

Ansibleを使ったネットワーク運用 ～ ENOG 45 Meeting ～

(株)NS・コンピュータサービス
江部 仁士

話の流れ

- 構成管理ツールの選定
- Ansibleの概要
- Ansible インストール ~ 設定 ~ 実行の流れ
- 検証



構成管理ツールの選定

構成管理ツールの選定

なぜ構成管理ツールを使うのか？

- 構築面のメリット
 - 作業工数の削減
 - 複数の機器を一括で変更可能
 - 品質の保証
 - 複数の機器に対して同じ設定が投入されるため漏れがなくなる
 - 誰が実施しても同じ結果(品質)となる
- 運用面のメリット
 - 人為的な操作ミスの削減
 - 操作内容をコード化するため、事前にレビューできる
 - 作業忘れや作業漏れをなくせる

構成管理ツールの選定

有名な構成管理ツール

ツール名	使用言語	手順書の形式	エージェント
Ansible	Python	YAML	不要 (SSH)
Chef	Ruby	Ruby	必要
Puppet	Ruby	独自	必要
Itamae	Ruby	Ruby	不要 (SSH)

構成管理ツールの選定

最近のトレンドはAnsible？

Google Trends

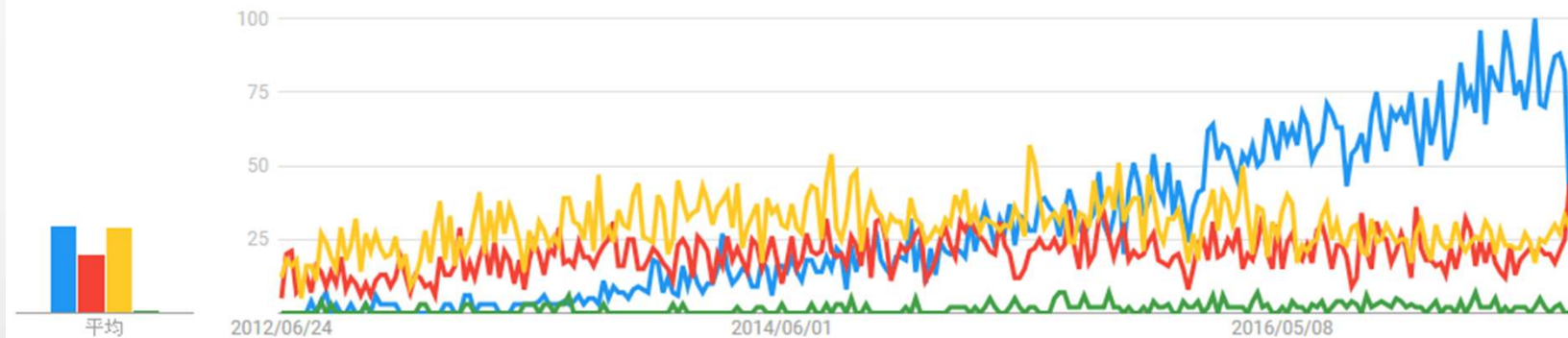
■ **install ansible**
検索キーワード
すべての国, 過去 5...

■ **install chef**
検索キーワード
すべての国, 過去 5...

■ **install puppet**
検索キーワード
すべての国, 過去 5...

■ **itamae**
検索キーワード
すべての国, 過去 5...

人気度の動向 ?



Ansible の概要

Ansible の 概要

Ansibleについて

- Python ベースの構成管理ツール
- 2015年 RedHatがAnsibleを買収
- 最新バージョンは v2.3



ANSIBLE



Ansible の 概要

Ansibleの特徴

- エージェントレス
 - 管理対象の機器に対してエージェントソフトのインストールが不要
 - SSHでターゲットを管理
- 手順書(PlayBook)はYAML形式の記述
 - 読みやすい、書きやすい、わかりやすい
- 冪等性を考慮したモジュール
 - 同じ処理を何回実行しても同じ結果になる
- 多数の製品に対応している
 - すぐに利用できるモジュールが充実している
- ●

