YAML, Jinja2 and Ansible DMVPN Case Study

Ivan Pepelnjak (ip@ipSpace.net)
Network Architect

ipSpace.net AG

The Goal

Automated Large-Scale DMVPN Deployment

Requirements

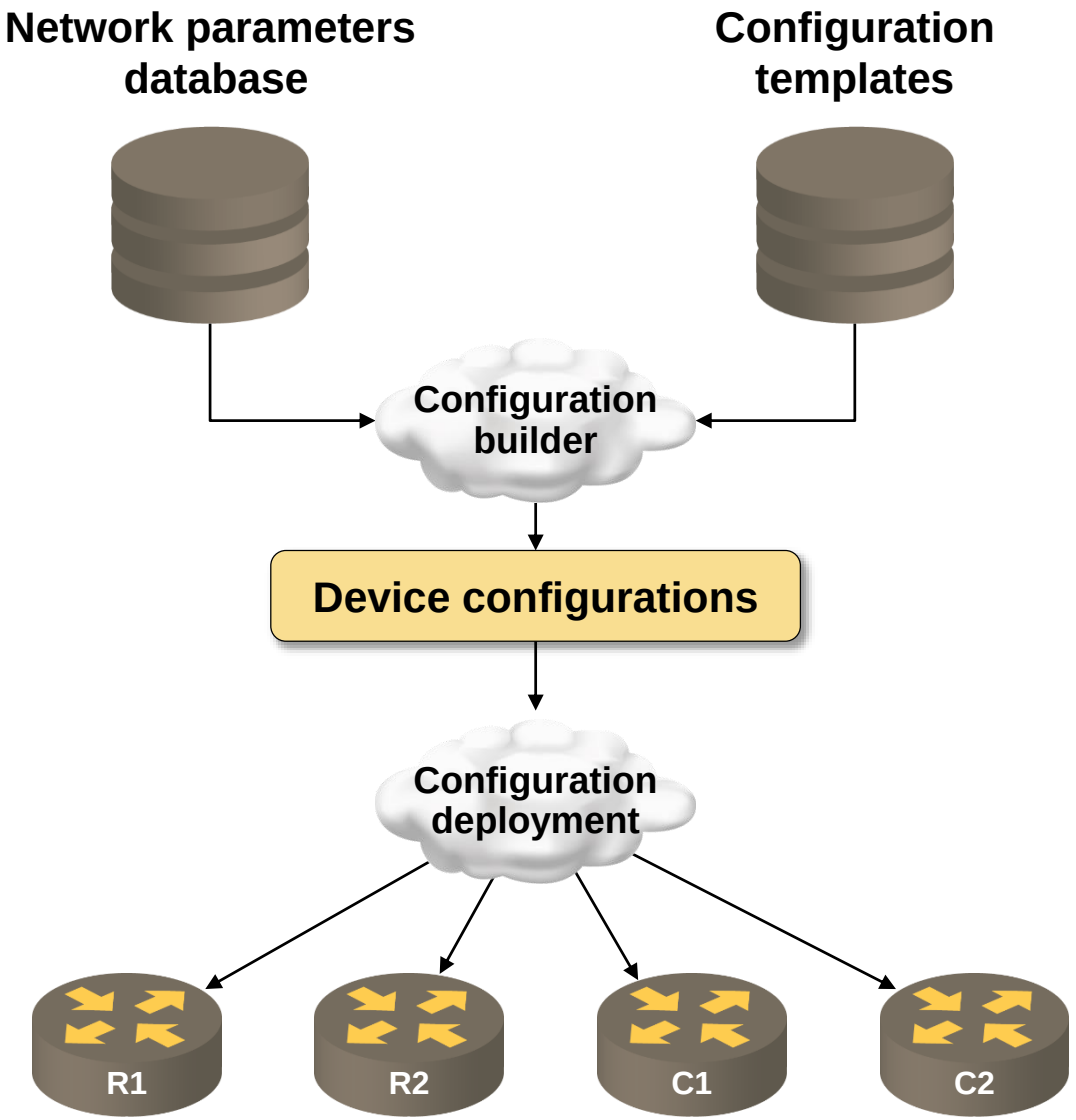
Large scale DMVPN deployment

- Multiple hub routers
- Thousands of remote sites
- Remote sites with one or two routers

