ネットワーク運用自動化のススメ

NTT Communications
先端IPアーキテクチャセンタ

水口　孝則
杉本　周

本日のお話

- 運用/運用システムの現状
- ネットワーク運用の自動化のススメ

  1) ネットワーク管理の自動化

  2) ネットワーク工事（設定）の自動化

  3) ネットワーク運用時（障害時）の自動化

ネットワーク運用の現状

NWはますます高度化・複雑化

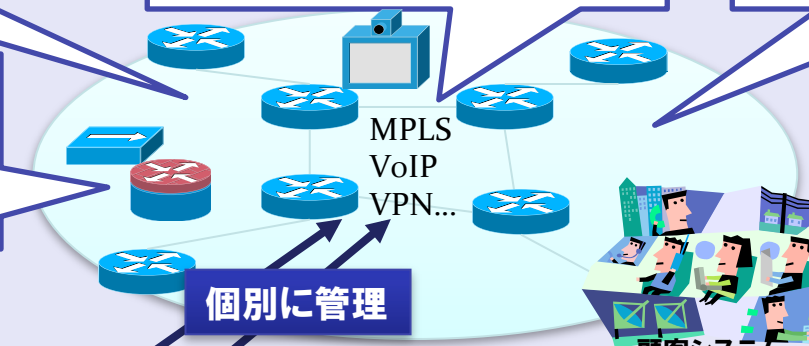
様々なトラブル
回線、機器、ルーティング、DoS攻撃、経路ハイジャック

様々なサービス
Ethernet、Internet接続、IP-VPN、MPLS-VPN、IPv4、IPv6

様々なベンダ
Cisco, Juniper, Alcatel Lucent, Brocade, Alaxala, NEC, …

様々なNW機器
ハイエンドルータ、ミッドレンジ、ローレンジルータとL3スイッチ、MPLSスイッチとIPルータ、マルチサービスルータ

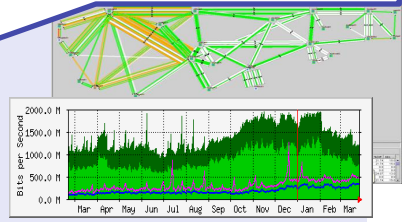
様々な運用システム
NMS、トラフィック監視、ログ管理、顧客システム、チケットシステム



顧客システム


課金システム


個別のスクリプト



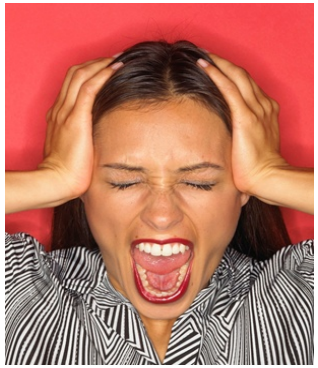
NMS・トラフィック監視

個別に管理

個別に手動設定

個別に設定・操作


オペレータ



【現状】個別システムでの運用、情報連携の不足

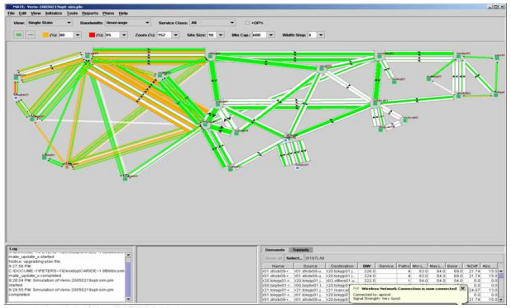
→手作業増加、稼働の圧迫、作業ミス、運用品質低下！

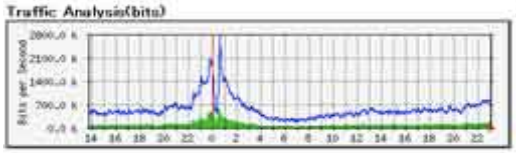
オペレーションの現状（ツール）

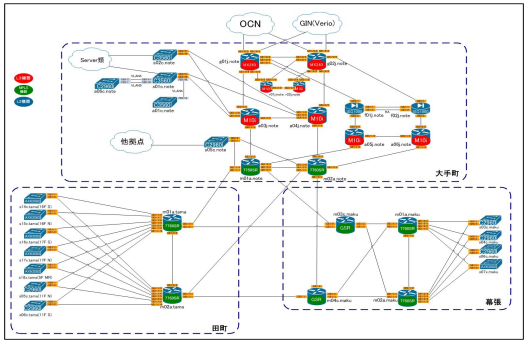
- ネットワーク死活監視
- インターフェーストラフィック監視
- トラフィック解析
- 設備構成管理
- ルータインタフェース収容管理
- IPアドレス管理
- ルーティング管理
- ユーザ情報管理