Ansible の 概要

Ansibleで操作可能なNWベンダー



など

Ansible の 概要

Ansibleラインナップ

- Ansible Core

- 自動化を提供する基盤
- オープンソース



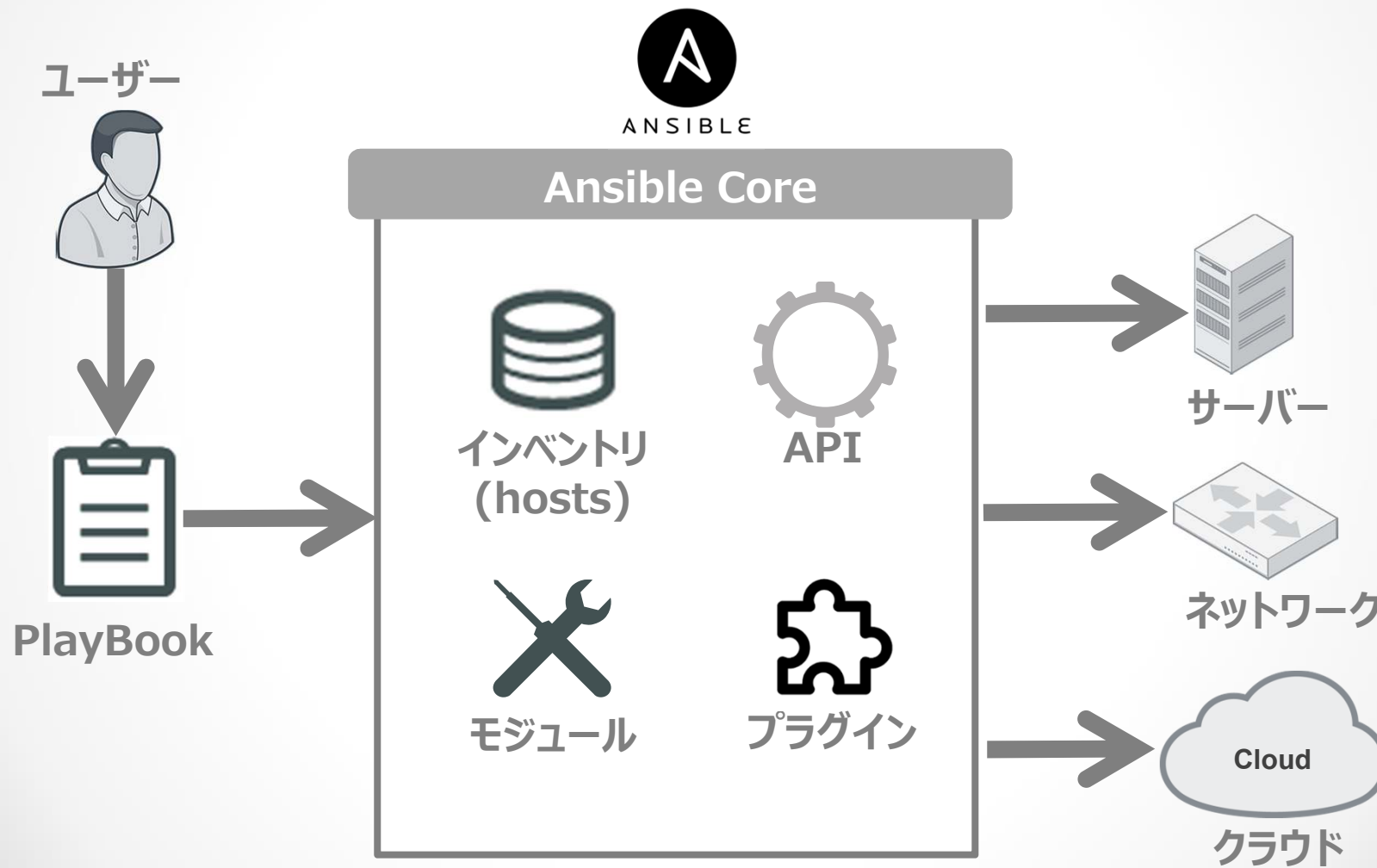
- Ansible Tower

- Ansible Coreを管理するためのWeb UIツール
 - ダッシュボード (可視化)
 - ユーザーの権限管理
 - ジョブコントロール
- 管理対象10ノードまでならフリーライセンスが提供されている