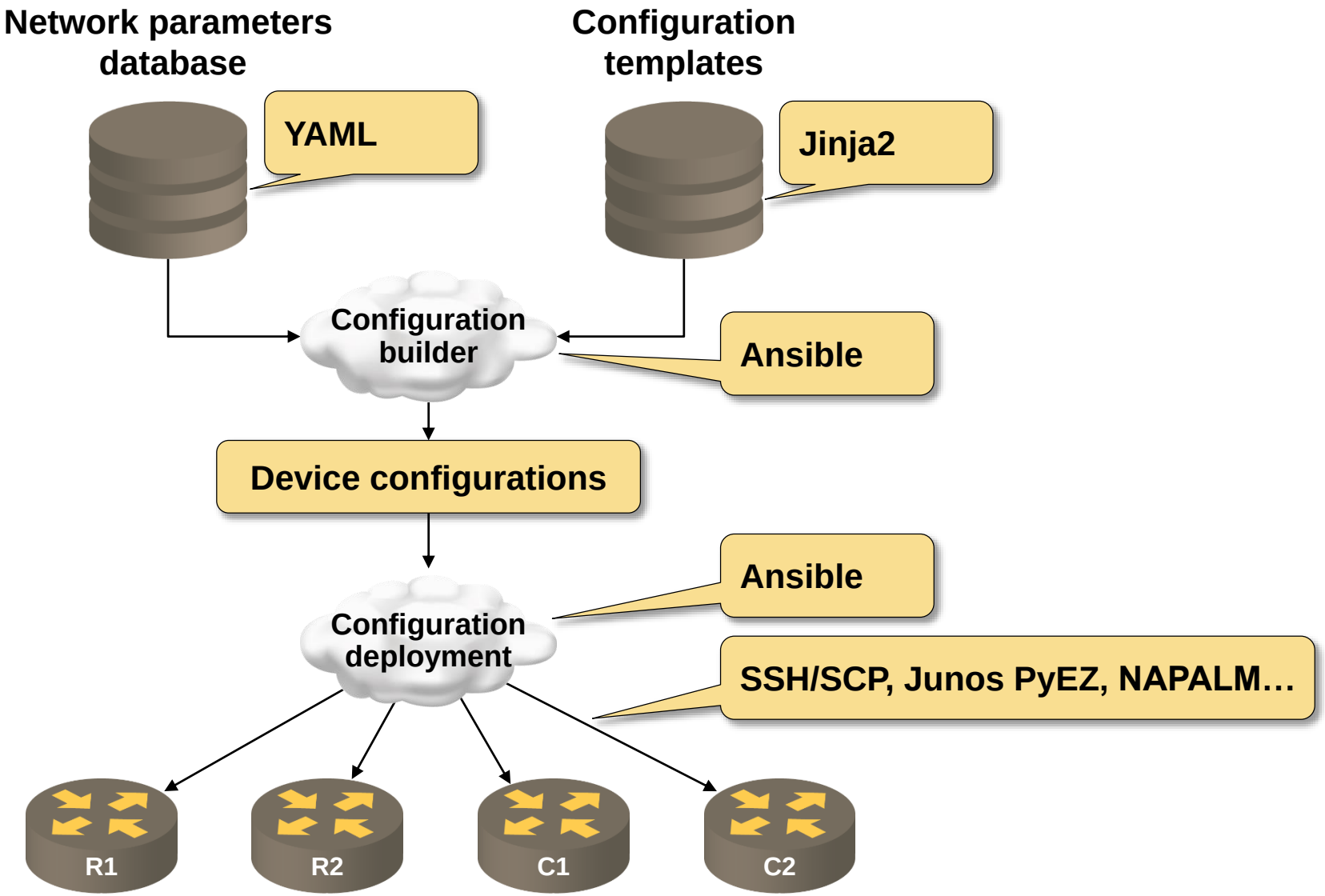
Fully automated provisioning

- Minimal number of manual operations per site
- Router configurations are generated by automation tools
- Configurations (or configuration changes) are deployed automatically