：

：







ノードリスト	IP	POP	機能	ベンダ
g01a.tky	192.168.0.1	東京	GW	A社
a02c.tky	192.168.0.2	東京	顧客接続	C社
a03b.tky	192.168.0.3	東京	顧客接続	B社
g01c.osk	192.168.2.1	大阪	GW	C社
a02b.osk	192.168.2.2	大阪	顧客接続	B社
p01a.ngo	192.168.8.1	名古屋	ピア接続	A社
g02b.ngo	192.168.8.2	名古屋	GW	B社
g03b.ngo	192.168.8.3	名古屋	GW	B社

運用者の悩み…

- 数多くの機器
- 情報のアンマッチ
- 膨大な作業
- ：
- ：

ルータ増えたら勝手に情報反映してくれないかな…

彼女ほしいな…

給料増やして…

ノード・サーバの情報が常に最新になればいいのに

ルーティンワーク誰か代わりにやってくれないかな…

仕事つまんないな…

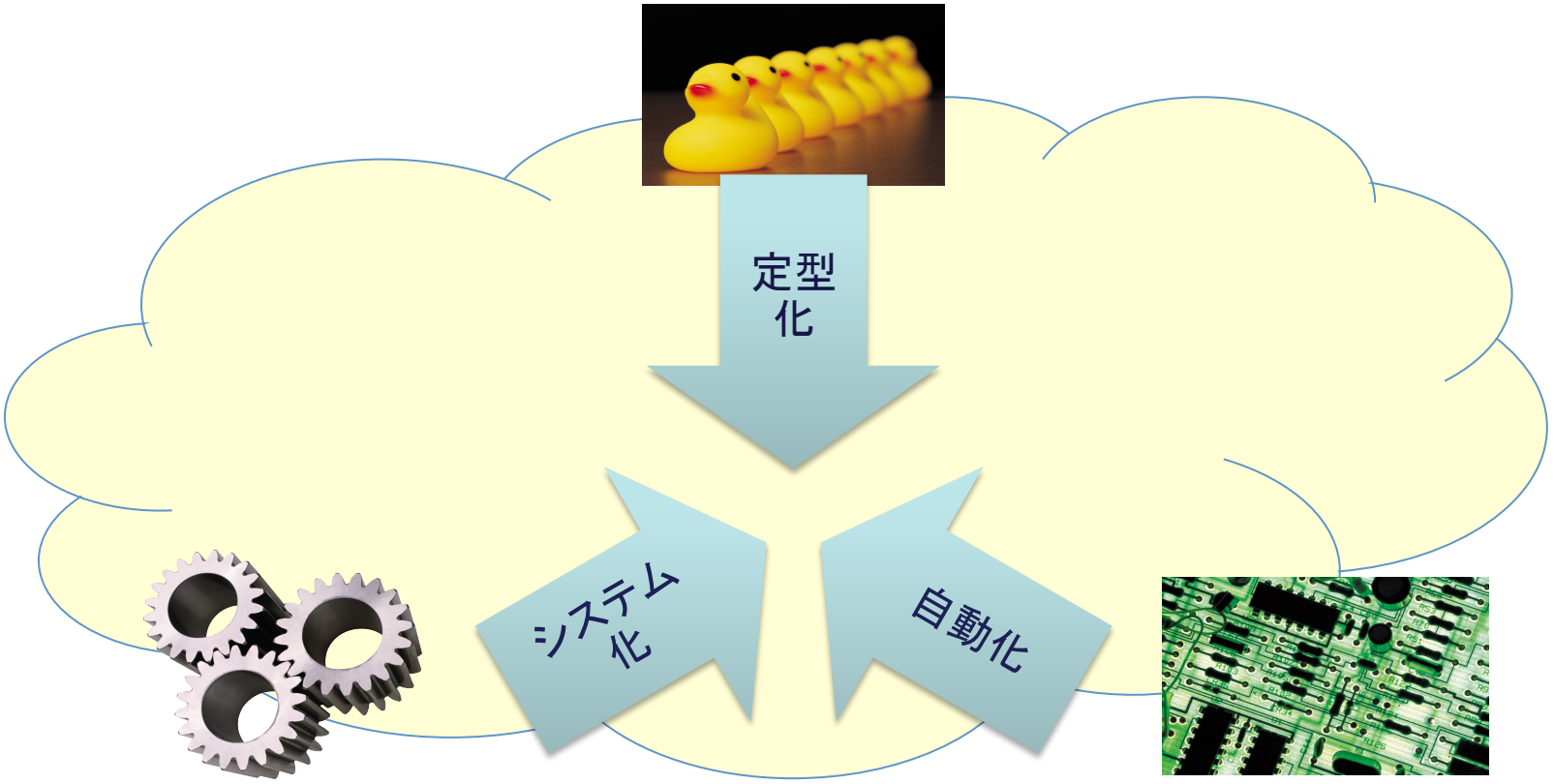
壊れても勝手に復旧できないかな…