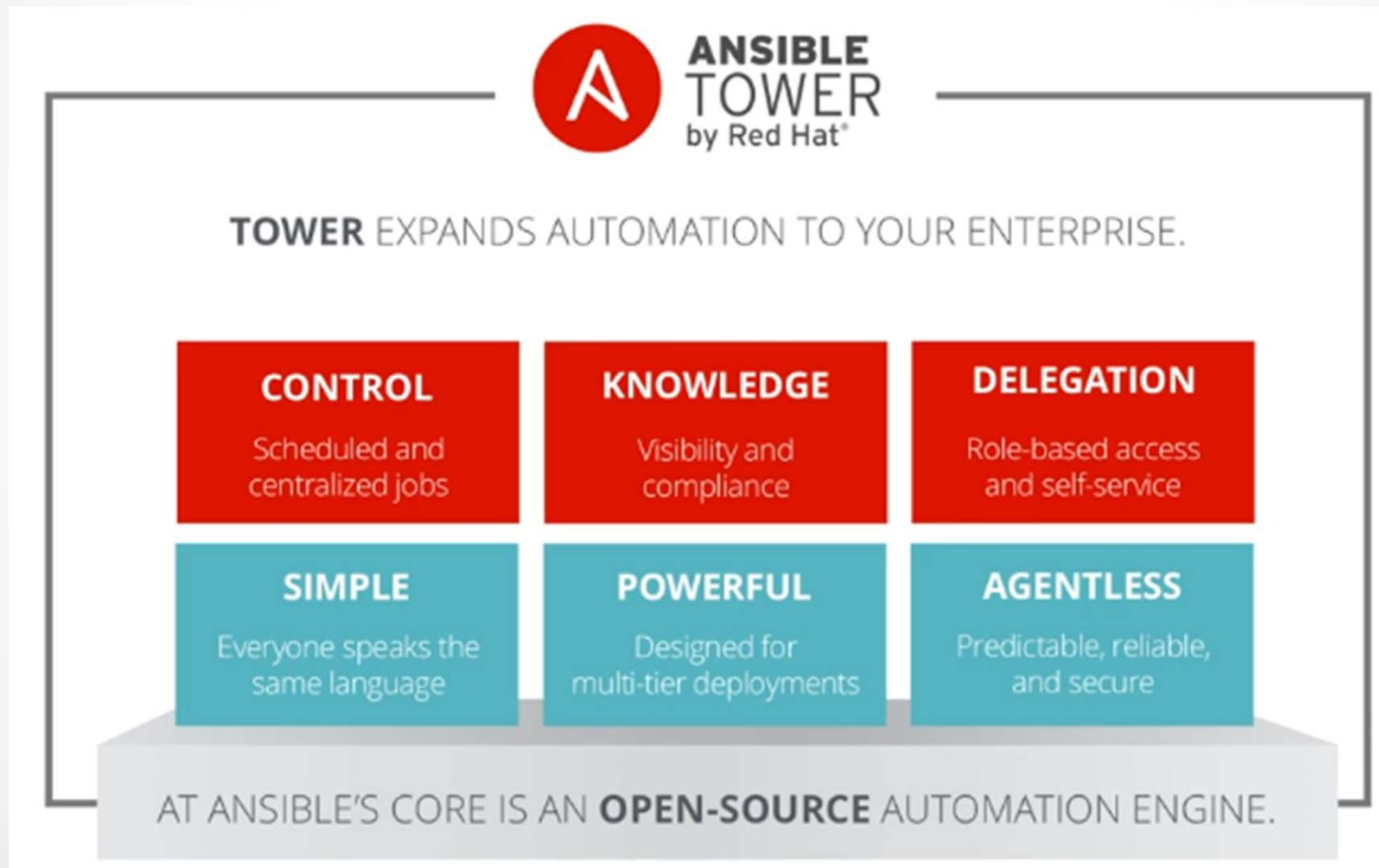
Ansible の概要

Ansible Core



Ansible の概要

Ansible Tower



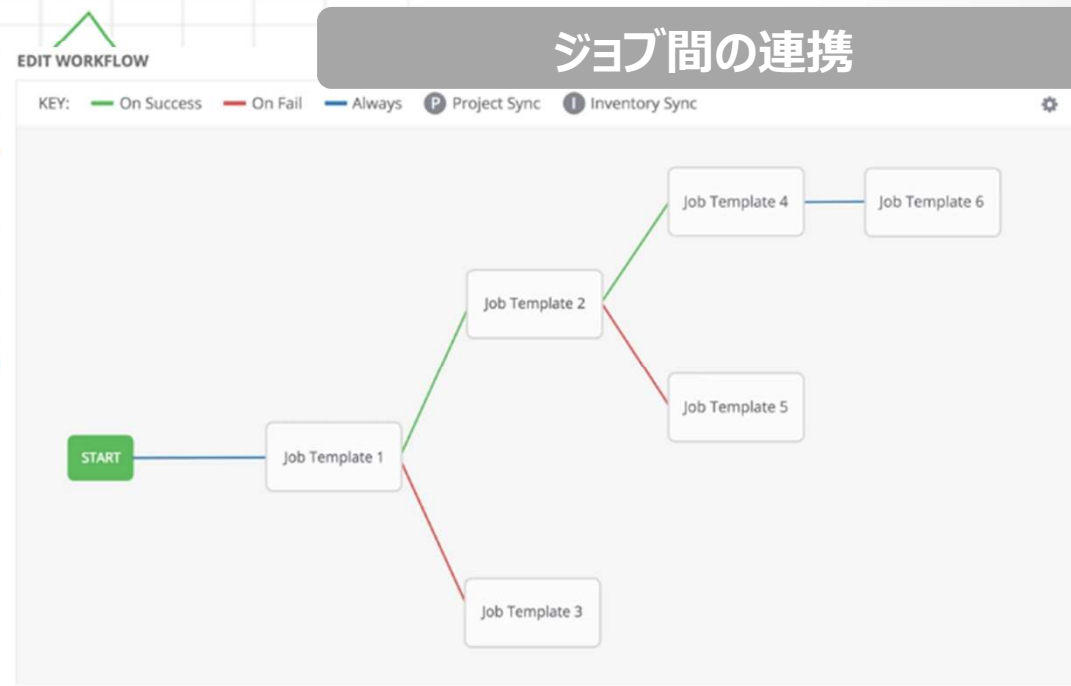
Ansible の概要

Ansible Tower Web UI

ダッシュボード管理



ジョブ間の連携



Ansible の 概要

Ansible Towerライセンス

- 価格

License	Standard	Premium
100ノード ライセンス	130万円/年	182万円/年

- ライセンスの違いはサポート内容
 - Standard サポート時間: 8 x 5 support
 - Premium サポート時間: 24 x 7 support
 - Premiumの方がサポート時の対応時間が優遇されている

- 機能面に違いはない
 - <https://www.ansible.com/tower-editions>



Ansible

インストール ~ 設定 ~ 実行の流れ

Ansible インストール ~ 設定 ~ 実行の流れ

Ansible 実行までの流れ



Ansible インストール ~ 設定 ~ 実行の流れ

Ansible 実行までの流れ

ターゲット側の設定(IPアドレス/SSH設定等)

