Bridging the Automation Gap: Real Cases with Ansible

Engineering adaptive intelligence

dvloper.io

ABOUT US

Who We Are: A global system integrator headquartered in Bucharest, Romania, with a team of 100+ skilled Product Engineers & Software Developers.

Our Pillars:

- Network Transformation
- Cloud Services
- Applications.
- Data & AI

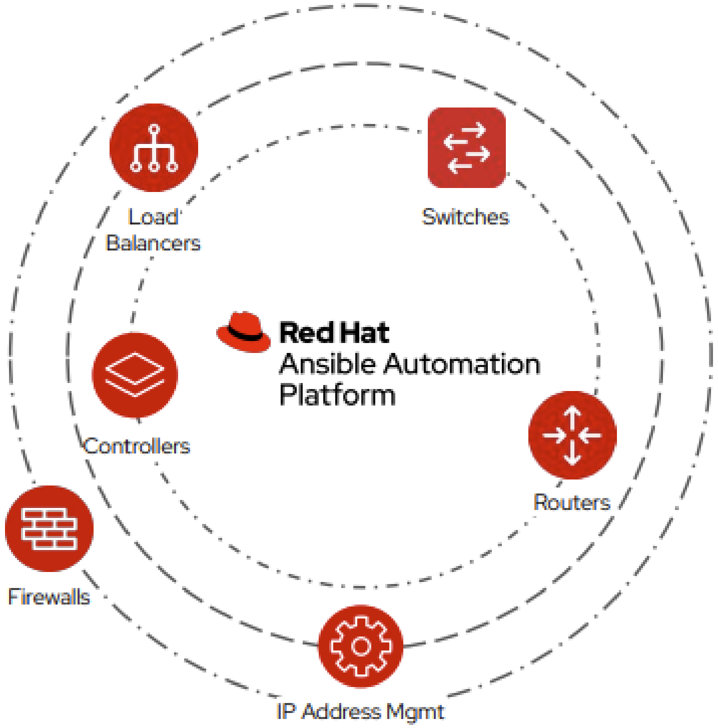
Our Reach: Trusted partnerships with customers worldwide, emphasizing integration, automation, and analysis of IaaS, PaaS, and SaaS platforms.

dvloper.io

Agenda

- Ansible Network Automation
- Use Case: Configuration Management
    - Client infrastructure
    - Complexities faced by the client
    - Goals

Ansible Network Automation



Configuration Management

Platform agnostic configuration management to standardize and enforce best-practices.

Infrastructure Awareness

Track network resources through facts gathering, to perform preventive maintenance, reducing outage risks and costs of unnecessary hardware-refresh.

Network Validation

Examine operational state to check network connectivity and protocols and to enhance operational workflows to help measure network intent.

dvloper.io

Use Case: Configuration Management

Client infrastructure:

- Thousands of network devices
    - Hub and spoke network model
    - Branches: Routers, Firewalls, Branch Switches
    - Data Centers: Routers, Firewalls, Datacenter Switches
    - Aggregators

- Different OSs from multiple vendors

Complexities faced by client:

- Scalability Issues
    - Resource Intensiveness
- Human Error
    - Configuration Discrepancies
    - Time-Consuming Updates
- Operational Downtime
    - Service Disruptions
- Lack of Centralized Control
    - Limited Visibility
    - Difficulty in Tracking Changes

Use Case: Configuration Management

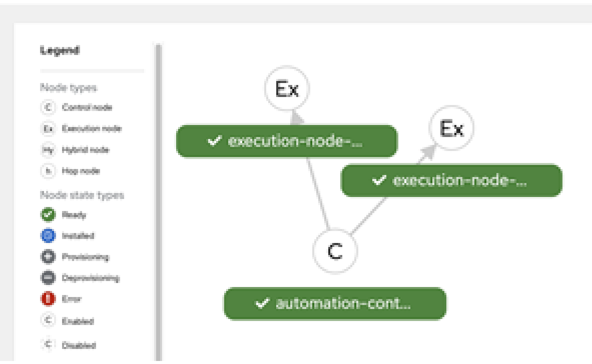
Goals:

