



EUREKALABRIA

Idee in connessione

#ideeinconnessione



Automating a Service Provider network

A practical introduction

Elia Aielli

Timenet SpA

Introduzione

WHO

Elia Aielli
Network Engineer
Timenet SpA



WHAT

Fornire mezzi pratici per
introdurre automazione
nella propria rete



Aiutare a superare
l'inerzia iniziale
Imparare nuovi concetti

WHY



Partire da piccoli task
standard e ripetitivi
Trade off tempo/risorse/errori

WHEN



Il progetto

Border ACL

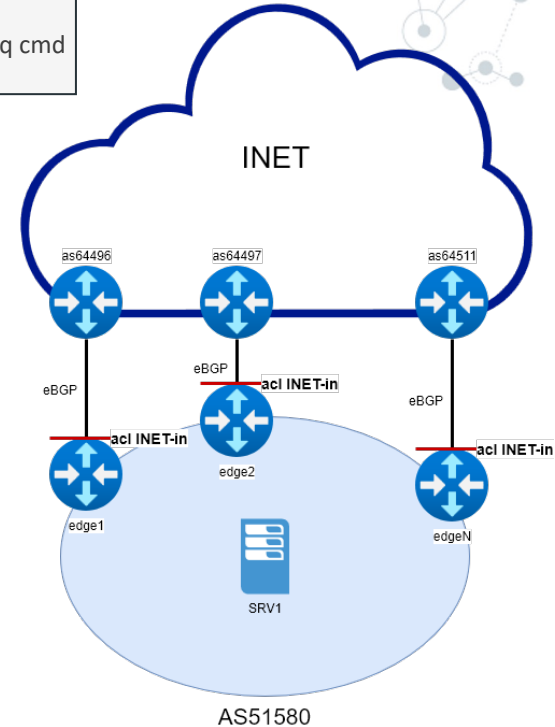
- ◎ Unica ACL allineata tra tutti gli edge
- ◎ Cisco Asr9k
- ◎ Garantire la sicurezza anche durante la modifica
 - Modifica ACL in unico Atomic Update
 - Rete mai aperta
 - TCAM Exhaustion
 - Cisco best practices

```
show access-list INET-in
Tue May 20 14:40:16.243 CET
ipv4 access-list INET-in
 10 permit ipv4 any host 10.0.0.254
 20 permit tcp 10.56.108.0 0.0.0.255 host 10.0.230.210 eq cmd
[...]
```

```
show pfilter-ea fea summary location 0/0/CPU0
Sun May 4 20 14:40:16.243 CET
```

***** NP Resource Usage Summary *****

Chan #	160-bit TCAM Entries	640-bit TCAM Entries	Stats	SS Hash Entries
0	5215	0	69	0
1	5215	0	69	0