Ansible インストール

インベントリファイル(hosts) / PlayBookの記述

PlayBookの実行

実行結果の確認

Ansible インストール ~ 設定 ~ 実行の流れ

Cisco IOSのSSH設定

Step1: ユーザー認証の設定

```
(config)# username "ユーザー名" password "パスワード"
```

Step2: line vtyにローカル認証の設定

```
(config)# line vty 0 4
```

```
(config-line)# login local
```

Step3: ホスト名、ドメイン名の設定(SSH暗号鍵生成のため)

```
(config)# hostname "ホスト名"
```

```
(config)# ip domain-name "ドメイン名"
```



Ansible インストール ~ 設定 ~ 実行の流れ

Cisco IOSのSSH設定

Step4: RSA暗号鍵の生成

```
(config)# crypto key generate rsa
```

鍵長は1024bitで設定してます。

Step5: SSHバージョンの設定

```
(config)# ip ssh version 2
```

Step6: SSH接続の許可設定

```
(config)# line vty 0 4
```

```
(config-line)# transport input ssh
```



Ansible インストール ~ 設定 ~ 実行の流れ

Ansible 実行までの流れ



Ansible インストール ~ 設定 ~ 実行の流れ

Ansible インストール (CentOS7)

Python 2.6以降が必要。

Step1: EPELリポジトリを追加

```
# yum install epel-release
```

Step2: Ansibleをインストール

```
# yum install ansible
```

Step3: インストール後のバージョン確認

```
# ansible --version
```

```
ansible 2.3.0.0
```

```
config file = /etc/ansible/ansible.cfg
```

```
~~
```

```
• python version = 2.7.5  ~~ •
```

Ansible インストール ~ 設定 ~ 実行の流れ

Ansible 実行までの流れ



Ansible インストール ~ 設定 ~ 実行の流れ

インベントリファイルについて

- インベントリファイルとは
 - 管理対象のホストやホストグループを定義するファイル
 - 各種変数の定義も可能
- インベントリファイルの場所
 - ansible.cfg内でインベントリファイルを指定している
Hostfile = [インベントリファイルのパス]
 - デフォルト: /etc/ansible/hosts
 - コマンド実行時に指定
ansible-playbook -i [インベントリファイルパス] [プレイブックパス]

Ansible インストール ~ 設定 ~ 実行の流れ

インベントリファイルの記述

- インベントリファイルの記述

/etc/ansible/hosts

[webserver]	…[]でホストグループを定義
web1.nscs.jp	…ホストグループに属させる個別ホストを定義
web2.nscs.jp	
[webserver:vars]	…[ホストグループ名:vars] で変数の存在を定義
key1= value1	…"変数 = 値"の形式で定義
key2= value2	

Ansible インストール ~ 設定 ~ 実行の流れ

インベントリファイルの記述例

数字連番の省略記法

[webserver]

web1.nscs.jp

… web[1:3].nscs.jp に省略可

web2.nscs.jp

web3.nscs.jp

英字連番の省略記法

[webserver]

webA.nscs.jp

… web[A:C].nscs.jpに省略可

webB.nscs.jp

webC.nscs.jp

Ansible インストール ~ 設定 ~ 実行の流れ

PlayBookについて

- PlayBookとは
 - 手順書のようなもの
 - YAML形式のテキストファイル
- PlayBookの場所
 - PlayBookの実行時にパスを指定する
ansible-playbook [プレイブックパス]
- ベストプラクティス
 - http://docs.ansible.com/ansible/playbooks_best_practices.html



Ansible インストール ~ 設定 ~ 実行の流れ

PlayBookの記述

- PlayBookの記述

```
/etc/ansible/test.yml
```

```
- hosts: "管理対象"           … ホスト / ホストグループ名 / all
```

```
tasks:                       … 管理対象への指示内容
```

```
  - name: "タスク名"
```

```
    "モジュール":
```

```
      - "属性"
```

…操作で使用するモジュールを指定

…モジュールに渡す属性を指定

インデントは
半角スペースで！

モジュールと属性を指定することで
管理対象を特定の状態へ推移させる

Ansible インストール ~ 設定 ~ 実行の流れ

インベントリファイル/PlayBookの記述

- PlayBookのCisco IOS記述例

```
/etc/ansible/test.yml
```

```
- hosts: cisco-router
```

```
tasks:
```

```
  - name: set hostname
```

```
    ios_config:
```

```
      authorize: yes
```

```
      username: "{{ ssh_user }}"
```

```
      password: "{{ ssh_pass }}"
```

```
      auth_pass: "{{ enable_pass }}"
```

```
      lines:
```

```
        - hostname nscs-r1
```

Ansible インストール ~ 設定 ~ 実行の流れ

Cisco IOS モジュール

- ios_command
 - 特権モードでコマンドを実行するためのモジュール
- ios_config
 - 設定するときに使用するモジュール
 - Version2.2で有能なオプションが多数追加されている
 - Backup: 変更前にrunning-configのバックアップが実行される
 - Defaults: 設定前に"show running-config all"が実行される
- ios_template
 - テンプレートファイルを別に用意しておき、それを流し込むためのモジュール