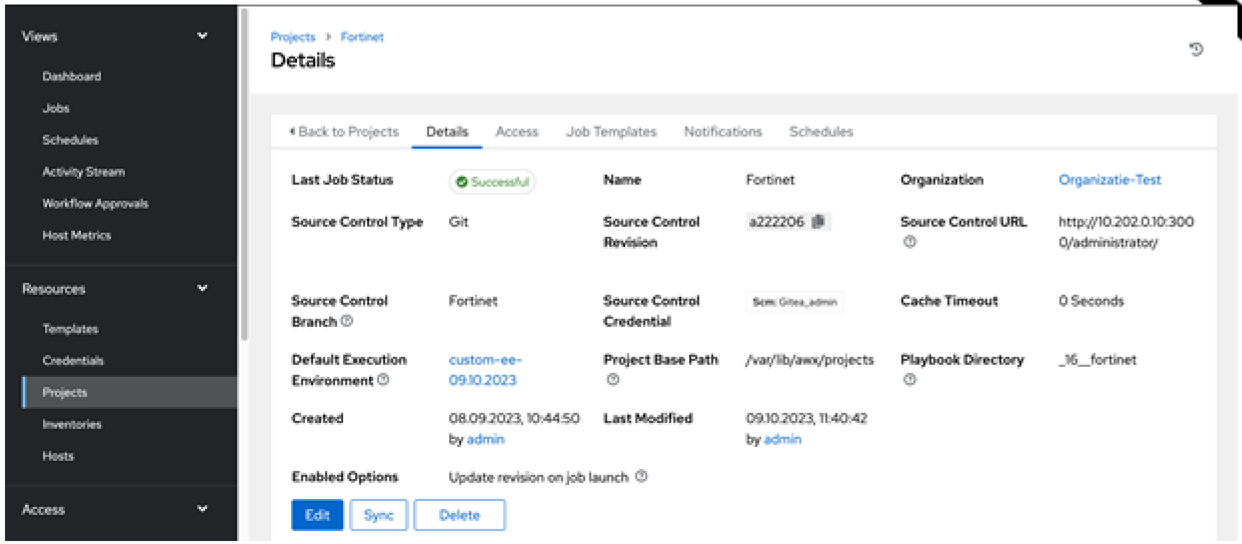
- Red Hat Ansible Automation Platform
  - Install in air gapped environment
  - Configure LDAP integration
  - Configure integration with Git Repo

Challenge:

- Air gapped environment installation

Topology View





Details

Last Job Status: Successful

Name: Fortinet

Organization: Organizatie-Test

Source Control Type: Git

Source Control Revision: a222206

Source Control URL: http://10.202.0.10:3000/administrator/

Source Control Branch: Fortinet

Source Control Credential: Scm: Gitea_admin

Cache Timeout: 0 Seconds

Default Execution Environment: custom-ee-09.10.2023

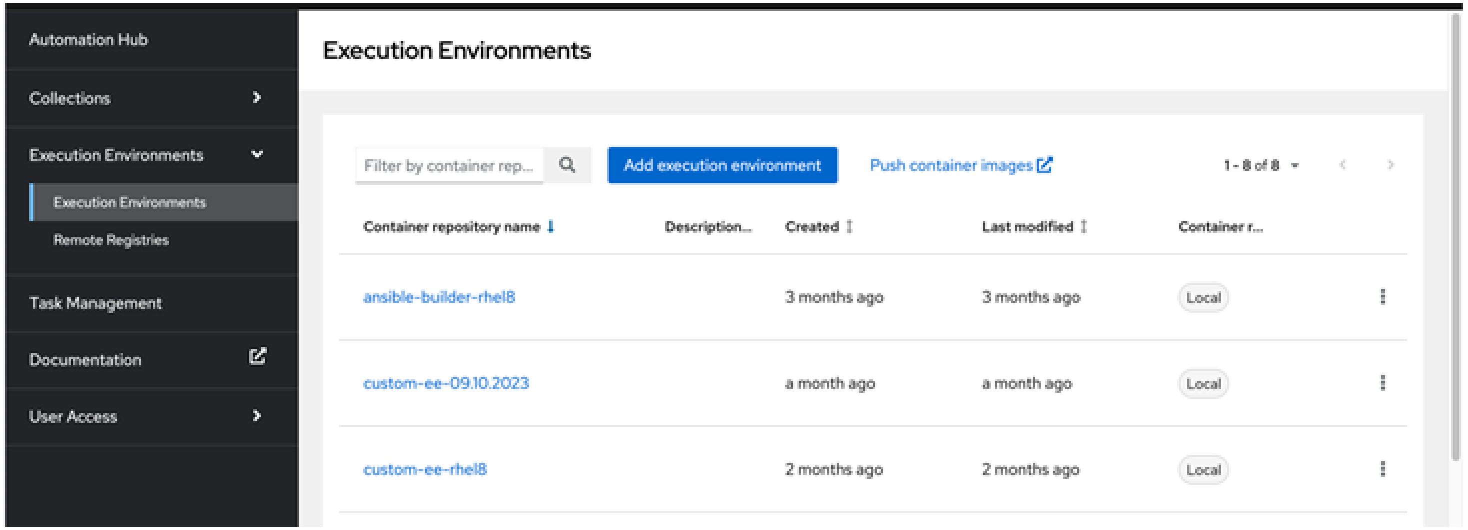
Project Base Path: /var/lib/awx/projects