朝起きたら工事終わってたらいいのに…

上司の顔みたくないな…

誰か代わりに運用[illegible]れないかな…

悩みを解決するには…



- 情報管理を効率化し、最新の情報で運用性向上
- 各種作業のオートメーション化による効率性向上
- システム化による稼働の削減、ミスの軽減

本日の発表

- 自動化というキーワードで3つのデモ

1. 管理:
   ネットワークから情報収集し簡単に管理

2. 設定:
   ルータの設定を自動化

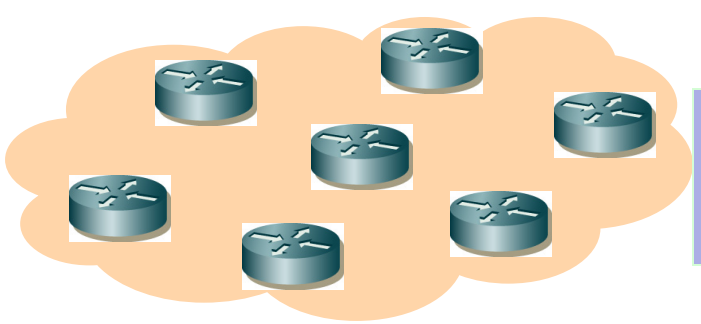
3. 運用:
   （障害時に自動で切り分け、対処）

Demo1:
ネットワークの情報管理

- ネットワークからノードの情報を収集し、DBで管理
- ノード情報からトポロジ情報を生成

ネットワーク情報の収集

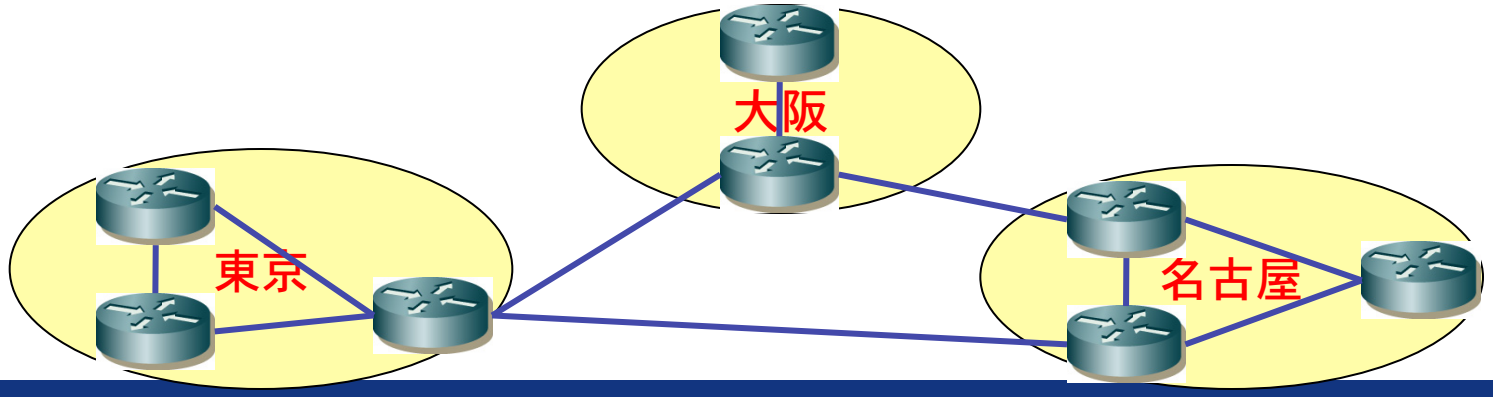
- SNMPでの取得/設定から抜き出す
- ルータ、インタフェースなどをグループ化
- ネットワークトポロジー等を生成