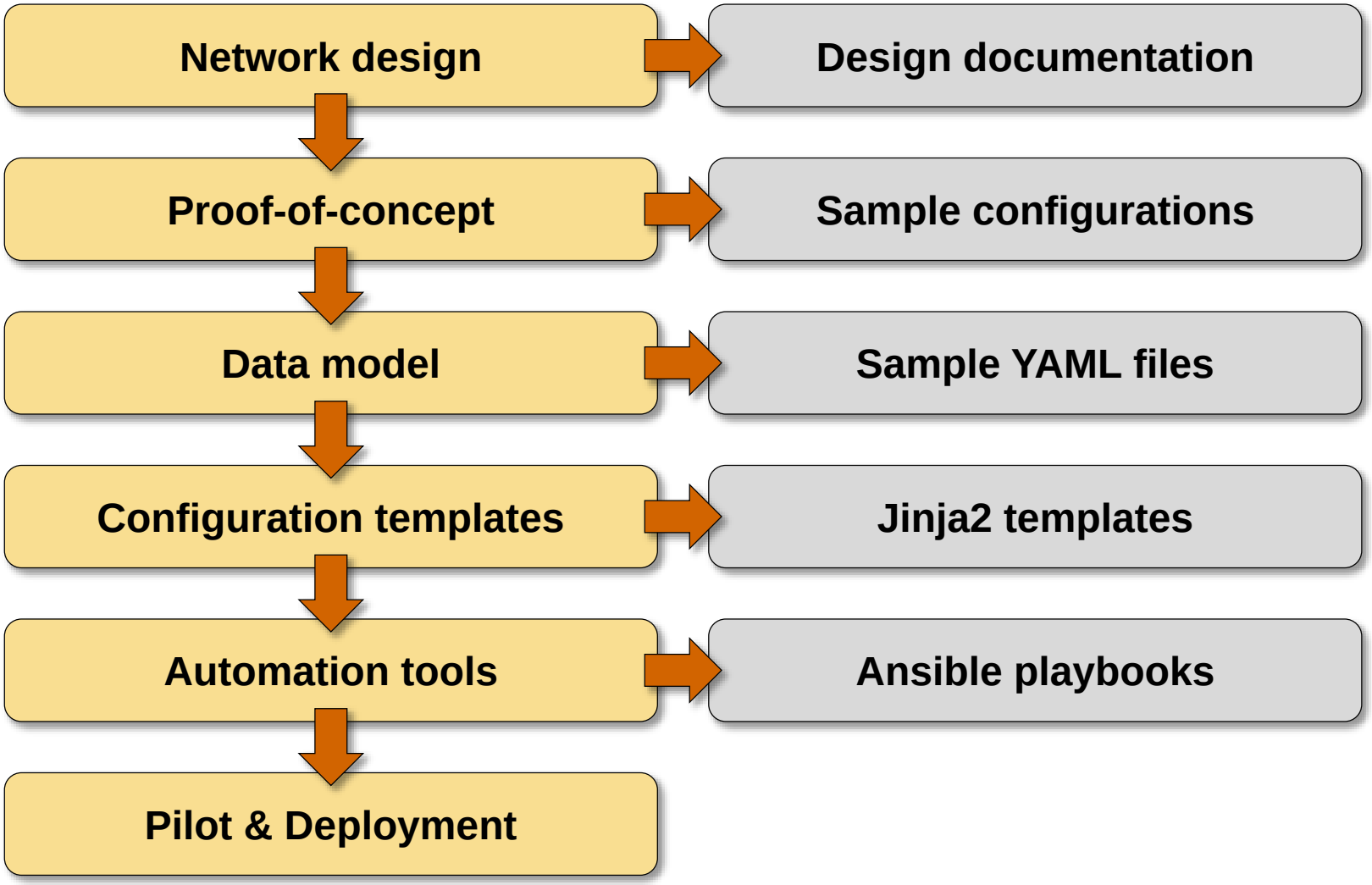
The Goal



Implementing Configuration Builder with Ansible

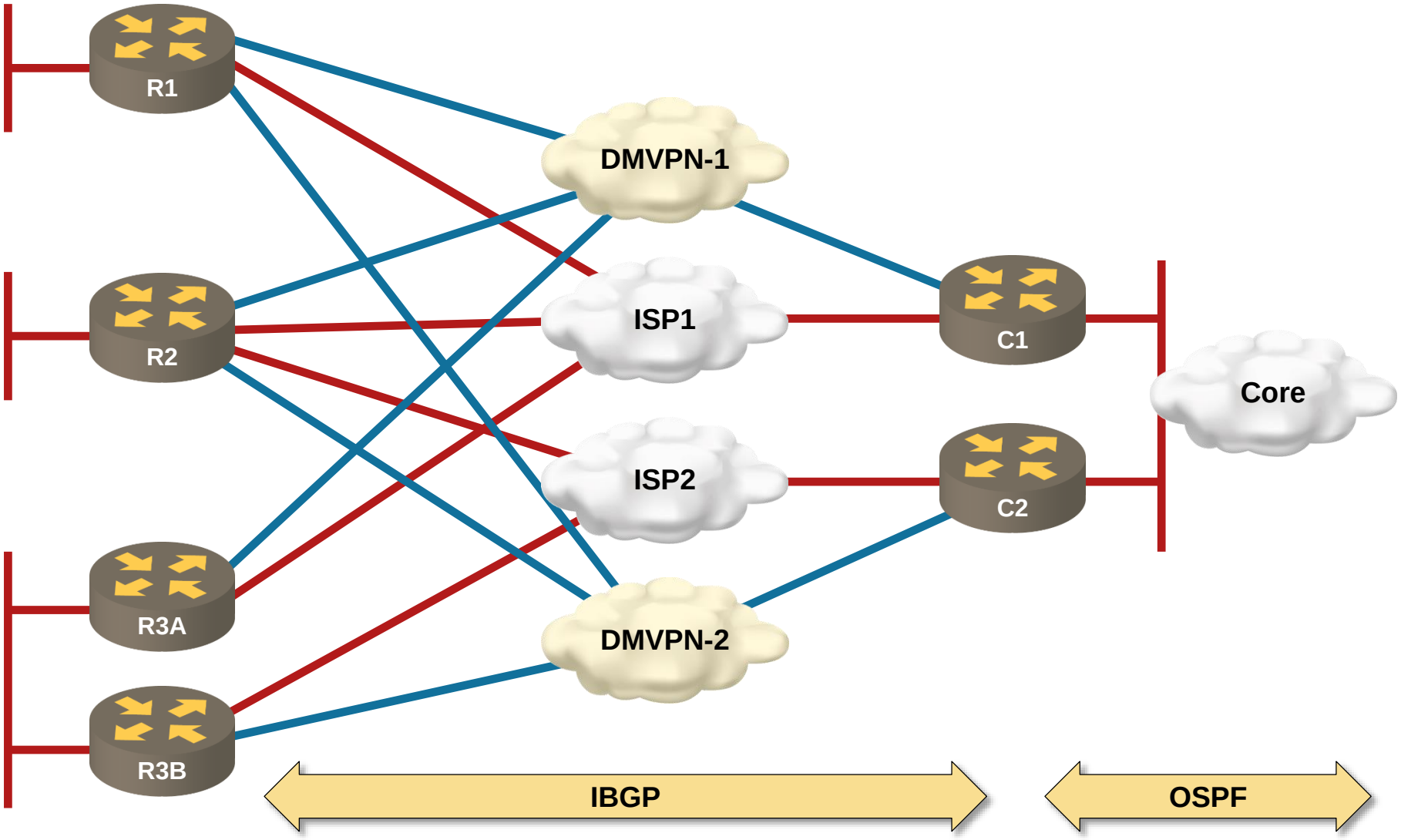


Getting There



Network Design

Network Design

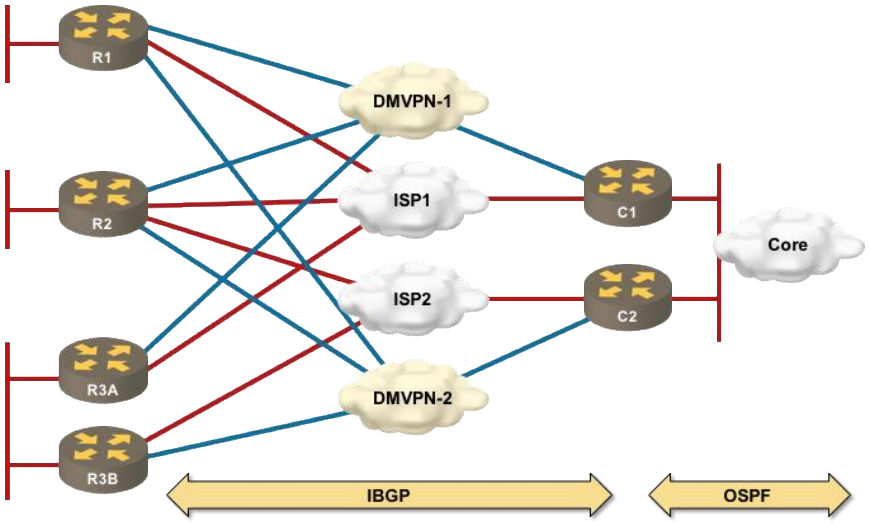


Network Design Overview

- Single hub site
- Phase-3 DMVPN
- Two separate DMVPN tunnels
- Two hub routers (one per tunnel)
- Preshared IPsec keys
- IBGP over DMVPN, OSPF toward the network core