Playbook Directory: _16__fortinet

Created: 08.09.2023, 10:44:50 by admin

Last Modified: 09.10.2023, 11:40:42 by admin

Enabled Options: Update revision on job launch



Execution Environments

Container repository name	Description...	Created	Last modified	Container r...
ansible-builder-rhel8		3 months ago	3 months ago	Local
custom-ee-09.10.2023		a month ago	a month ago	Local
custom-ee-rhel8		2 months ago	2 months ago	Local

dvloper.io

Use Case: Configuration Management

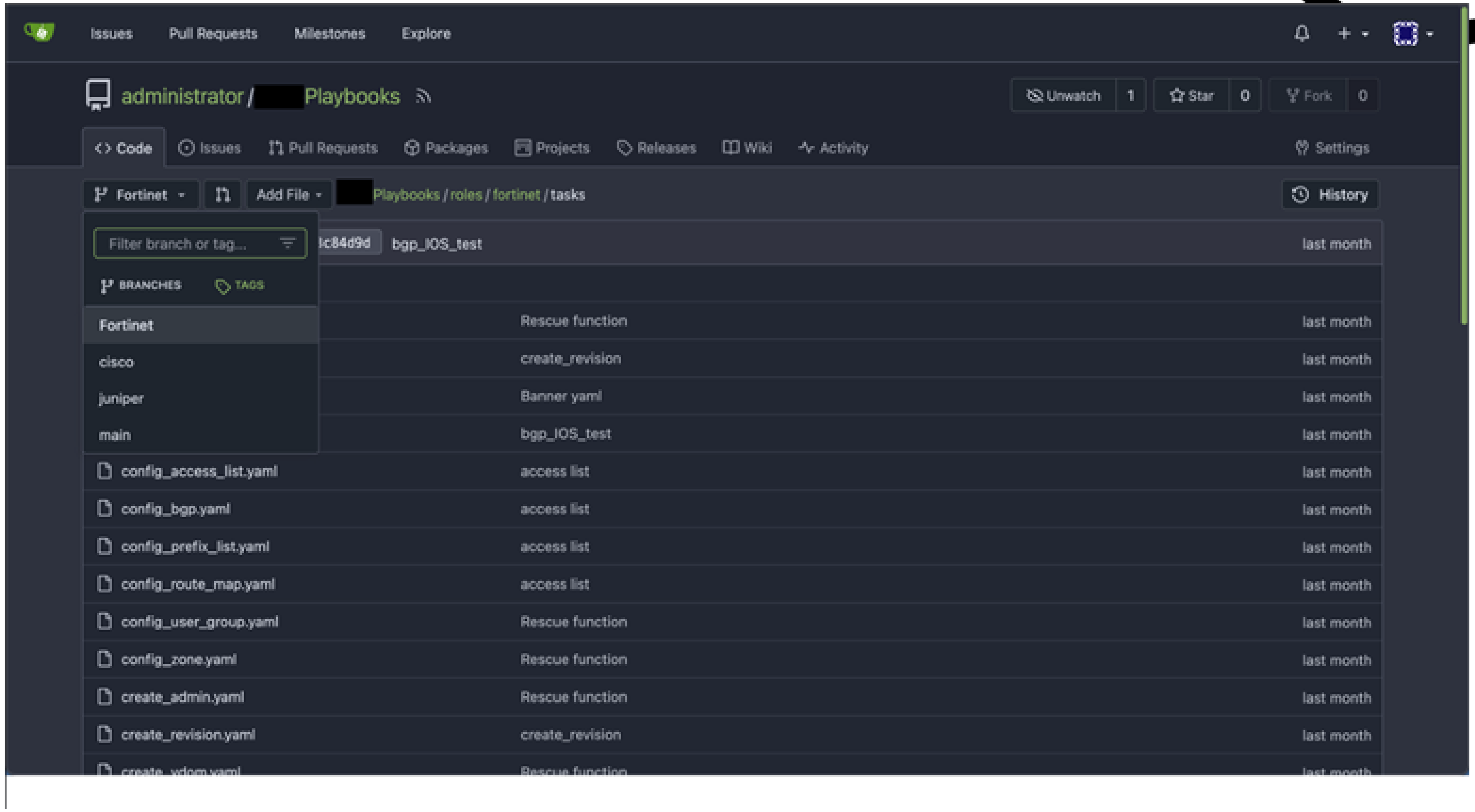
Goals:

- Git repository tool:
  - Install Git repository software

Benefits:

- Version Control and History Tracking:
  - Code Consistency
  - Traceability

- Collaboration and Teamwork:
  - Concurrent Development
  - Code Review and Approval