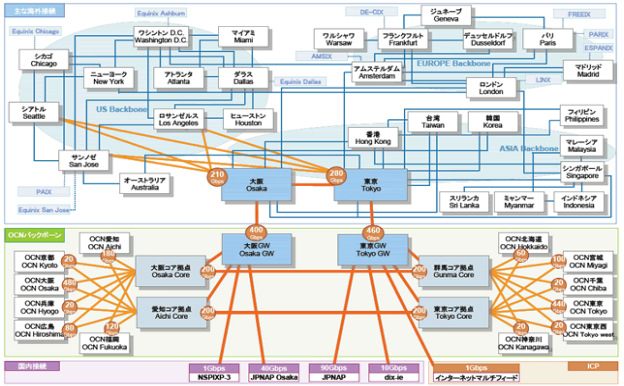
情報抽出

ノードリスト	IP	POP	機能	ベンダ
g01a.tky	192.168.0.1	東京	GW	A社
a02c.tky	192.168.0.2	東京	顧客接続	C社
a03b.tky	192.168.0.3	東京	顧客接続	B社
g01c.osk	192.168.2.1	大阪	GW	C社
a02b.osk	192.168.2.2	大阪	顧客接続	B社
p01a.ngo	192.168.8.1	名古屋	ピア接続	A社
g02b.ngo	192.168.8.2	名古屋	GW	B社
g03b.ngo	192.168.8.3	名古屋	GW	B社

