DB
ShowNet Hall6
Controller
プロビジョニング
情報を抽出
出展者ごとに
L2網の作成を指示
OpenFlow Controller
PF6800
OpenFlow switch
PF52 Series
出展者
出展者
出展者
出展者
出展者

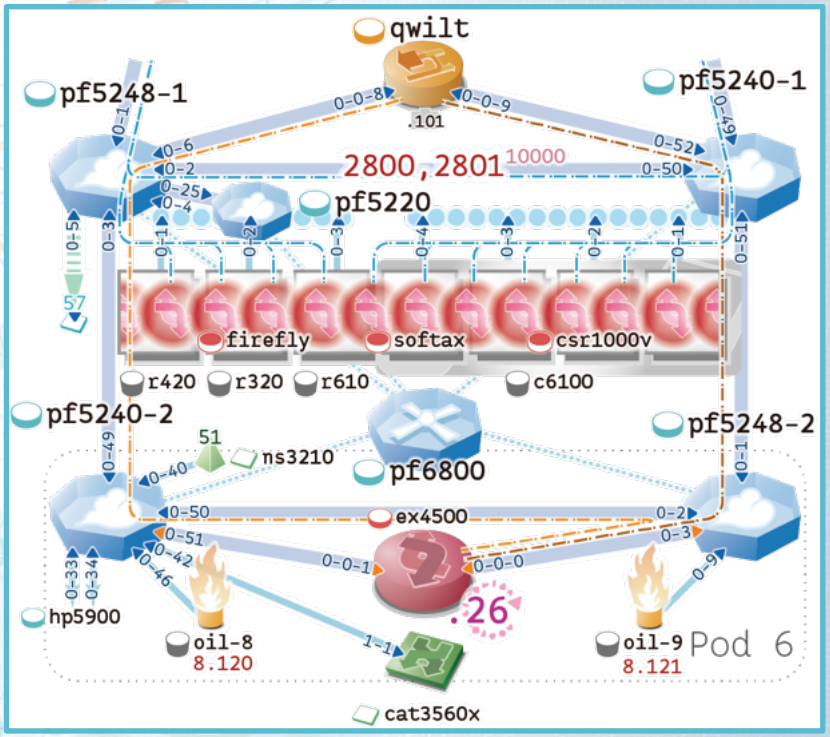
ShowNetでの構成

- **実際に出展者ネットワークを構築**
  - 6ホールにおける出展者を収容
  - 1出展者毎に1仮想ネットワークを構築
  - 各機能について、コントリビューションされた技術を使用

利用した機器及びソフトウェア

種類	機器名	企業
仮想ルータ	SoftAX	A10 Networks
仮想ルータ	CSR1000V	Cisco Systems
仮想ルータ	JUNOS V Firefly	Juniper Networks
OpenFlowスイッチ	PF52シリーズ	NEC
OpenFlow コントローラ	PF6800	NEC
HVソフトウェア	VMware ESX	VMware

トポロジー図



テンプレートの例

Jinja2によるCSR1000Vのテンプレートファイル

```
!
interface GigabitEthernet1
 description mgmt
 vrf forwarding mgmt
 ip address {{ mgmt_addr }} 255.255.0.0
 negotiation auto
 ipv6 address fd00::2800:{{ vlan - 2000}}/64
 ipv6 enable
 ipv6 nd ra suppress all
!
interface GigabitEthernet2
 description uplink
 ip address {{ uplink_addr }} 255.255.255.0
 ip ospf authentication message-digest
 ip ospf message-digest-key 11 md5 7 011208554D59091B29
 ip ospf hello-interval 10
 ip ospf 290 area 0
 ip ospf cost {{ ospf_cost }}
 negotiation auto
 ipv6 address {{ uplink_addr_v6_ll }} link-local
 ipv6 address {{ uplink_addr_v6 }}/64
 ipv6 enable
 ipv6 nd ra suppress
 ipv6 ospf 290 area 0
 ipv6 ospf hello-interval 10
 ipv6 ospf cost {{ ospf_cost }}
!
```