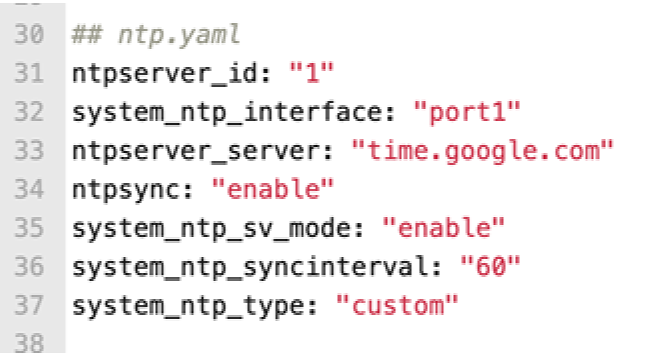
[redacted] Playbooks / roles / fortinet / tasks

c84d9d	bgp_IOS_test	last month
	Rescue function	last month
	create_revision	last month
	Banner yaml	last month
	bgp_IOS_test	last month
config_access_list.yaml	access list	last month
config_bgp.yaml	access list	last month
config_prefix_list.yaml	access list	last month
config_route_map.yaml	access list	last month
config_user_group.yaml	Rescue function	last month
config_zone.yaml	Rescue function	last month
create_admin.yaml	Rescue function	last month
create_revision.yaml	create_revision	last month
create_vdom.yaml	Rescue function	last month

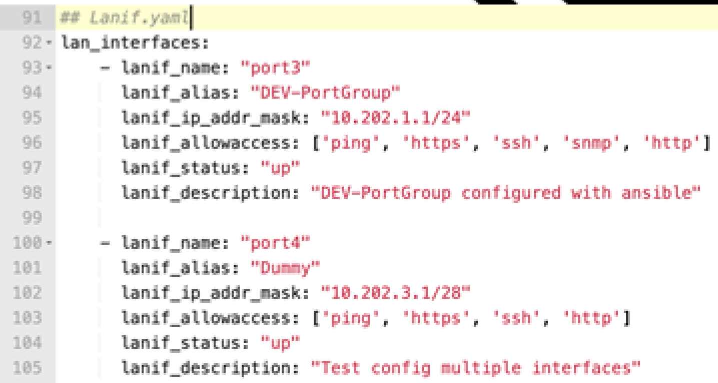
Use Case: Configuration Management

Goals:

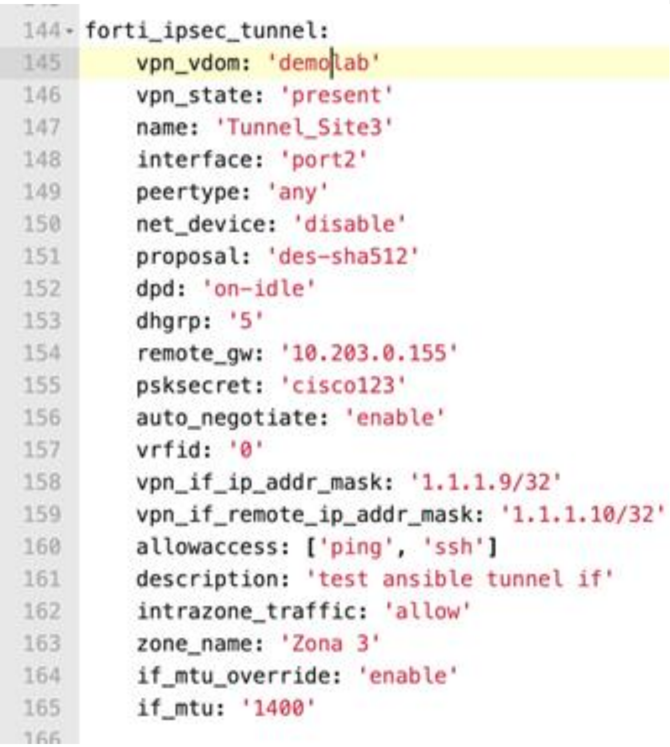
- Create Network automation playbooks:
  - Initial Configuration: Hostname, SSH, NTP, TACACS, SNMP, DNS, Banner, etc.
  - VLAN and VRF configuration
  - Interface Configuration
  - Static and Dynamic Routing protocols configuration: Static routes/ BGP/OSPF
  - Site-to-site IPsec VPN tunnels
  - ACLs / Firewall policies
  - Zone configuration
  - Automated Roll Back



```
## ntp.yaml
ntpserver_id: "1"
system_ntp_interface: "port1"
ntpserver_server: "time.google.com"
ntpsync: "enable"
system_ntp_sv_mode: "enable"
system_ntp_syncinterval: "60"
system_ntp_type: "custom"

```



```
## Lanif.yaml
lan_interfaces:
    - lanif_name: "port3"
      lanif_alias: "DEV-PortGroup"
      lanif_ip_addr_mask: "10.202.1.1/24"
      lanif_allowaccess: ['ping', 'https', 'ssh', 'snmp', 'http']
      lanif_status: "up"
      lanif_description: "DEV-PortGroup configured with ansible"

    - lanif_name: "port4"
      lanif_alias: "Dummy"
      lanif_ip_addr_mask: "10.202.3.1/28"
      lanif_allowaccess: ['ping', 'https', 'ssh', 'http']
      lanif_status: "up"
      lanif_description: "Test config multiple interfaces"
```