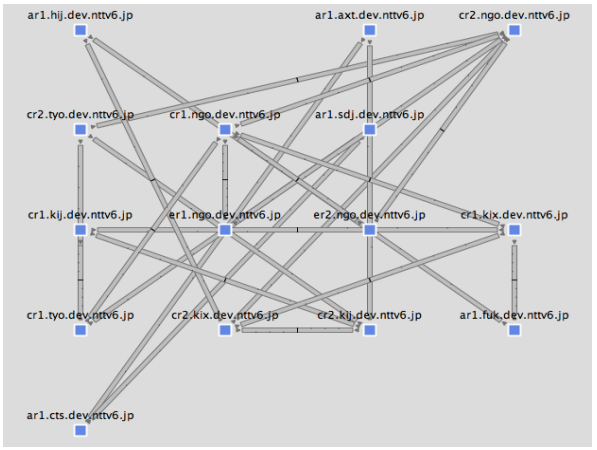
ネットワーク構成



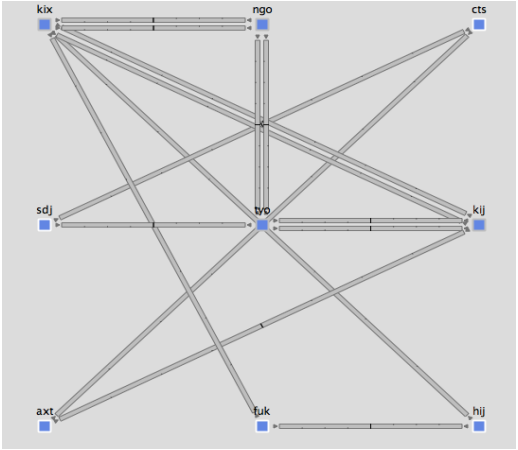
トポロジの簡単生成



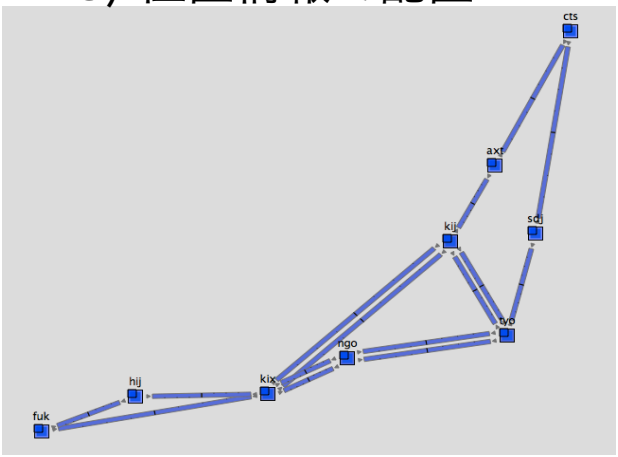
1) ネットワーク接続



2) POPで階層化



3) 位置情報で配置



4) Geographic表示

5) トラフィック挿入

Demo

管理のためのポイント

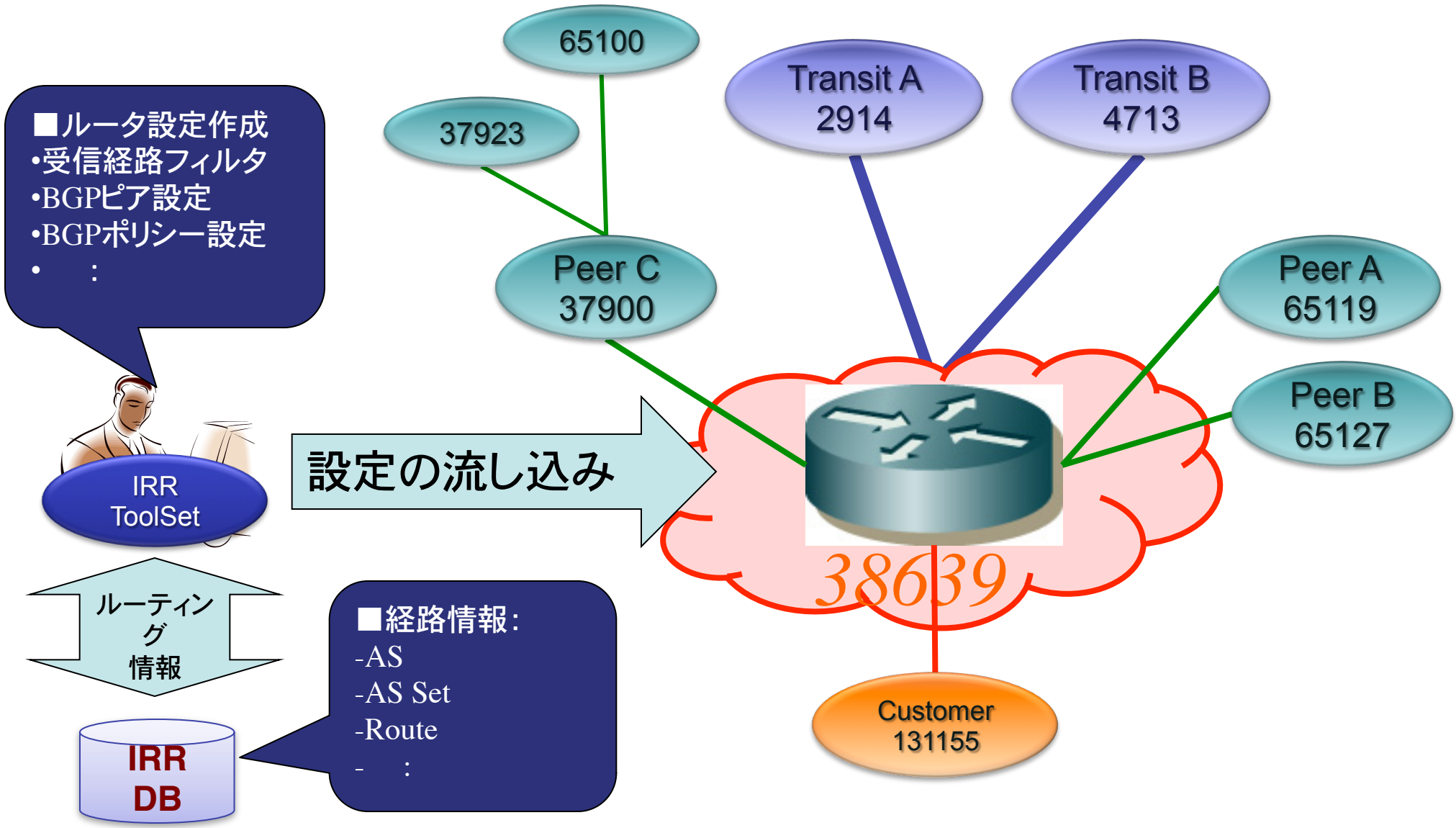
- 名前（ネーミングルール）を統一する
  - ホスト名、IF Descriptionなどをルール化
  - 管理したい情報/IDなどを盛り込む
  - bit、正規表現などで識別できるようにする