•

•

Ansible インストール ~ 設定 ~ 実行の流れ

Cisco IOS モジュール

- ios_banner (v2.2)
 - バナーを設定するためのモジュール
- ios_vrf (v2.3)
 - VRF定義、管理するためのモジュール
- ios_facts (v2.2)
 - システム情報やインターフェース情報を収集するモジュール
- ios_system (v2.3)
 - Manage the system attributes on Cisco IOS devices

•

•

Ansible インストール ~ 設定 ~ 実行の流れ

Cisco IOS以外のモジュール

- Cisco IOS-XR
 - iosxr_command - Run commands on remote devices running Cisco IOS XR
 - Iosxr_config - Manage Cisco IOS XR configuration sections
 - Iosxr_fact - Collect facts from remote devices running IOS XR
 - iosxr_system - Manage the system attributes on Cisco IOS XR devices
 - Iosxr_template - Manage Cisco IOS XR device configurations over SSH
- Cisco ASA
 - asa_acl - Manage access-lists on a Cisco ASA
 - asa_command - Run arbitrary commands on Cisco ASA devices
 - asa_config - Manage configuration sections on Cisco ASA devices



Ansible インストール ~ 設定 ~ 実行の流れ

Ansible 実行までの流れ



Ansible インストール ~ 設定 ~ 実行の流れ

PlayBookの実行

- 実行コマンド

 - # ansible-playbook "PlayBookパス"

- オプション

 - --syntax-check
 - PlayBookの文法チェック
 - -C , --check
 - 変更は行わないが条件の確認などは実行
 - --start-at-task=START_AT
 - 指定のタスクから実行する



Ansible インストール ~ 設定 ~ 実行の流れ

Ansible 実行までの流れ



Ansible インストール ~ 設定 ~ 実行の流れ

実行結果の確認

```
# ansible-playbook show_bgp_summary.yml -u cisco -k  
SSH password:
```

```
PLAY [bgp]  
*****
```

```
TASK [show ip bgp summary]  
*****
```

```
ok: [192.168.1.21]
```

```
PLAY RECAP  
*****
```

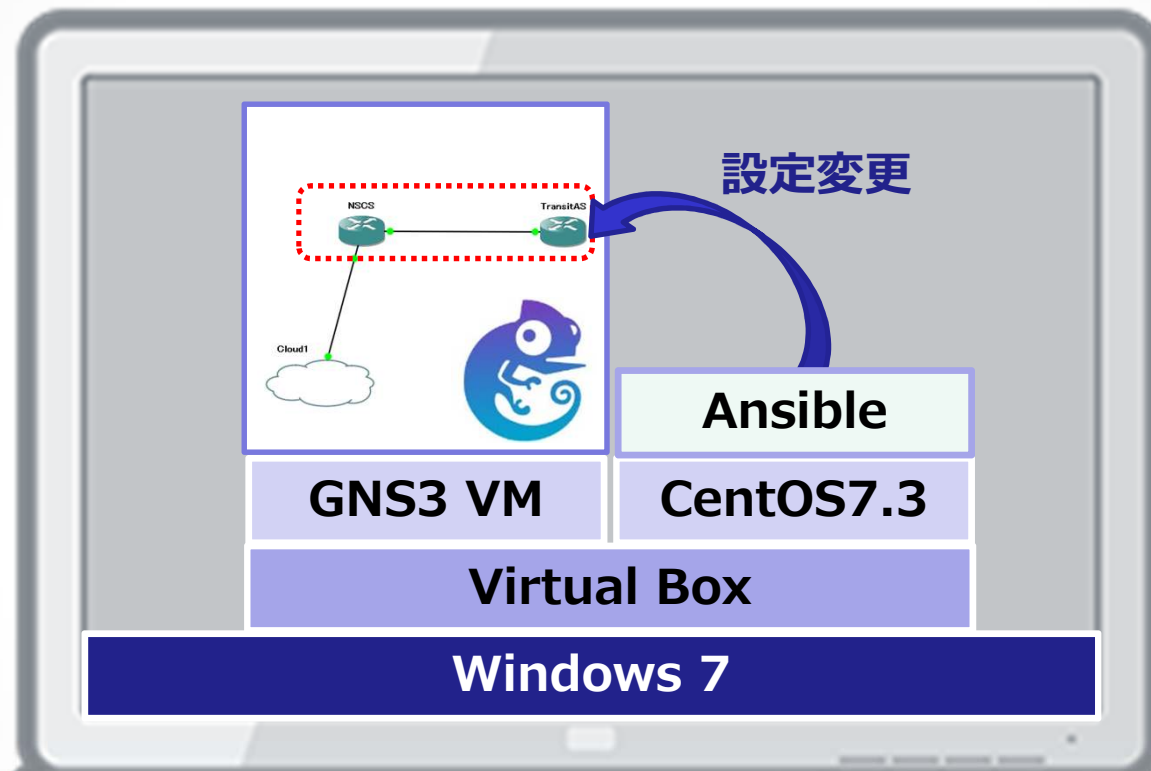
```
192.168.1.21 : ok=1  changed=0  unreachable=0  failed=0
```