BGP design

- Single AS
- BGP sessions established between DMVPN interface addresses
- Hub routers are BGP RR



Watch the DMVPN webinars for more details

Proof-of-Concept Configurations

IPsec Configuration

```
crypto keyring DMVPN
  pre-shared-key address 0.0.0.0 0.0.0.0 key TESTING
!
crypto isakmp policy 10
 authentication pre-share
 group 2
!
!
crypto ipsec transform-set DMVPN esp-des esp-sha-hmac
 mode transport
!
crypto ipsec profile DMVPN
 set transform-set DMVPN
```

Hub Router DMVPN Tunnel

```
interface Tunnel0
 bandwidth 10000
 ip address 192.168.0.1 255.255.255.0
 no ip redirects
 ip mtu 1400
 ip nhrp authentication WanExamp
 ip nhrp map multicast dynamic
 ip nhrp network-id 12345
 ip tcp adjust-mss 1360
 tunnel source Serial1/0
 tunnel mode gre multipoint
 tunnel key 12345
 tunnel protection ipsec profile DMVPN shared
```

Spoke Router DMVPN Tunnels

```
interface Tunnel0
 ip address 192.168.0.5 255.255.255.0
 no ip redirects
 ip mtu 1400
 ip nhrp authentication WanExamp
 ip nhrp map 192.168.0.1 10.0.7.17
 ip nhrp map multicast 10.0.7.17
 ip nhrp network-id 12345
 ip nhrp holdtime 60
 ip nhrp nhs 192.168.0.1
 ip nhrp registration timeout 30
 ip tcp adjust-mss 1360
 tunnel source Serial1/0
 tunnel mode gre multipoint
 tunnel key 12345
 tunnel vrf Internet
 tunnel protection ipsec profile DMVPN shared
```

Spoke Router Interface Configuration

```
ip vrf Internet
 rd 65000:1
!
interface Loopback0
 ip address 10.0.1.5 255.255.255.255
!
interface FastEthernet0/0
 ip address 172.16.11.1 255.255.255.0
 duplex auto
 speed auto
!
interface Serial1/0
 description Link to Internet(ROUTER) s1/2
 ip vrf forwarding Internet
 ip address 10.0.7.9 255.255.255.252
 encapsulation ppp
 no peer neighbor-route
!
ip route vrf Internet 0.0.0.0 0.0.0.0 Serial1/0
```

Hub Router BGP Configuration

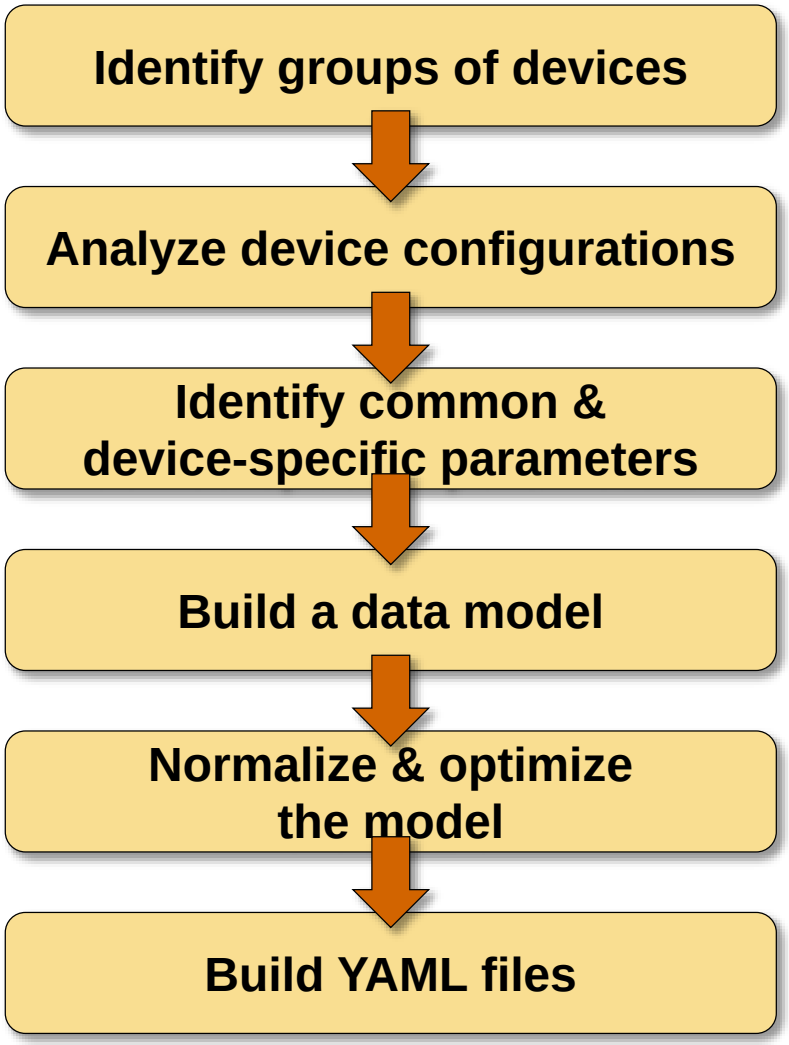
```
router bgp 65000
 !
 no synchronization
 bgp log-neighbor-changes
 redistribute connected route-map Internal
 redistribute ospf 1 route-map Internal
 neighbor 10.0.1.2 remote-as 65000
 neighbor 10.0.1.2 update-source Loopback0
 neighbor spokes peer-group
 bgp listen range 192.168.0.0/24 peer-group spokes
 neighbor spokes update-source Tunnel0
 neighbor spokes remote-as 65000
 neighbor spokes route-reflector-client
 neighbor spokes next-hop-self all
 neighbor spokes send-community
 neighbor spokes default-information originate
```