Ambiente e Tool utilizzati – Keep it simple



Ubuntu

Focal Fossa
Ambiente di sviluppo



Python

Python3 & pip
Swiss-Army Knife
via apt



Ansible

Automation tool
Via pip per evitare problemi relativi
ad OS update



Jinja2

Manipolazione sintassi
Templating engine
Via pip

NETMiko

Netmiko

Libreria per connettersi ai
dispositivi di rete
Via pip



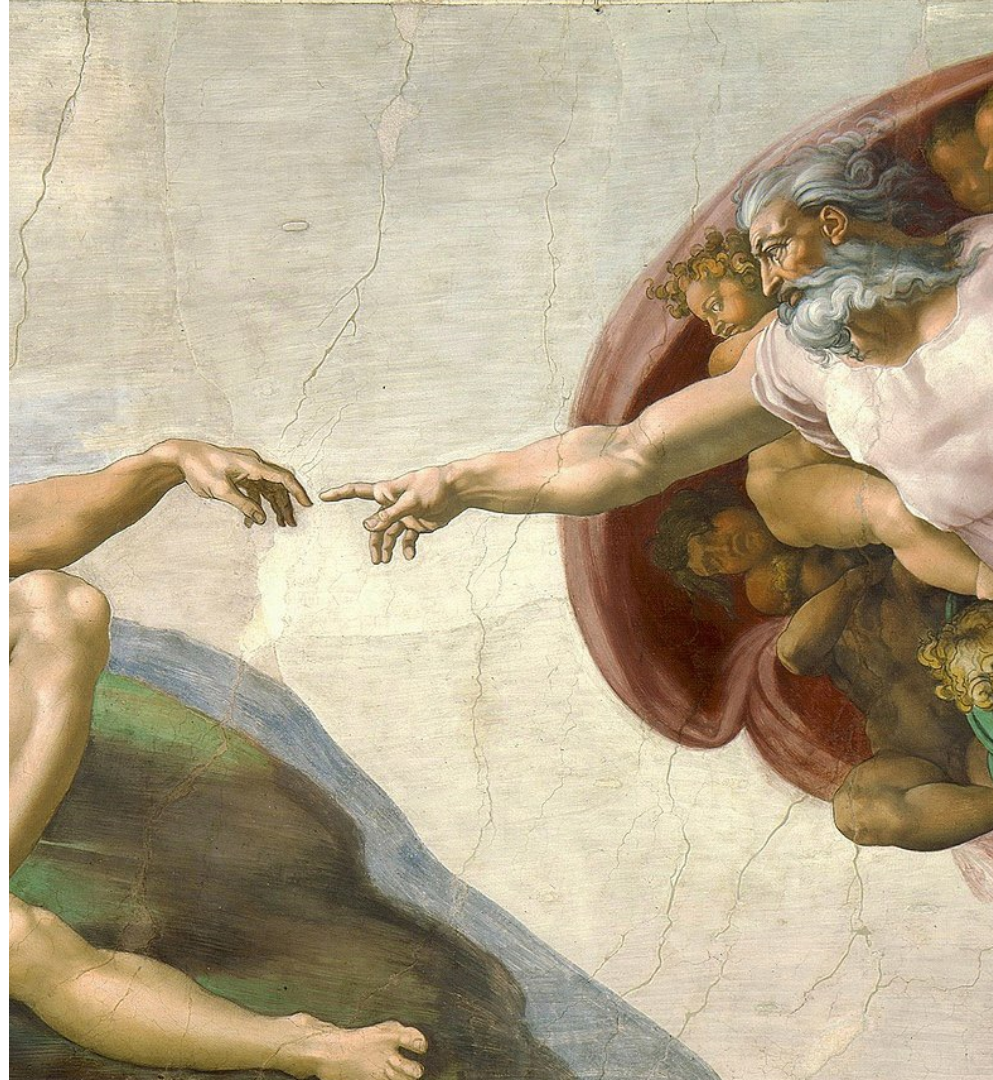
IOS-XR

Network operating system
utilizzato per questo progetto

Setup

Setup Connessione

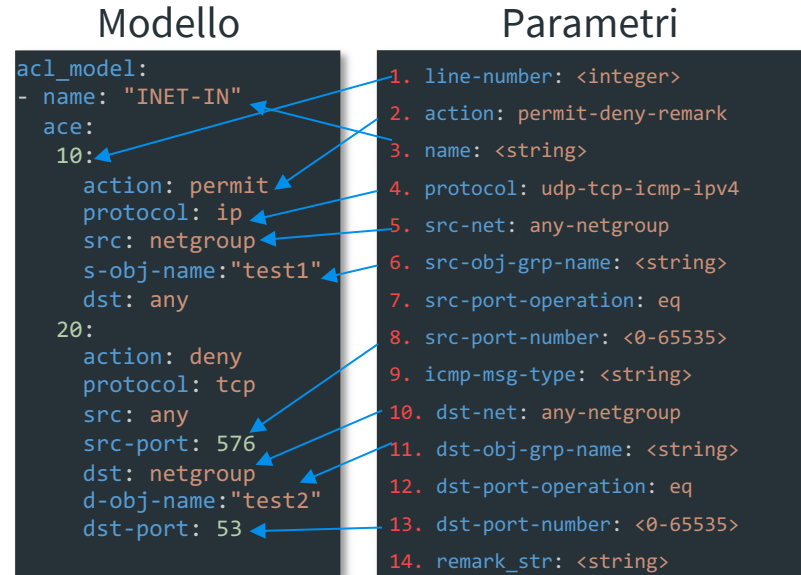
- Abilitare [ssh](#) sui router
- Predisporre credenziali [dedicate](#)
- Strutturare [inventario](#) Ansible
 - INI vs YAML
- [Network_CLI](#) vs NETCONF
 - Uniformità con CLI commands
 - Difficoltà di gestione XML
 - Differenti show proprietari



