

# Ansible とネットワーク自動化の概要

セイコーソリューションズ x エーピーコミュニケーションズ 合同ウェビナー  
ZTP(ゼロタッチプロビジョニング)体験ハンズオン



株式会社 エーピーコミュニケーションズ

横地 晃

# 自己紹介



名前 横地 晃

会社 株式会社エーピーコミュニケーションズ

業務 ネットワーク自動化のご支援

コミュニティ Ansible ユーザー会、JANOG



[@akira6592](#)



ブログ(てくなべ)

<https://tekunabe.hatenablog.jp>



過去発表資料

<https://www.slideshare.net/akira6592/>





# Ansible の概要

# 自動化ツール Ansible の特徴

## シンプル

- プログラミング不要
- 構成定義ファイル (Playbook) を利用

## パワフル

- 多数のサーバー、クラウド、**ネットワーク**向けのモジュールを利用可能
- カスタムモジュールも自作可能

## エージェントレス

- 操作対象機器側に専用ソフトはインストール不要

- 構成管理ツールとも呼ばれ、Chef や puppet などと比較されることが多い
- 本資料は Ansible 2.10 を前提

# 50以上のネットワークプラットフォームに対応

- Cisco IOS、IOS-XR、NX-OS、ACI
  - Arista EOS
  - Juniper Junos
  - Fortigate FortiOS
  - F5 BIG-IP
  - Paloalto PAN-OS
- など

国内ベンダーも対応が進んでいる

- seiko.smartcs collection
- yamaha\_network.rtx collection

# Ansible でネットワーク機器にできること

- 接続方式

- SSH、NETCONF、HTTP/HTTPS (API) などで接続

- できることの例

- 参照 (show) コマンド実行による情報取得
- 設定コマンド実行による設定追加・変更・削除






# Ansible を利用するメリット1

## ログインやモード変更の処理を簡略化できる

- 認証情報を定義しておくだけで暗黙的にログイン、ログアウト
- 設定系モジュールは暗黙的にコンフィギュレーションモードへ移行



```
login:  
password:  
conf t
```

# Ansible を利用するメリット2

## プロンプトのハンドリングが簡略化できる

- エラーや正常プロンプトがモジュールに組み込み済み

```
terminal_stdout_re = [  
    re.compile(br"[\r\n]?[\w\+\-\.:\\/\[\]]+(?:\([^\)]+\)){0,3}(?:[>#]) ?$")  
]  
  
terminal_stderr_re = [  
    re.compile(br"% ?Error"),  
    # re.compile(br"^% \w+", re.M),  
    re.compile(br"% ?Bad secret"),  
    re.compile(br"[\r\n%] Bad passwords"),  
    re.compile(br"invalid input", re.I),  
    re.compile(br"(?:incomplete|ambiguous) command", re.I),  
    re.compile(br"connection timed out", re.I),  
    re.compile(br"^[^r\n]+ not found"),  
    re.compile(br"'[^']*' +returned error code: ?\d+"),  
    re.compile(br"Bad mask", re.I),  
    re.compile(br"% ?(\S+) ?overlaps with ?(\S+)"), re.I),  
    re.compile(br"[%\S] ?Error: ?[\s]+", re.I),  
    re.compile(br"[%\S] ?Informational: ?[\s]+", re.I),  
    re.compile(br"Command authorization failed"),  
    re.compile(br"Command Rejected: ?[\s]+", re.I),  
]
```



# Ansible を利用するメリット3

## Ansible の他の機能と連携しやすい

- コマンド出力結果を copy モジュールに渡して、ファイルに保存
- テンプレート機能を利用して、コンフィグを生成して投入
- 監視サーバーに登録されているホスト情報を接続に利用

例えば





# サンプル1: コンフィグバックアップ

# サンプル1: コンフィグバックアップ(準備)

```
- hosts: ios  
gather_facts: no
```

対象ホストグループ

```
tasks:
```

```
- name: show command test  
  cisco.ios.ios_command:  
    commands:
```

実行したい show コマンド  
(変更すれば他のshowコマンドでも対応可)

```
- show running-config
```

```
register: result
```

```
- name: save config to file  
  copy:
```

保存先ファイル名

```
content: "{{ result.stdout[0] }}"
```

```
dest: "show_config_{{ inventory_hostname }}.txt"
```

※このほか、対象ホストの接続情報や、認証情報を定義するファイルを用意する



# サンプル1: コンフィグバックアップ(実行)

A Ansible 側

ansible-playbook コマンドを実行

```
$ ansible-playbook -i inventory show01.yml
```

```
PLAY [fw] *****
```

```
TASK [show command test] *****
```

```
ok: [172.16.0.1]
```

```
TASK [save config to file] **
```

```
changed: [172.16.0.1]
```