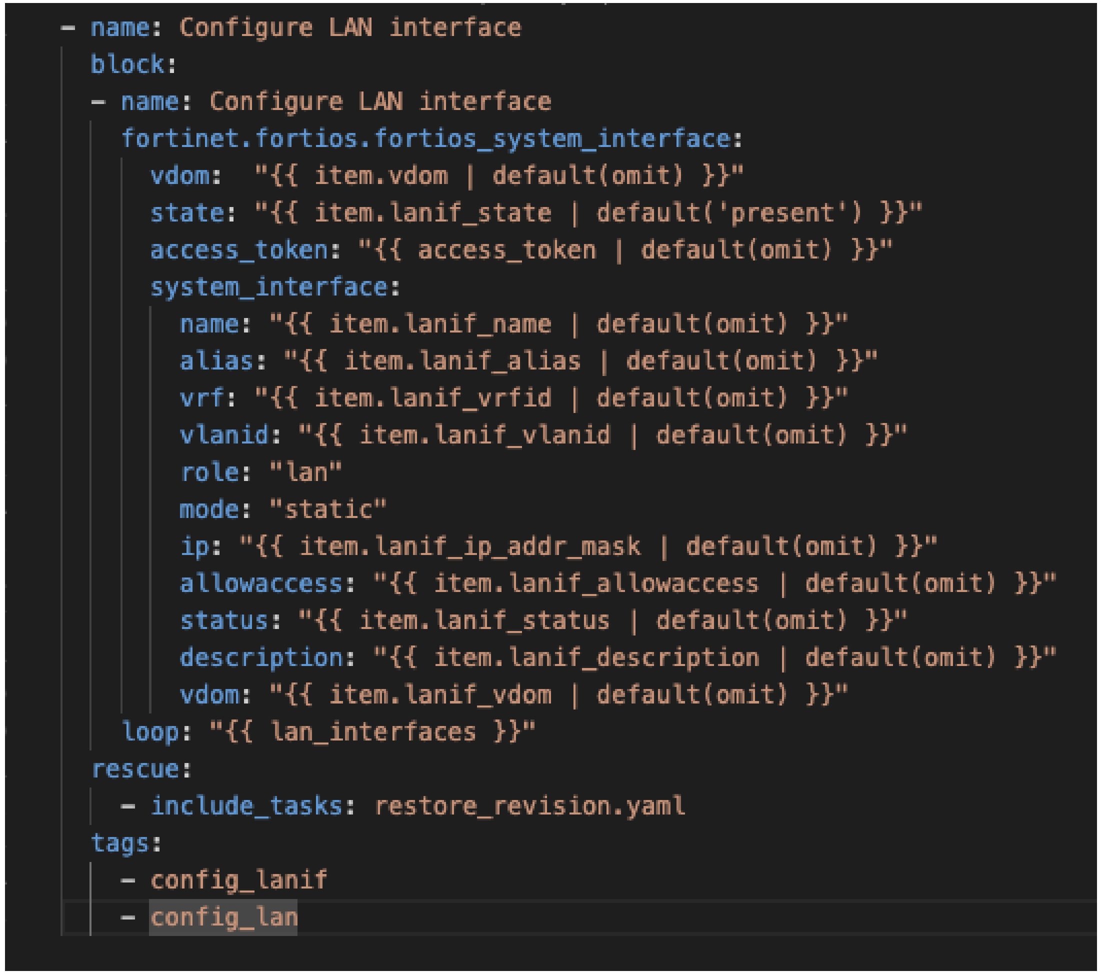
```
forti_ipsec_tunnel:
    vpn_vdom: 'demolab'
    vpn_state: 'present'
    name: 'Tunnel_Site3'
    interface: 'port2'
    peertype: 'any'
    net_device: 'disable'
    proposal: 'des-sha512'
    dpd: 'on-idle'
    dhgrp: '5'
    remote_gw: '10.203.0.155'
    psksecret: 'cisco123'
    auto_negotiate: 'enable'
    vrfid: '0'
    vpn_if_ip_addr_mask: '1.1.1.9/32'
    vpn_if_remote_ip_addr_mask: '1.1.1.10/32'
    allowaccess: ['ping', 'ssh']
    description: 'test ansible tunnel if'
    intrazone_traffic: 'allow'
    zone_name: 'Zona 3'
    if_mtu_override: 'enable'
    if_mtu: '1400'