Il contenitore dei dati

Scelta di un modello

- Scelta del **contenitore** dati
 - Text-Based Data Format
 - Source of Truth strutturata
- Agnostico**
 - No dipendenza da vendor
 - No dipendenza da tool
- YAML** vs JSON
 - Human readability
- Lista** dove ogni ACE è un dizionario
 - ACE: Access List Entry
 - Seq: chiave univoca
- Scelta dei **parametri** del modello



Data collection – parte 1

Popolazione del modello

- ◎ CLI o API
 - Non tutte le API sono supportate
 - Smooth transition
- ◎ Netmiko
 - Libreria Python per SSH
 - Supporto molteplici vendor
- ◎ ConnectHandler
 - Device object per gestire la connessione ssh
 - Chiudere la connessione
- ◎ Send_command
 - Otteniamo come output una stringa python ancora da interpretare



```
#!/usr/bin/python3

from netmiko import ConnectHandler

edge1_prod = {
    "device_type": "cisco_xr",
    "host": "198.51.100.1",
    "username": "my_username",
    "password": "my_password"
}

conn1 = ConnectHandler(**edge1_prod)
acl_raw = conn1.send_command('show ip access-lists')
```

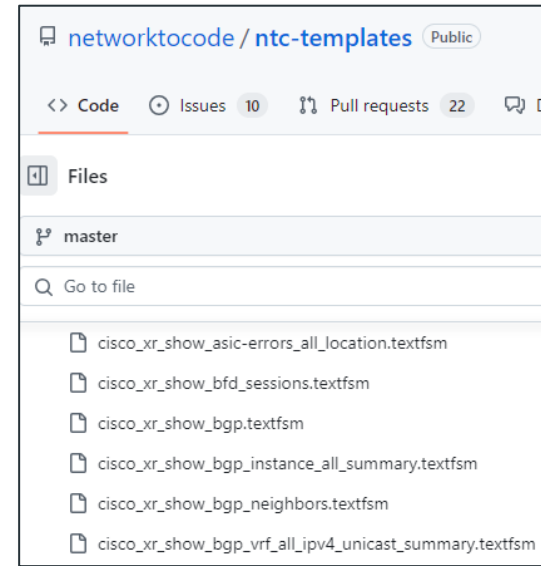
Data collection – parte 2

NTC Templates

- Parsing di una stringa non strutturata
- TextFSM
 - Open Source by Google
 - Conversione testo in dati strutturati
- NTC Template
 - Open Source by networktocode
 - Molteplici vendor supportati
 - Un template per ogni comando
- Utilizzo
 - Dipendenza di Netmiko – No install
 - Template per ACL su XR non esistente
 - Creazione di template locale

Risultato

Lista strutturata con parametri

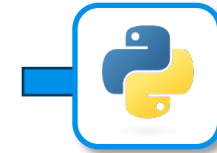


Data collection – parte 3

Arrivo al file modello

- ☉ Dato l'output precedente costruisco il file **YAML**
- ☉ Funzione in **python** che adatta i dati da list
- ☉ **One** time only / Source of Truth

```
[...]  
  
# Source section  
if entry.get("src_any"):  
    ace["src"] = "any"  
elif entry.get("src_network"):  
    ace["src"] = entry["src_network"]  
    if entry.get("src_wildcard"):  
        ace["src-wc"] = entry["src_wildcard"]  
elif entry.get("src_network_object_group_name"):  
    ace["src"] = "netgroup"  
    ace["s-obj-name"] = entry["src_network_object_group_name"]  
  
[...]  
return yaml.dump(result, sort_keys=False)
```



Output file



```
acl_model:  
- name: "INET-IN"  
  ace:  
    10:  
      action: remark  
      remark_str: "DENY-PROTOCOLS"  
    40:  
      action: permit  
      protocol: icmp  
      src: any  
      dst: netgroup  
      d-obj-name: Backbone  
      icmp-type: reassembly-timeout
```

Dal modello al router

Scelta del tool di automazione

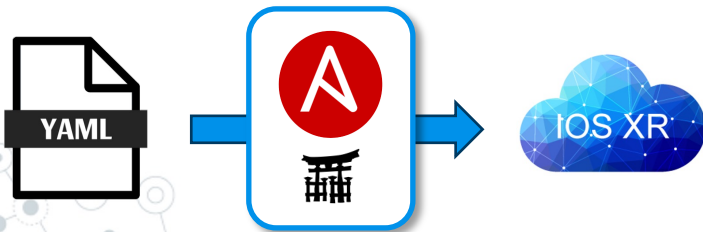
- ◎ Ristretto il campo a 3 possibili tool
 - Semplice comando via ssh tramite python
 - Ansible
 - Nornir & Netmiko
- ◎ Perchè **Ansible**
 - Idempotenza
 - Gestione degli errori
 - Librerie già pronte per ios-xr
 - Minima esperienza personale pregressa
- ◎ Cisco ios xr – **replaced**
 - Modulo ansible per rimpiazzare una acl
 - Rimozione e aggiunta ACE in singolo commit
 - ACL sempre applicate alle interfacce