Spoke Router BGP Configuration

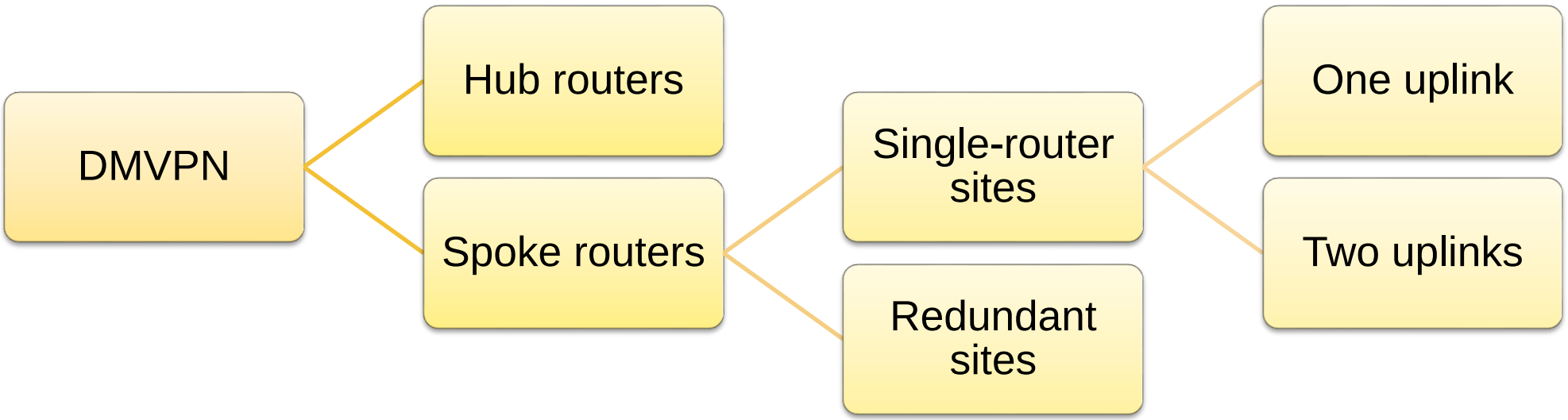
```
router bgp 65000
 no synchronization
 bgp log-neighbor-changes
 redistribute connected route-map Internal
 neighbor 192.168.0.1 remote-as 65000
 neighbor 192.168.0.1 update-source Tunnel0
 neighbor 192.168.1.1 remote-as 65000
 neighbor 192.168.1.1 update-source Tunnel1
 no auto-summary
```

Generate the Data Model

Generate the Data Model



Identify Groups of Devices



Identify Parameters

Common parameters

- DNS, NTP servers and Syslog servers
- Usernames and passwords
- Tunnel parameters (NHRP ID, hub IP address, GRE key…)

Per-group parameters

- LAN and WAN interface names

Per-router parameters

- Hostname
- LAN subnet
- Loopback IP address
- Tunnel IP address

Building the Data Model

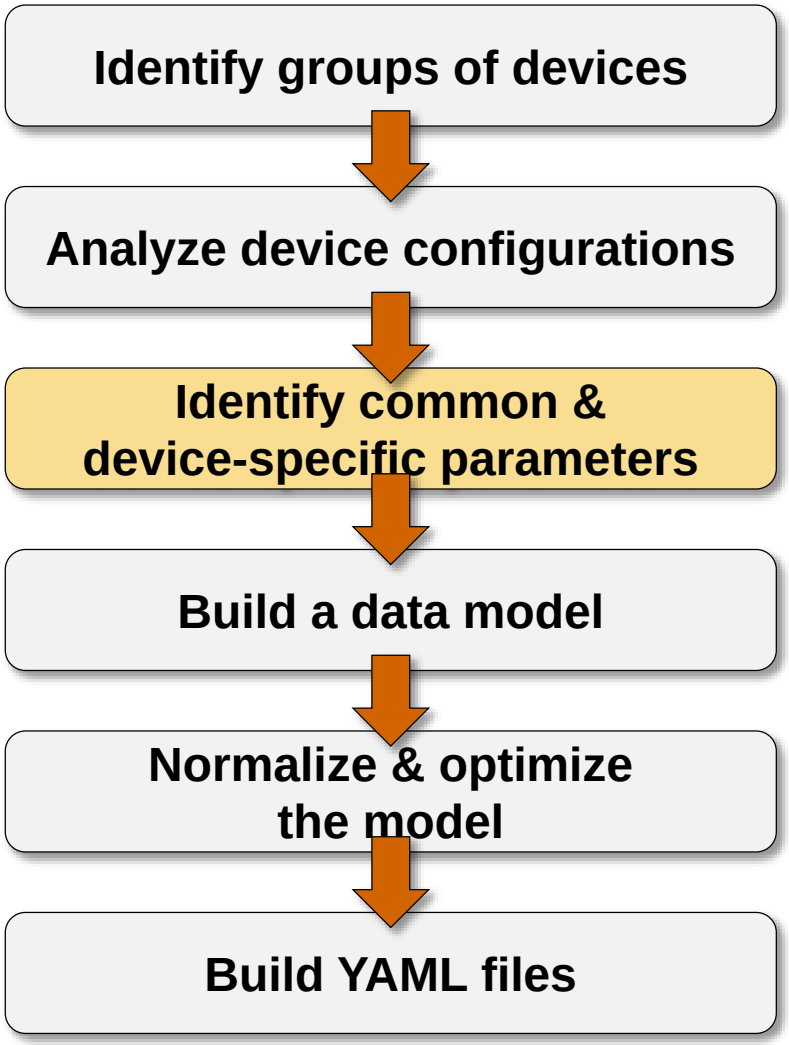
YAML = Object-Oriented Data Model

- Objects have properties
- Property can be a value, an object, or an array of values or objects

Ansible provides inheritance of variables (parameters) between objects

- Nodes inherit variables of the groups and roles they belong to
- Groups inherit variables of parent groups
- All nodes inherit common variables
- More-specific value replaces less-specific value