状態に変化があればchangeが上がる

Ansible 検証

Ansible 検証

検証環境の準備



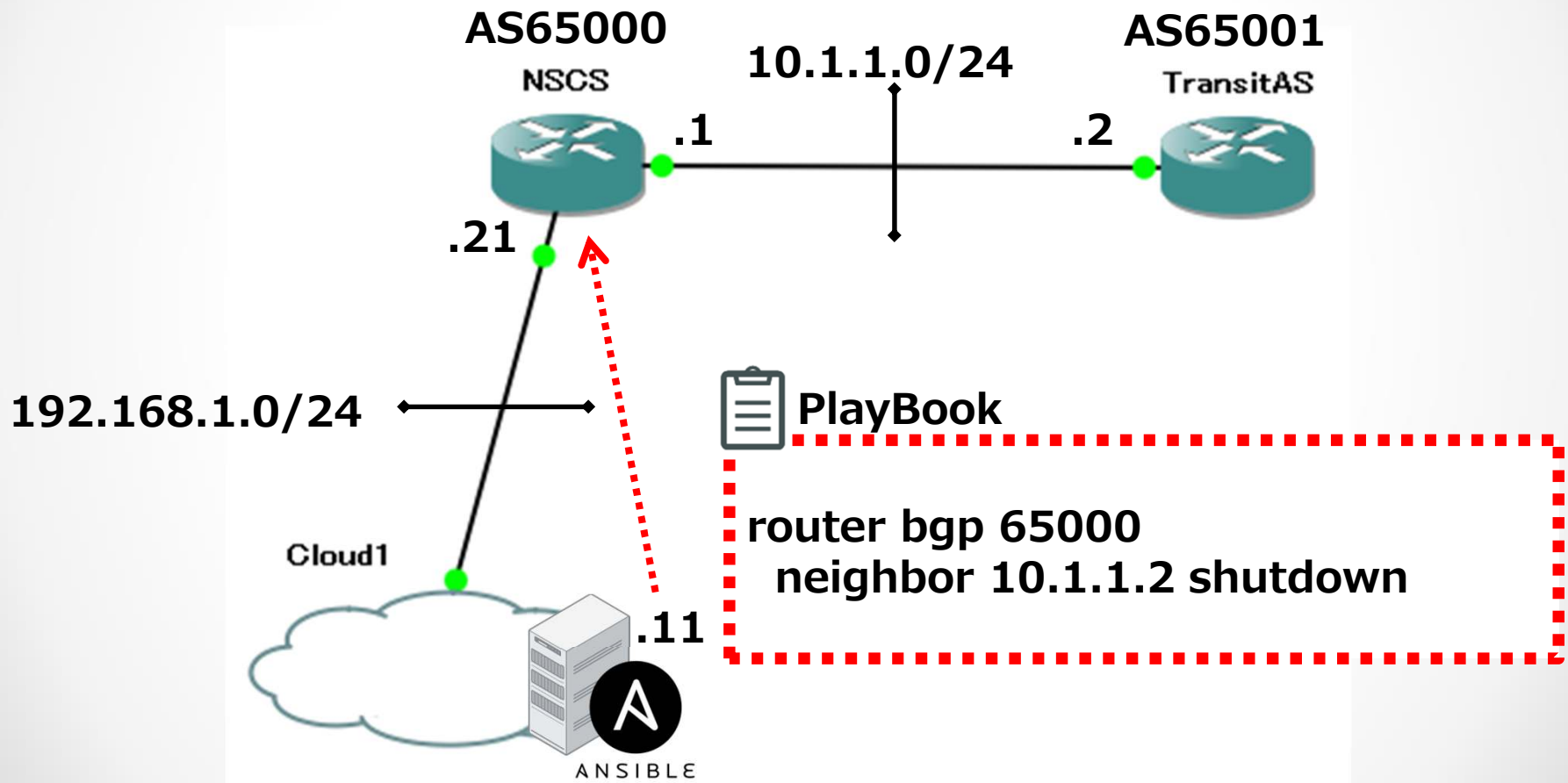
必要なものは1台のノートPCに集約

すべてのノードが同じネットワーク環境

Lenovo X230

Ansible 検証

検証環境の準備



Ansible 検証

インベントリ(hosts)の中身

```
# cat hosts
```

```
[bgp]
```

```
192.168.1.21
```

ルーターのIPアドレス

```
[bgp:vars]
```

```
ssh_user=cisco
```

```
ssh_pass=cisco0
```

```
ena_pass=cisco1
```



SSH認証情報の定義

Ansible 検証

PlayBookの中身

```
# cat bgp_AS65001_shut.yml
```

```
- hosts: bgp
```

```
gather_facts: no
```

```
connection: local
```

```
tasks:
```

```
- name: NSCS R1 AS65001 BGP Shutdown
```

```
ios_config:
```

```
authorize: yes
```

```
username: "{{ ssh_user }}"
```

```
password: "{{ ssh_pass }}"
```

```
auth_pass: "{{ ena_pass }}"
```

```
lines: neighbor 10.1.1.2 shutdown
```

```
parents: router bgp 65000
```