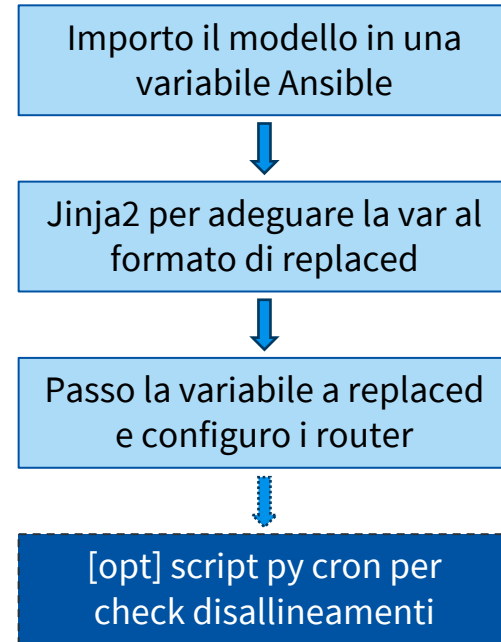
Fase finale

Templating & Deploy

- Necessità di adeguare il modello all'input richiesto dal modulo **replaced**
- **Jinja2**, modulo integrato in Ansible
 - Jinja: linguaggio di templating
 - Jinja2: libreria python per usare jinja
- **Templating**: creazione di un testo parametrizzato con struttura fissa e parti variabili



Ansible playbook



Deploy

```
- name: Load ACL variables from model to IOSxr
hosts: routers
connection: network_cli
gather_facts: no

tasks:
  - name: Importo il file modello nella variabile acl_data
    include_vars:
      file: /root/ansible/production/model.yml
      name: acl_data

  - name: Popolo il dizionario acl_config con le variabili acquisite dal modello
    set_fact:
      acls_config: >-
      [
        "afi": "ipv4"
        "acls": [
          {% for acl in acl_data.acl_vars %}
            [...]

  - name: Replace ACLs config
    cisco.iosxr.iosxr_acls:
      state: replaced
      config: "{{ acls_config }}"
```

Importo il modello in una variabile Ansible

Jinja2 per adeguare la var al formato di replaced

Passo la variabile a replaced e configuro i router

```
eaielli@ubuntu:~/ansible/production# ansible-playbook deploy_edge_ACL.yml -i iosxr_inv.ini
[...]
PLAY RECAP *****
edge1_prod : ok=3 changed=0 unreachable=0 failed=0 skipped=0
rescued=0 ignored=0
```





Considerazioni finali

Production-grade

Progetto nato per produzione, non per lab, e data la criticità delle funzioni coinvolte, deve essere efficiente e sicuro

Vendor Agnostic

L'approccio scelto ed i relativi passaggi sono totalmente indipendenti dal vendor degli apparati di rete e dai tool utilizzati

Scelta dei tool

Scelta voluta per imprimere fondamenta solide volte a comprendere le basi fondamentali della network automation.

“One machine can do the work of fifty ordinary men. No machine can do the work of one extraordinary man.” – Elbert Hubbard

Sviluppi futuri

Versioning

Aggiungere la repository su git per mantenere il versioning di tutto, SoT compresa. Possibile CI/CD

Object group check

Verificare l'allineamento anche degli object group ed eventualmente tenerli aggiornati ove presenti API online

Source of Truth

Valutare se inserire i dati contenuti attualmente nel modello, in un SoT più strutturato

Public NTC Template

A seguito test approfonditi pubblicare sul git ufficiale il template che ho costruito per IOS-XR ACL

Risorse

Link e risorse utili

<https://www.patreon.com/c/adainese/posts>

Network Programmability and Automation,
2nd Edition

Domande?



Contatti

elia.aielli@timenet.com

[Linkedin.com/in/elia-aielli](https://www.linkedin.com/in/elia-aielli)