- 可能になる事
  - ネットワークのトポロジの自動管理
  - タグによるグルーピング、情報検索
  - グループ単位の操作、管理などが実現

Demo2:
ルータの自動設定

- IRRに経路・AS等ルーティングに必要な情報を登録
- 登録情報からルータの設定を自動的に生成
- ルータに設定を自動でダウンロード

ネットワーク構成


■ルータ設定作成
•受信経路フィルタ
•BGPピア設定
•BGPポリシー設定
•　　:
IRR
ToolSet
設定の流し込み
ルーティング
情報
■経路情報:
-AS
-AS Set
-Route
-　　:
IRR
DB
65100
37923
Transit A
2914
Transit B
4713
Peer C
37900
Peer A
65119
Peer B
65127
38639
Customer
131155

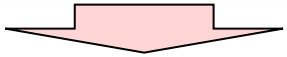
ルータの自動設定（BGP）

Step0)
IRRにAS/AS-Set/Routeオブジェクトを登録

Step1)
config用Templateの作成

Step2)
IRRから経路情報・ルーティング情報を抽出し、ルータの設定を生成

Step3)
生成した設定をルータにDownload



Step4)
ルータに設定が反映され、設定完了

ルータの自動設定(BGP)

Step0)
IRRにAS/AS-Set/Routeオブジェクトを登録

Step1)
config用Templateの作成

Step2)
IRRから経路情報・ルーティング情報を抽出し、ルータの設定を生成

Step3)
生成した設定をルータにDownload

Step4)
ルータに設定が反映され、設定完了

```
> whois -h localhost AS38639
aut-num:    AS38639
as-name:    HANABI
descr:      HANABI by NTT Communications Corporation
import:     from AS2914
            action pref=0;
            accept ANY
import:     from AS4713
            action pref=10;
            accept ANY
import:     from AS37900
            accept AS37900 AS37900:AS-CHILDREN
import:     from AS131155
            accept AS131155
export:     to AS2914
            announce AS38639 AS131155
export:     to AS4713
            announce AS38639 AS131155
export:     to AS37900
            announce AS38639 AS131155
export:     to AS131155
            announce ANY
```

ルーティングポリシを記述

- IRRにBGPピア接続に関するルーティングポリシーを記述
- 受信経路・広告経路のフィルタ、受信経路に対するLP/metric/communityなどを記述可能

ルータの自動設定(BGP)

Step0)
IRRにAS/AS-Set/Routeオブジェクトを登録

Step1)
config用Templateの作成

Step2)
IRRから経路情報・ルーティング情報を抽出し、ルータの設定を生成

Step3)
生成した設定をルータにDownload

Step4)
ルータに設定が反映され、設定完了

```
> more template.rpsl

@rtconfig import AS38639 192.0.2.2   AS2914   192.0.2.1
@rtconfig import AS38639 192.0.2.6   AS4713   192.0.2.5
@rtconfig import AS38639 192.0.2.21  AS37900  192.0.2.22
@rtconfig import AS38639 192.0.2.33  AS131155 192.0.2.34

@rtconfig export AS38639 192.0.2.2   AS2914   192.0.2.
@rtconfig export AS38639 192.0.2.6   AS4713   192.0.2.
@rtconfig export AS38639 192.0.2.21  AS37900  192.0.2
@rtconfig export AS38639 192.0.2.33  AS131155 192.0.2
```

ピア毎に設定を投入

- 上記の例は、4つのBGP PeerをTemplateに記述
- それぞれ、受信経路・広告経路に対する設定を生成するようにしている

ルータの自動設定(BGP)