Generate the Data Model



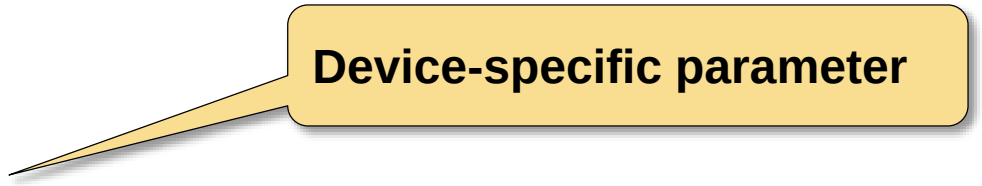
IPsec Configuration

```
crypto keyring DMVPN
  pre-shared-key address 0.0.0.0 0.0.0.0 key TESTING
!
crypto isakmp policy 10
 authentication pre-share
 group 2
!
!
crypto ipsec transform-set DMVPN esp-des esp-sha-hmac
 mode transport
!
crypto ipsec profile DMVPN
 set transform-set DMVPN
```

Common parameter

Might be a common parameter

Hub Router DMVPN Tunnel




```
interface Tunnel0
 bandwidth 10000
 ip address 192.168.0.1 255.255.255.0
 no ip redirects
 ip mtu 1400
 ip nhrp authentication WanExamp
 ip nhrp map multicast dynamic
 ip nhrp network-id 12345
 ip tcp adjust-mss 1360
 tunnel source Serial1/0
 tunnel mode gre multipoint
 tunnel key 12345
 tunnel protection ipsec profile DMVPN shared
```

Spoke Router DMVPN Tunnels

```
interface Tunnel0
 ip address 192.168.0.5 255.255.255.0
 no ip redirects
 ip mtu 1400
 ip nhrp authentication WanExamp
 ip nhrp map 192.168.0.1 10.0.7.17
 ip nhrp map multicast 10.0.7.17
 ip nhrp network-id 12345
 ip nhrp holdtime 60
 ip nhrp nhs 192.168.0.1
 ip nhrp registration timeout 30
 ip tcp adjust-mss 1360
 tunnel source Serial1/0
 tunnel mode gre multipoint
 tunnel key 12345
 tunnel vrf Internet
 tunnel protection ipsec profile DMVPN shared
```

Could be global parameter or derived from hub router parameters

Spoke Router Interface Configuration

```
ip vrf Internet
 rd 65000:1
!
interface Loopback0
 ip address 10.0.1.5 255.255.255.255
!
interface FastEthernet0/0
 ip address 172.16.11.1 255.255.255.0
 duplex auto
 speed auto
!
interface Serial1/0
 description Link to Internet(ROUTER) s1/2
 ip vrf forwarding Internet
 ip address 10.0.7.9 255.255.255.252
 encapsulation ppp
 no peer neighbor-route
!
ip route vrf Internet 0.0.0.0 0.0.0.0 Serial1/0
```

Hub Router BGP Configuration

```
router bgp 65000
 !
 no synchronization
 bgp log-neighbor-changes
 redistribute connected route-map Internal
 redistribute ospf 1 route-map Internal
 neighbor 10.0.1.2 remote-as 65000
 neighbor 10.0.1.2 update-source Loopback0
 neighbor spokes peer-group
 bgp listen range 192.168.0.0/24 peer-group spokes
 neighbor spokes update-source Tunnel0
 neighbor spokes remote-as 65000
 neighbor spokes route-reflector-client
 neighbor spokes next-hop-self all
 neighbor spokes send-community
 neighbor spokes default-information originate
```

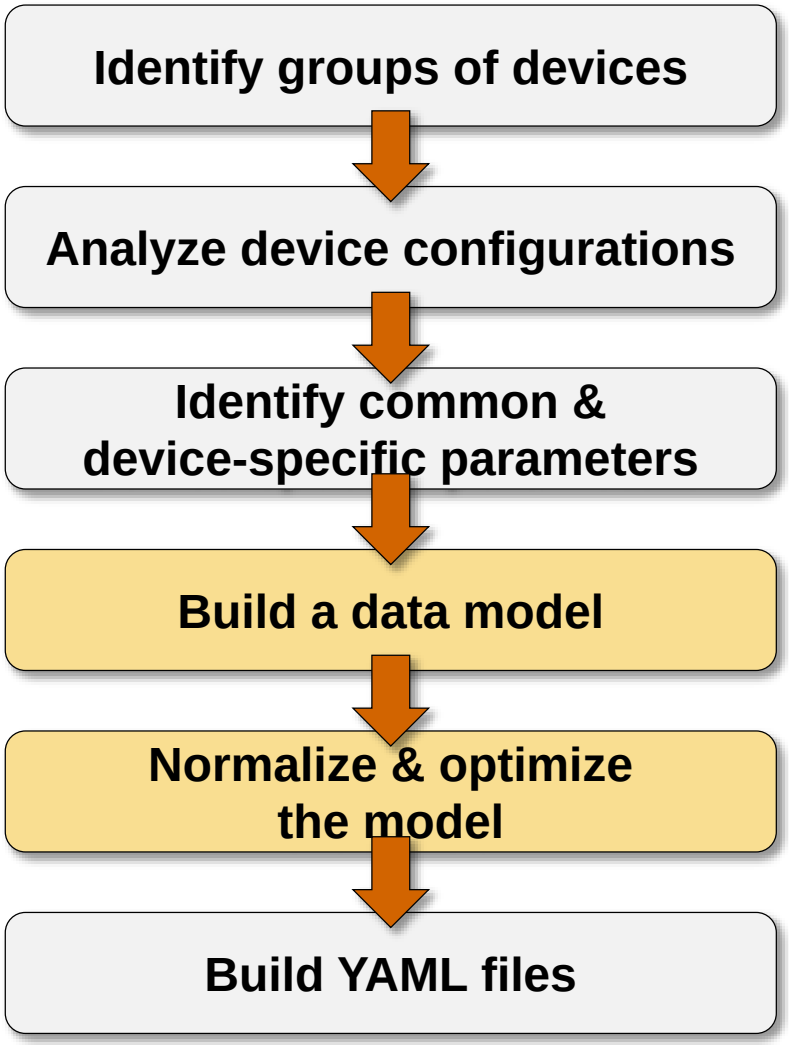
Could be local parameter or derived from list of peers