- **Python Jinja2を利用**

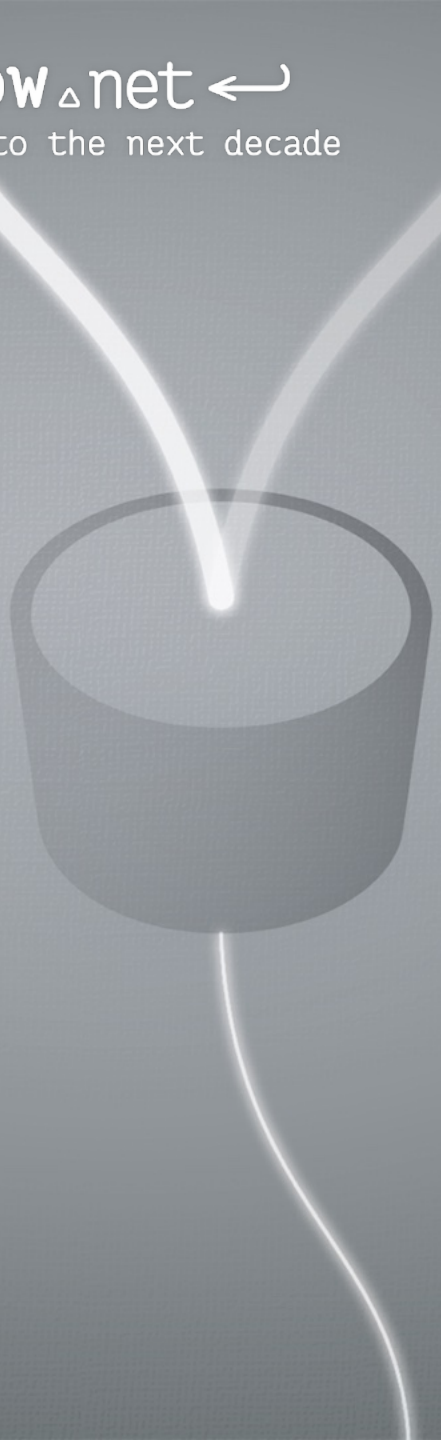
マネージメントアドレス

中で計算もできる。
if文とかforも書けます

アップリンクの
IPv6アドレス

やってみて解ったこと

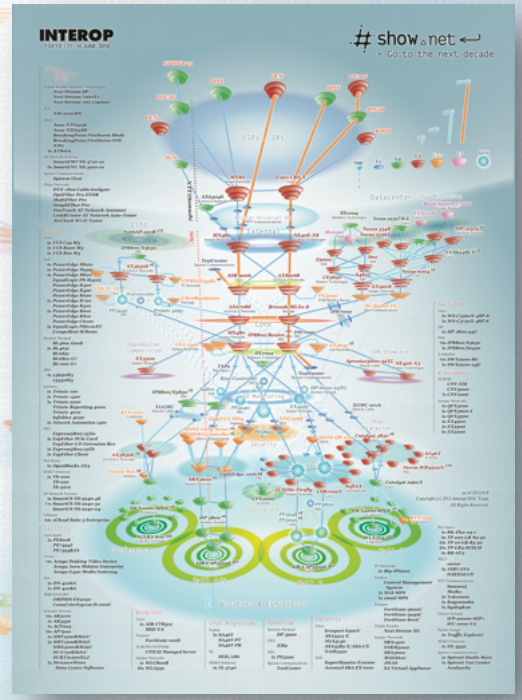
- **SDNによるプロビジョニングは、実現可能**
  - 実現するための各パーツは既に揃っている
  - 下回りの仕込みさえ完成すれば、ユーザネットワークをつくるのはものの数分でできる
  - 一斉に構成変更を行うのも簡単
- **論理構成の多重化と複雑化**
  - ネットワークとサーバと仮想化の組み合わせ
  - それぞれの技術にある程度習熟する必要はある
- **仮想化環境における監視は今後の課題**
  - 論理トポロジの把握
  - 仮想化レイヤーの状態の把握
  - 今年は監視まではいけなかった。来年にご期待ください。



まとめ

ShowNetにおけるSDN

- **様々な形へのチャレンジ**
  - SDNを適用すると幸せになる要求や構成
  - SDNだからできること
- **Live Network**
  - ライブトラフィックの流れるネットワーク
  - 実際に動くSoftware Defined Network
- **相互接続性 : INTEROPerability**
  - 機器間の相互接続性
  - ソフトウェア的な相互接続性
  - そしてフィードバック




show△net
> Go to the next decade