```

dvloper.io

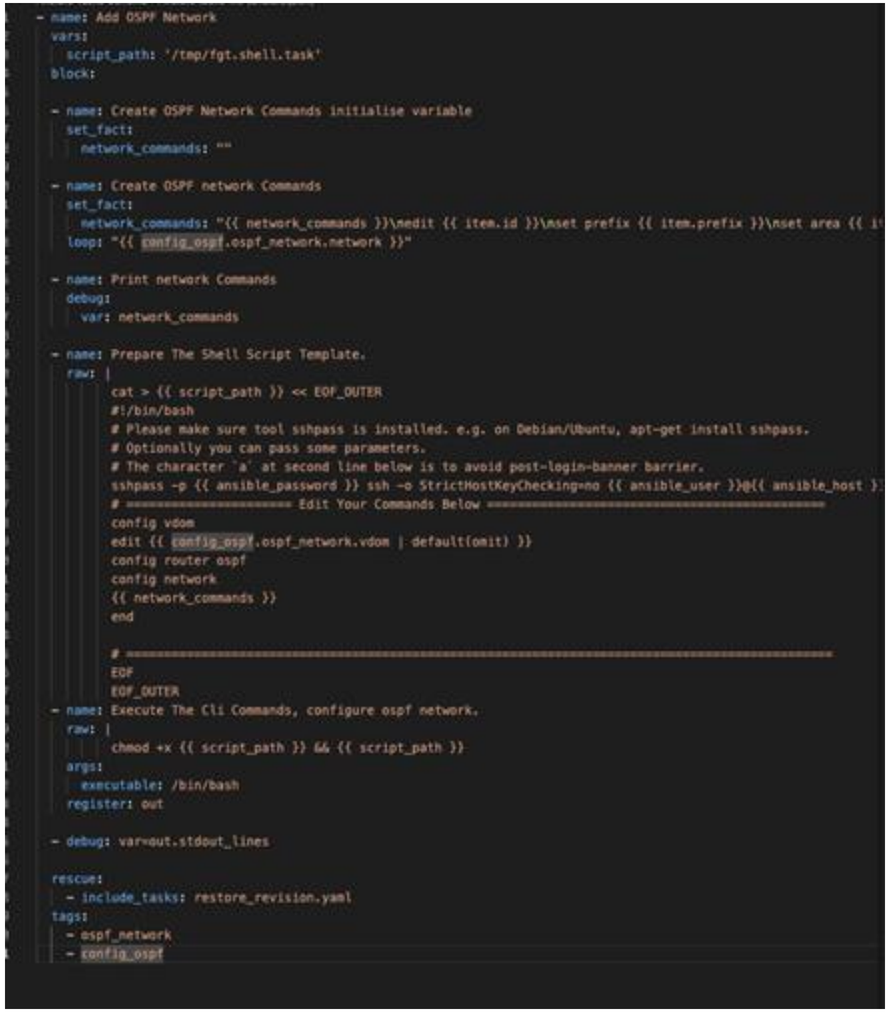
Thank you!

dvloper.io

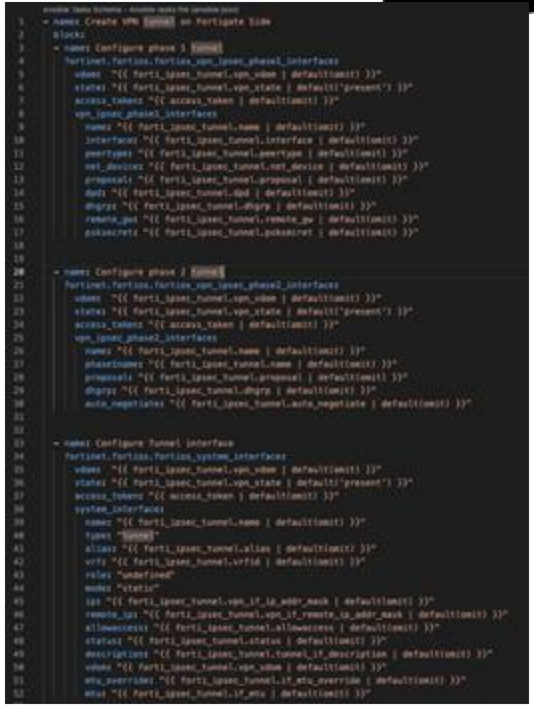
Use Case: Configuration Management



```yaml
- name: Configure LAN interface
  block:
  - name: Configure LAN interface
    fortinet.fortios.fortios_system_interface:
      vdom:  "{{ item.vdom | default(omit) }}"
      state: "{{ item.lanif_state | default('present') }}"
      access_token: "{{ access_token | default(omit) }}"
      system_interface:
        name: "{{ item.lanif_name | default(omit) }}"
        alias: "{{ item.lanif_alias | default(omit) }}"
        vrf: "{{ item.lanif_vrfid | default(omit) }}"
        vlanid: "{{ item.lanif_vlanid | default(omit) }}"
        role: "lan"
        mode: "static"
        ip: "{{ item.lanif_ip_addr_mask | default(omit) }}"
        allowaccess: "{{ item.lanif_allowaccess | default(omit) }}"
        status: "{{ item.lanif_status | default(omit) }}"
        description: "{{ item.lanif_description | default(omit) }}"
        vdom: "{{ item.lanif_vdom | default(omit) }}"
    loop: "{{ lan_interfaces }}"
  rescue:
    - include_tasks: restore_revision.yaml
  tags:
    - config_lanif
    - config_lan
```