Spoke Router BGP Configuration

```
router bgp 65000
 no synchronization
 bgp log-neighbor-changes
 redistribute connected route-map Internal
 neighbor 192.168.0.1 remote-as 65000
 neighbor 192.168.0.1 update-source Tunnel0
 neighbor 192.168.1.1 remote-as 65000
 neighbor 192.168.1.1 update-source Tunnel1
 no auto-summary
```

Could be global parameter or derived from hub router parameters

Generate the Data Model



Build, Normalize and Optimize the Data Model

- List all global and per-node variables
- Identify which global variables are truly global and which belong to groups or roles
- Try to group variables into objects
- Avoid value duplication as much as possible

Example: Per-Router Variables

Hostname	R2
Loopback_IP	10.0.1.5
LAN_Interface	FastEthernet0/0
LAN_IP	172.16.11.1
LAN_Subnet	255.255.255.0
WAN_0_Interface	Serial0/1
WAN_0_IP	10.0.7.9
WAN_0_Subnet	255.255.255.252
WAN_1_Interface	FastEthernet1/0
WAN_1_IP	DHCP
DMVPN_0_IP	192.168.0.5
DMVPN_1_IP	192.168.1.5

Optimization: Turn Interface Variables into Objects

Hostname			R2
Loopback		IP	10.0.1.5
LAN		Interface	FastEthernet0/0
		IP	172.16.11.1
		Subnet	255.255.255.0
WAN	0	Interface	Serial0/1
		IP	10.0.7.9
		Subnet	255.255.255.252
	1	Interface	FastEthernet1/0
		IP	DHCP
DMVPN	0	IP	192.168.0.5
	1	IP	192.168.1.5

Optimization: Default Subnet Masks

Hostname			R2
Loopback		IP	10.0.1.5
LAN		Interface	FastEthernet0/0
		IP	172.16.11.1
		~~Subnet~~	~~255.255.255.0~~
WAN	0	Interface	Serial0/1
		IP	10.0.7.9
		Subnet	255.255.255.252
	1	Interface	FastEthernet1/0
		IP	DHCP
DMVPN	0	IP	192.168.0.5
	1	IP	192.168.1.5

Optimization: Base Interface Addresses on Router ID

Hostname			R2
ID			5
LAN	Interface		FastEthernet0/0
WAN	0	Interface	Serial0/1
		IP	10.0.7.9
		Subnet	255.255.255.252
	1	Interface	FastEthernet1/0
		IP	DHCP
DMVPN	0		
	1		

Example: DMVPN Tunnel Parameters

Tunnel_MTU	1400
Tunnel_MSS	1360
Tunnel_1_Auth	WanExamp
Tunnel_1_Hub_IP	192.168.0.1
Tunnel_1_Hub_Phy	10.0.7.17
Tunnel_1_NHRP_ID	12345
Tunnel_1_GRE	12345
Tunnel_2_Auth	WanExamp
Tunnel_2_Hub_IP	192.168.1.1
Tunnel_1_Hub_Phy	10.0.7.13
Tunnel_1_NHRP_ID	12346
Tunnel_1_GRE	12346

Optimization: Convert to Objects

Tunnel	MTU	1400	
	MSS	1360	
	0	Auth	WanExamp
		Hub_IP	192.168.0.1
		Hub_Phy	10.0.7.17
		NHRP_ID	12345
		GRE	12345
	1	Auth	WanExamp
		Hub_IP	192.168.1.1
		Hub_Phy	10.0.7.13
		NHRP_ID	12346
		GRE	12346

Optimization: Lookup Values from Hub Nodes

Tunnel	MTU	1400	
	MSS	1360	
	0	Auth	WanExamp
		Hub_Router	C1
		NHRP_ID	12345
		GRE	12345
	1	Auth	WanExamp
		Hub_Router	C2
		NHRP_ID	12346
		GRE	12346

Lookup into C1.WAN.0.IP

DMVPN Data Model in YAML Format

Preparing YAML Data Model for Ansible

- YAML is just a data markup language
- Data stored in YAML format must be saved in text files according to conventions used by the target tool

Typical Ansible deployment:

- Shared variables will be stored in *group_vars* directory

  group_vars/all.yml

- Per-node variables will be stored in individual files in *host_vars* directory

  host_vars/router.yml

Sample all.yml

```
---
log_dir: logs
build_dir: build
domain_name: lab.ipspace.net

as: 65000
tunnel:
  mtu: 1400
  mss: 1360
  0:
    auth: WanExamp
    hub_router: C1
    nhrp_id: 12345
    gre: 12345
  1:
    auth: WanExamp
    hub_router: C2
    nhrp_id: 12346
    gre: 12346
```

Tunnel	MTU	1400	
	MSS	1360	
	0	Auth	WanExamp
		Hub_Router	C1
		NHRP_ID	12345
		GRE	12345
	1	Auth	WanExamp
		Hub_Router	C2
		NHRP_ID	12346
		GRE	12346

Sample R2.yml

```
---
hostname: 'R2'
loopback: { ip: 10.0.1.5 }
LAN:
  interface: 'FastEthernet0/0'
  ip: 172.16.11.1
WAN:
  0:
    interface: 'Serial0/1'
    ip: 10.0.7.9
    subnet: 255.255.255.252
  1: { interface: 'FastEthernet1/0', ip: DHCP }
DMVPN: { 0: { ip: 192.168.0.5'}, 1: { ip: 192.168.1.5} }
```

Hostname			R2
Loopback	IP		10.0.1.5
LAN	Interface		FastEthernet0/0
	IP		172.16.11.1
	Subnet		255.255.255.0
WAN	0	Interface	Serial0/1
		IP	10.0.7.9
		Subnet	255.255.255.252
	1	Interface	FastEthernet1/0
		IP	DHCP
DMVPN	0	IP	192.168.0.5
	1	IP	192.168.1.5