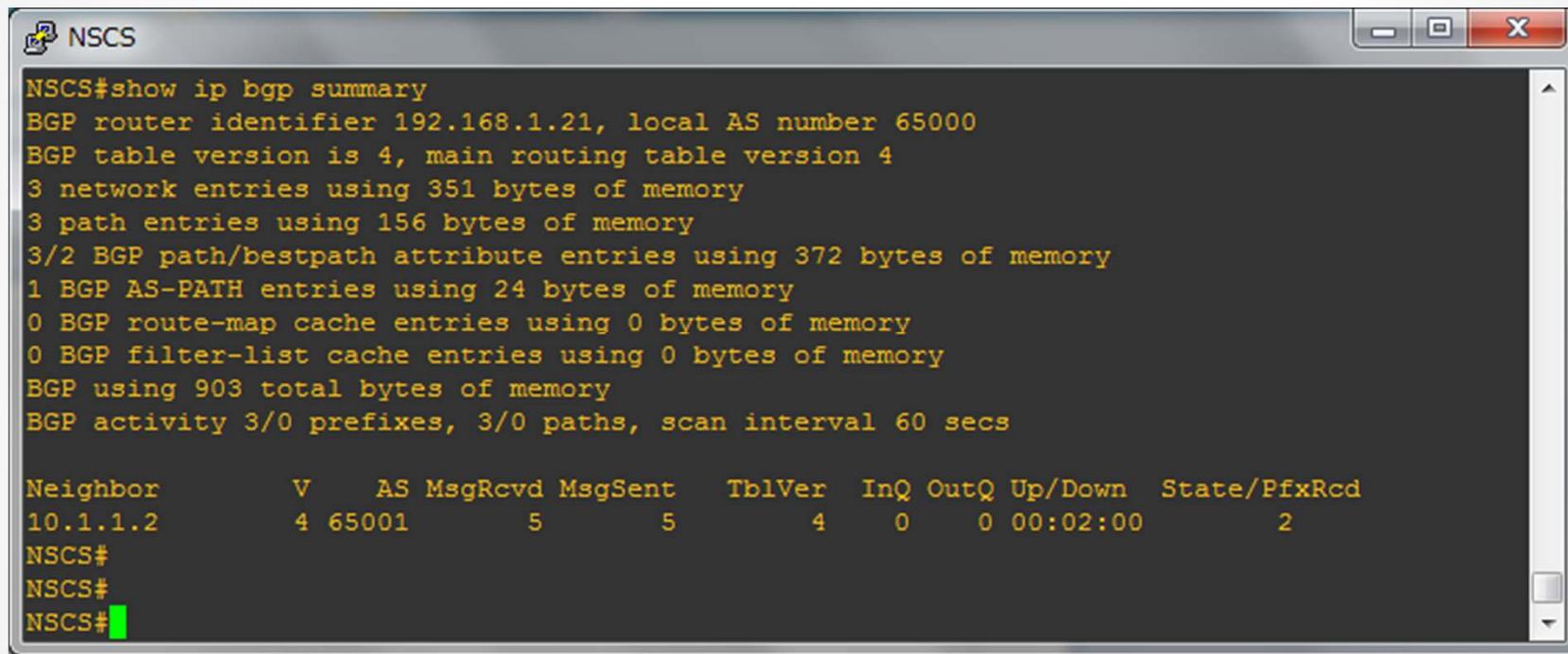
ターゲット情報の収集を無効化

実行したいコマンド

コマンドを実行する階層を指定

Ansible 検証

PlayBookの実行前の状態



```
NSCS#show ip bgp summary
BGP router identifier 192.168.1.21, local AS number 65000
BGP table version is 4, main routing table version 4
3 network entries using 351 bytes of memory
3 path entries using 156 bytes of memory
3/2 BGP path/bestpath attribute entries using 372 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 903 total bytes of memory
BGP activity 3/0 prefixes, 3/0 paths, scan interval 60 secs

Neighbor      V    AS MsgRcvd MsgSent  TblVer  InQ  OutQ Up/Down  State/PfxRcd
10.1.1.2      4 65001     5      5        4    0    0 00:02:00      2
NSCS#
NSCS#
NSCS#
```