コンフィグがファイルとして新たに  
保存された旨のログ

```
Current configuration : 4332 bytes
```

```
!
```

```
! Last configuration change at 13:18:40 UTC Mon May 24  
2021 by admin
```

```
!
```

```
version 15.9
```

```
service timestamps debug datetime msec
```

```
service timestamps log datetime msec
```

```
service password-encryption
```

```
!
```

```
hostname ios01
```

```
!
```

```
...(略)...
```



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

```
172.16.0.1      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



# サンプル2: 設定変更

## サンプル2: 設定変更 (Playbook)

```
- hosts: ios  
gather_facts: no
```

対象ホストグループ

```
tasks:
```

```
- name: config test  
  cisco.ios.ios_config:  
    lines:
```

実行したい設定コマンド  
(変更すれば他の設定コマンドでも対応可)

```
- ntp server 10.0.0.123
```

※このほか、対象ホストの接続情報や、認証情報を定義するファイルを用意する



## サンプル2: 設定変更(実行)

 Ansible 側

ansible-playbook コマンドを実行

```
$ ansible-playbook -i inventory set01.yml
```

```
PLAY [ios] *****
```


設定変更された旨のログ

```
TASK [config test] *****
```

```
changed: [172.16.0.1]
```

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

```
172.16.0.1      : ok=1    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

 ネットワーク機器側

```
ios01# show running-config | inc ntp
```

```
ntp server 10.0.0.123
```

設定変更された

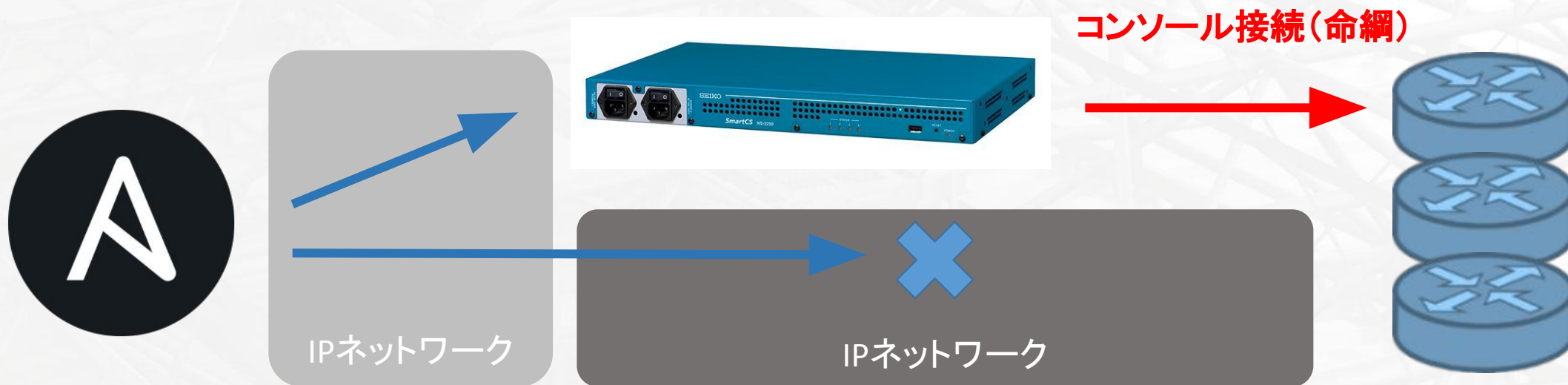


ところで、  
こんな時はどうする？

# こんな時には

## ● IP到達性がない機器に Ansible を使うには？

- Ansible にも対応したコンソールサーバー「SmartCS」で
- 詳細はこの後のパートで







まとめ

# まとめ

- **はじめやすい自動化ツール**

- シンプル、パワフル、エージェントレス

- **多数のネットワークプラットフォームに対応**

- 参照や設定などができる

- **マクロにはないメリットも**

- Ansible の他機能との連携
- ログイン/ログアウトの簡略化や、組み込みのエラー検出、など

- **IP到達性がない機器には SmartCS で操作可能**

# 参考情報

- 公式ドキュメント

- トップ <https://docs.ansible.com/>
- Getting Started [https://docs.ansible.com/ansible/latest/user\\_guide/intro\\_getting\\_started.html](https://docs.ansible.com/ansible/latest/user_guide/intro_getting_started.html)

- ブラウザだけで独習できる日本語コース

- Ansible 101 by irixjp | Katacoda <https://www.katacoda.com/irixjp/scenarios/ansible-101>

- 書籍

- Ansible実践ガイド 第3版  
<https://book.impress.co.jp/books/1118101094>