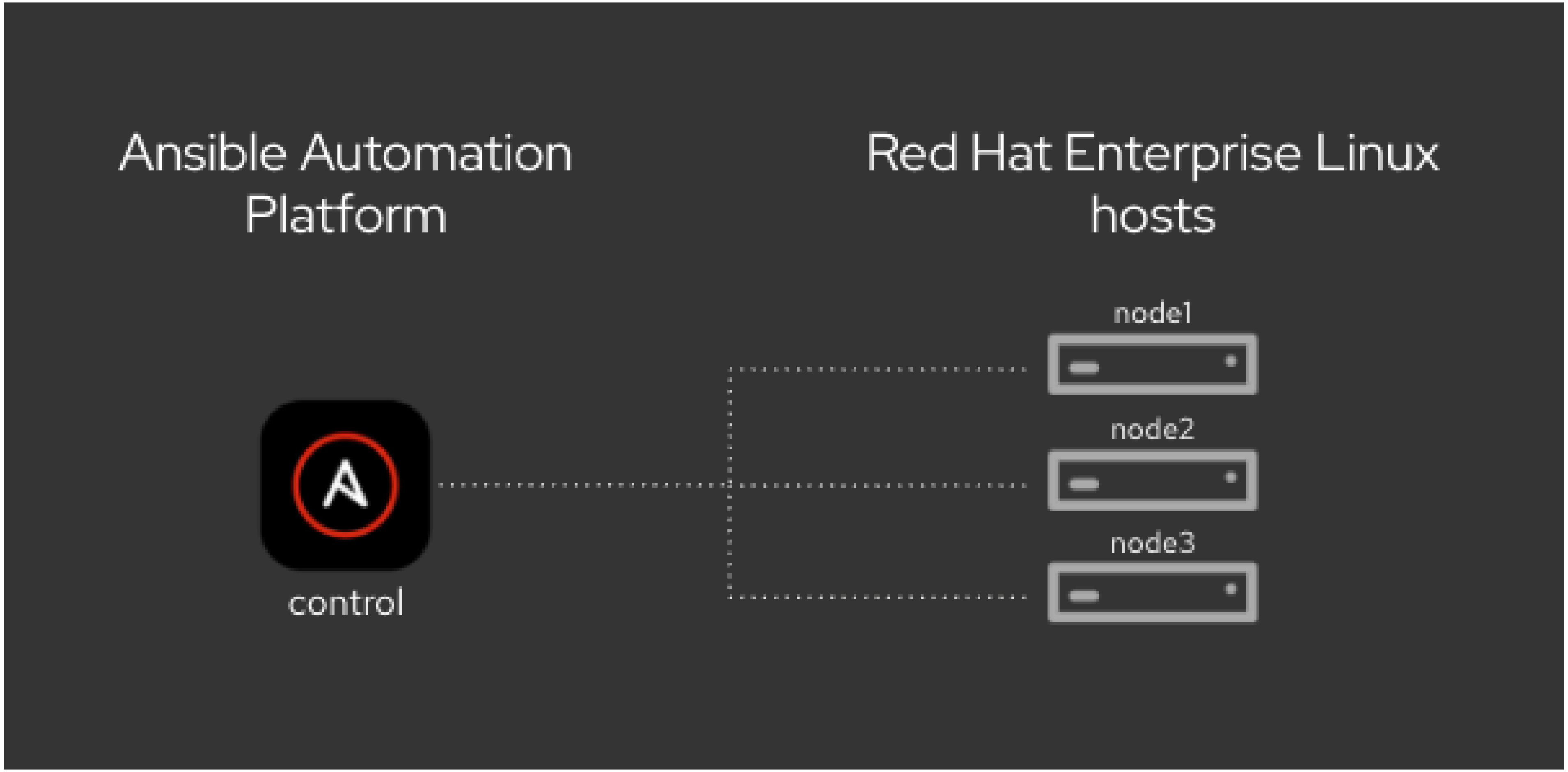
```yaml
- name: Add OSPF Network
  vars:
    script_path: '/tmp/fgt.shell.task'
  block:

  - name: Create OSPF Network Commands initialise variable
    set_fact:
      network_commands: ""

  - name: Create OSPF network Commands
    set_fact:
      network_commands: "{{ network_commands }}\nedit {{ item.id }}\nset prefix {{ item.prefix }}\nset area {{ i
    loop: "{{ config_ospf.ospf_network.network }}"

  - name: Print network Commands
    debug:
      var: network_commands

  - name: Prepare The Shell Script Template.
    raw: |
          cat > {{ script_path }} << EOF_OUTER
          #!/bin/bash
          # Please make sure tool sshpass is installed. e.g. on Debian/Ubuntu, apt-get install sshpass.
          # Optionally you can pass some parameters.
          # The character `a` at second line below is to avoid post-login-banner barrier.
          sshpass -p {{ ansible_password }} ssh -o StrictHostKeyChecking=no {{ ansible_user }}@{{ ansible_host }
          # ====================== Edit Your Commands Below ==============================================
          config vdom
          edit {{ config_ospf.ospf_network.vdom | default(omit) }}
          config router ospf
          config network
          {{ network_commands }}
          end

          # =============================================================================================
          EOF
          EOF_OUTER
  - name: Execute The Cli Commands, configure ospf network.
    raw: |
          chmod +x {{ script_path }} && {{ script_path }}
    args:
      executable: /bin/bash
    register: out

  - debug: var=out.stdout_lines

  rescue:
    - include_tasks: restore_revision.yaml
  tags:
    - ospf_network
    - config_ospf
```