Ansible 検証

PlayBookの実行

```
192.168.1.11:22 - root@ansible:/etc/ansible/playbook VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
*
TASK [NSCS R1 AS65001 BGP Shutdown] *****
*
[WARNING]: argument username has been deprecated and will be removed in a
future version

[WARNING]: argument auth_pass has been deprecated and will be removed in a
future version

[WARNING]: argument password has been deprecated and will be removed in a
future version

changed: [192.168.1.21]

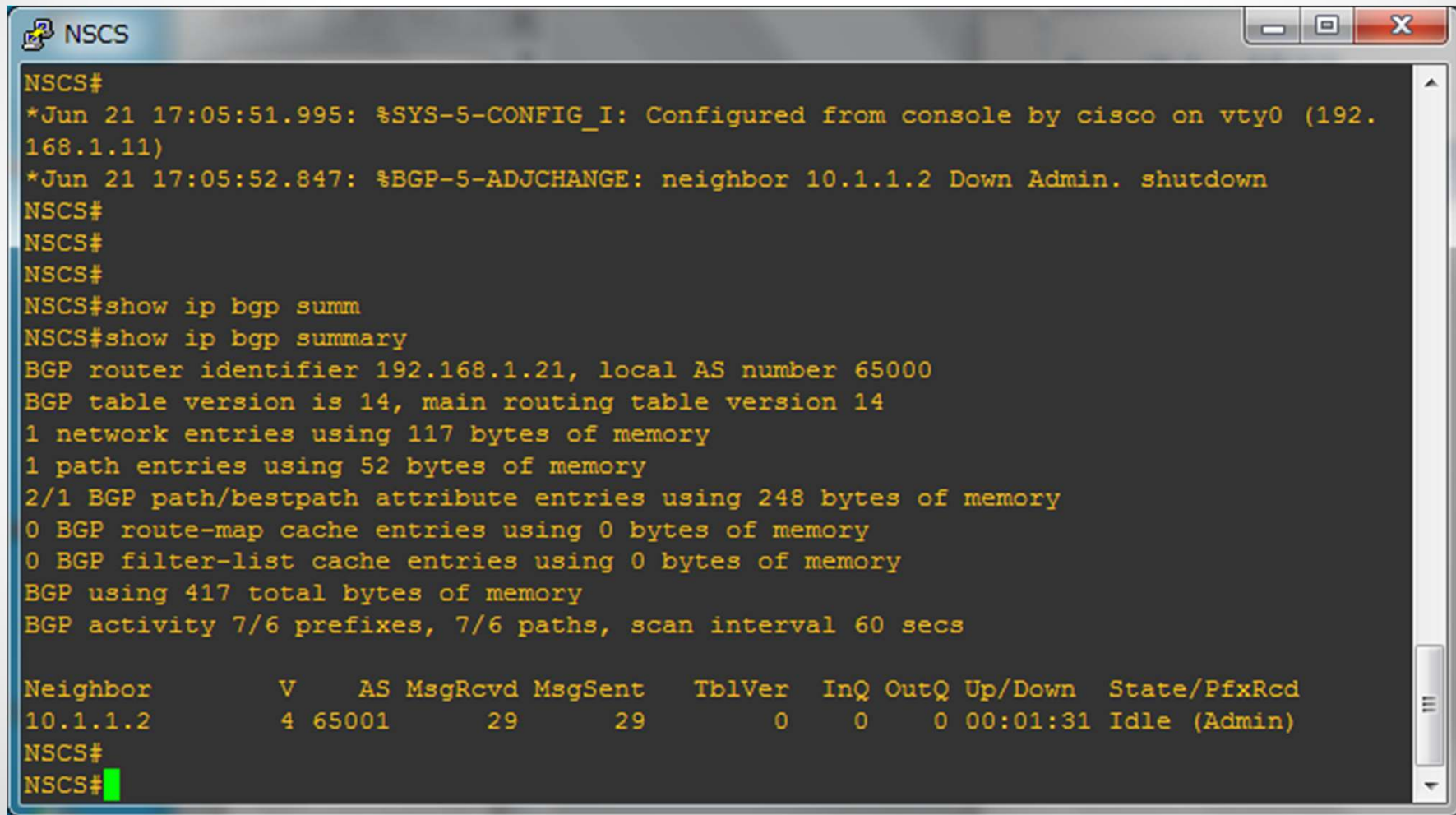
PLAY RECAP *****
*
192.168.1.21      : ok=1    changed=1    unreachable=0    failed=0

[root@ansible playbook]#
```

```
NSCS
NSCS#
NSCS#
NSCS#
NSCS#
*Jun 21 17:05:51.995: %SYS-5-CONFIG_I: Configured from console by cisco on vty0 (192.
168.1.11)
*Jun 21 17:05:52.847: %BGP-5-ADJCHANGE: neighbor 10.1.1.2 Down Admin. shutdown
NSCS#
NSCS#
NSCS#
```

Ansible 検証

PlayBookの実行結果確認



```
NSCS#
*Jun 21 17:05:51.995: %SYS-5-CONFIG_I: Configured from console by cisco on vty0 (192.
168.1.11)
*Jun 21 17:05:52.847: %BGP-5-ADJCHANGE: neighbor 10.1.1.2 Down Admin. shutdown
NSCS#
NSCS#
NSCS#
NSCS#show ip bgp summ
NSCS#show ip bgp summary
BGP router identifier 192.168.1.21, local AS number 65000
BGP table version is 14, main routing table version 14
1 network entries using 117 bytes of memory
1 path entries using 52 bytes of memory
2/1 BGP path/bestpath attribute entries using 248 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 417 total bytes of memory
BGP activity 7/6 prefixes, 7/6 paths, scan interval 60 secs

Neighbor          V    AS MsgRcvd MsgSent   TblVer  InQ  OutQ Up/Down  State/PfxRcd
10.1.1.2          4 65001    29     29       0    0    0 00:01:31 Idle (Admin)
NSCS#
NSCS#
```

まとめ

- 簡単に導入することができるのは間違いない！
- Ansible Coreだけで自動化を楽しめる！
- ドキュメントがしっかり整備されている
 - <https://docs.ansible.com/>
- 実運用環境への投入はテストとレビューを怠るな！
 - 検証時にBGPネイバーを飛ばしている。。。
 - ピアをアップさせる際に“no neighbor 10.1.1.2” を実行していた。。。



Fin

ご清聴いただきありがとうございました