Step0)
IRRにAS/AS-Set/Routeオブジェクトを登録

Step1)
config用Templateの作成

Step2)
IRRから経路情報・ルーティング情報を抽出し、ルータの設定を生成

Step3)
生成した設定をルータにDownload

Step4)
ルータに設定が反映され、設定完了

```
no access-list 100
access-list 100 permit ip 10.37.90.0    0.0.1.0    255.255.255.0    0.0.0.0
access-list 100 permit ip 10.37.92.0    0.0.0.0    255.255.255.0    0.0.0.0
access-list 100 deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
!
no route-map MyMap_37900_5
!
route-map MyMap_37900_5 permit 1
 match ip address 100
exit
!
router bgp 38639
!
 neighbor 192.0.2.22 remote-as 37900
 neighbor 192.0.2.22 route-map MyMap_37900_5 in
!
exit
!
no access-list 102
access-list 102 permit ip 103.2.128.0    0.0.0.0    255.255.252.0    0.0.0.0
access-list 102 permit ip 115.69.224.0    0.0.0.0    255.255.248.0    0.0.0.0
access-list 102 deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
!
no route-map MyMap_37900_11
!
route-map MyMap_37900_11 permit 1
 match ip address 102
exit
!
router bgp 38639
!
 neighbor 192.0.2.22 remote-as 37900
 neighbor 192.0.2.22 route-map MyMap_37900_11 out
!
exit
```

Templateを展開して設定を生成

ルータの自動設定(BGP)

Step0)
IRRにAS/AS-Set/Routeオブジェクトを登録

Step1)
config用Templateの作成

Step2)
IRRから経路情報・ルーティング情報を抽出し、ルータの設定を生成

Step3)
生成した設定をルータにDownload

Step4)
ルータに設定が反映され、設定完了

```
<snip>

# open terminal
spawn /usr/bin/telnet -l $USER $HOST

# logging in...
expect "Password:"; send "$PASS¥r"
sleep 1

# enter the configuration mode and "load merge"
expect ">"; send "configure exclusive¥r"; sleep 1
expect "#"; send "load replace terminal¥r"; sleep 1
expect "input]";

# read and paste
set fileID [open $FILE]
while {! [eof $fileID]} {
            set line [gets $fileID]
            send -h "$line¥r"
}
close $fileID
# send Ctrl+D
send "¥r¥004"

# check, commit, then close
expect "#"; send "¥r"
expect "#"; send "show | compare | no-more¥r"
expect "#"; send "commit check¥r"
expect "#"; send "commit¥r"
expect "#"; send "exit¥r"
expect ">"; send "exit¥r"
```

loginし生成した設定を流し込む

Demo

管理のためのポイント

- IRRに必要なルーティング情報を“正確”に登録する
  - 広告経路、受信経路（フィルター）
  - 送受信経路に対するBGPアトリビュート制御のポリシー

- ルータの設定の生成
  - Cisco IOS/Juniper JUNOSに対応
  - ルータの情報を管理（ベンダ/ファームウェアなど）
  - expectなどのツールと組み合わせると自動でルータの設定まで可能
  - スケジューリング（cron）し、定期的に実行

IRRToolSetを使ってできること

- 対応機器
  - Cisco　IOS
  - Juniper　JUNOS

- BGPの経路制御
  - 経路Filter
    - AS-PathでのFilter
    - PrefixでのFilter（exact match）
    - AS-Path＋Prefix
    - 特定経路のFilter（0.0.0.0/0, 10.0.0.0/8, …）
    - 特定ASのFilter（AS64512～AS65534など）

    ＃これらの組み合わせも可能

IRRToolSetを使ってできること

- BGPの経路制御（cont.）
  - BGP Attributeの操作
    - Local-Preference（注意：Default値からの差分を登録）
    - Metric （追加のみ）
    - community (付替え、追加、削除)
    - AS-Path prepend
    - nexthop（nexthop-self含む）

導入に必要なもの

- 費用:
  - IRRサーバ(H/W) 20万円
- 稼働: 2日
  - IRRサーバ構築(データ投入込み): 2h
  - Rtconfig/RPSLのお勉強: 1d
  - expectのスクリプト作成: 2h
  - テスト: 4h

２日間で動く状態に

最後に

- システム化・自動化は不可能ではない
- 手頃な所から自動化すると楽になるのでは…

- そのためには、
- 工事・運用のプロセスをなるべく定型化
- 定型化したプロセスに合わせたシステム化
- 品質は、定型プロセス・システム次第
- 最後は、運用をドラスティックに変える決断