Ansible Tasks Schema - Ansible tasks file (ansible.json)

```yaml
- name: Create VPN tunnel on Fortigate Side
  block:
  - name: Configure phase 1 tunnel
    fortinet.fortios.fortios_vpn_ipsec_phase1_interface:
      vdom:  "{{ forti_ipsec_tunnel.vpn_vdom | default(omit) }}"
      state: "{{ forti_ipsec_tunnel.vpn_state | default('present') }}"
      access_token: "{{ access_token | default(omit) }}"
      vpn_ipsec_phase1_interface:
        name: "{{ forti_ipsec_tunnel.name | default(omit) }}"
        interface: "{{ forti_ipsec_tunnel.interface | default(omit) }}"
        peertype: "{{ forti_ipsec_tunnel.peertype | default(omit) }}"
        net_device: "{{ forti_ipsec_tunnel.net_device | default(omit) }}"
        proposal: "{{ forti_ipsec_tunnel.proposal | default(omit) }}"
        dpd: "{{ forti_ipsec_tunnel.dpd | default(omit) }}"
        dhgrp: "{{ forti_ipsec_tunnel.dhgrp | default(omit) }}"
        remote_gw: "{{ forti_ipsec_tunnel.remote_gw | default(omit) }}"
        psksecret: "{{ forti_ipsec_tunnel.psksecret | default(omit) }}"


  - name: Configure phase 2 tunnel
    fortinet.fortios.fortios_vpn_ipsec_phase2_interface:
      vdom:  "{{ forti_ipsec_tunnel.vpn_vdom | default(omit) }}"
      state: "{{ forti_ipsec_tunnel.vpn_state | default('present') }}"
      access_token: "{{ access_token | default(omit) }}"
      vpn_ipsec_phase2_interface:
        name: "{{ forti_ipsec_tunnel.name | default(omit) }}"
        phase1name: "{{ forti_ipsec_tunnel.name | default(omit) }}"
        proposal: "{{ forti_ipsec_tunnel.proposal | default(omit) }}"
        dhgrp: "{{ forti_ipsec_tunnel.dhgrp | default(omit) }}"
        auto_negotiate: "{{ forti_ipsec_tunnel.auto_negotiate | default(omit) }}"


  - name: Configure tunnel interface
    fortinet.fortios.fortios_system_interface:
      vdom:  "{{ forti_ipsec_tunnel.vpn_vdom | default(omit) }}"
      state: "{{ forti_ipsec_tunnel.vpn_state | default('present') }}"
      access_token: "{{ access_token | default(omit) }}"
      system_interface:
        name: "{{ forti_ipsec_tunnel.name | default(omit) }}"
        type: "tunnel"
        alias: "{{ forti_ipsec_tunnel.alias | default(omit) }}"
        vrf: "{{ forti_ipsec_tunnel.vrfid | default(omit) }}"
        role: "undefined"
        mode: "static"
        ip: "{{ forti_ipsec_tunnel.vpn_if_ip_addr_mask | default(omit) }}"
        remote_ip: "{{ forti_ipsec_tunnel.vpn_if_remote_ip_addr_mask | default(omit) }}"
        allowaccess: "{{ forti_ipsec_tunnel.allowaccess | default(omit) }}"
        status: "{{ forti_ipsec_tunnel.status | default(omit) }}"
        description: "{{ forti_ipsec_tunnel.tunnel_if_description | default(omit) }}"
        vdom: "{{ forti_ipsec_tunnel.vpn_vdom | default(omit) }}"
        mtu_override: "{{ forti_ipsec_tunnel.if_mtu_override | default(omit) }}"
        mtu: "{{ forti_ipsec_tunnel.if_mtu | default(omit) }}"
```

Ansible Server Automation



dvloper.io