Preparing Configuration Templates

Preparing Configuration Templates

Create a generic sample configuration for every group of devices

- Try to make configurations as generic as possible
- Try to merge specifics of similar groups into a single template (we'll use conditional expressions to use them)
- Try to use loops as much as possible
- Replace global or node-specific parameters with variable names
- Split configurations into smaller templates (common, interfaces, routing protocols, ACLs…)
- Try to retain the order of commands (if you want to use vendor configuration comparison tools)

Example: Spoke Router DMVPN Configuration

```
interface Tunnel0
 ip address DMVPN.0.IP 255.255.255.0
 no ip redirects
 ip mtu Tunnel.MTU
 ip nhrp authentication Tunnel.0.Auth
 …
 tunnel source WAN.0.Interface
 tunnel mode gre multipoint
 tunnel key Tunnel.0.GRE
 tunnel protection ipsec profile DMVPN shared
```

Tunnel	MTU		1400
	MSS		1360
	0	Auth	WanExamp
		Hub_IP	192.168.0.1
		Hub_Phy	10.0.7.17
		NHRP_ID	12345
		GRE	12345
	1	Auth	WanExamp
		Hub_IP	192.168.1.1
		Hub_Phy	10.0.7.13
		NHRP_ID	12346
		GRE	12346

WAN	0	Interface	Serial0/1
		IP	10.0.7.9
		Subnet	255.255.255.252
	1	Interface	FastEthernet1/0
		IP	DHCP
DMVPN	0	IP	192.168.0.5
	1	IP	192.168.1.5

Generalize: One or Two Tunnel Interfaces

If exists DMVPN.1.IP

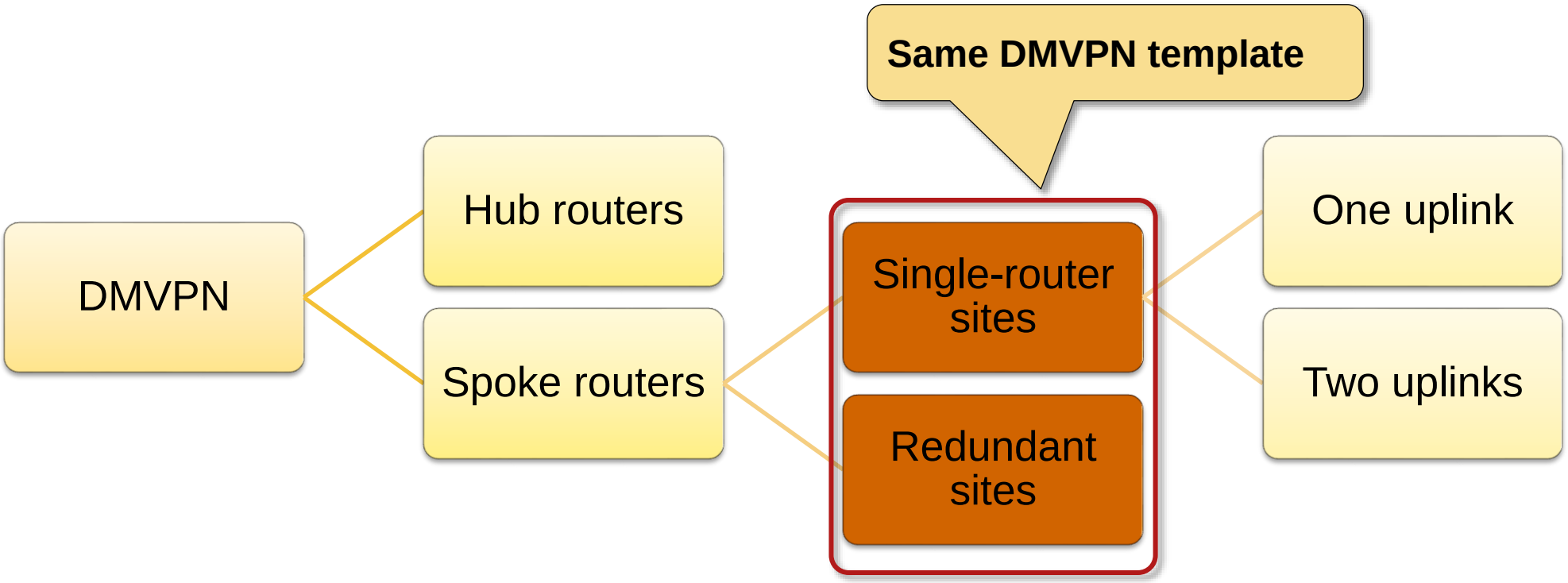
```
interface Tunnel1
 ip address DMVPN.1.IP 255.255.255.0
 no ip redirects
 ip mtu Tunnel.MTU
 ip nhrp authentication Tunnel.1.Auth
 …
 tunnel source WAN.1.Interface
 tunnel mode gre multipoint
 tunnel key Tunnel.1.GRE
 …
```

End If

Tunnel	MTU		1400
	MSS		1360
	0	Auth	WanExamp
		Hub_IP	192.168.0.1
		Hub_Phy	10.0.7.17
		NHRP_ID	12345
		GRE	12345
	1	Auth	WanExamp
		Hub_IP	192.168.1.1
		Hub_Phy	10.0.7.13
		NHRP_ID	12346
		GRE	12346

WAN	0	Interface	Serial0/1
		IP	10.0.7.9
		Subnet	255.255.255.252
	1	Interface	FastEthernet1/0
		IP	DHCP
DMVPN	0	IP	192.168.0.5
	1	IP	192.168.1.5

Result: Merged Single- and Dual-DMVPN Templates



Generalize: One or Two WAN Uplinks

If exists DMVPN.1.IP

```
interface Tunnel0
 ip address DMVPN.1.IP 255.255.255.0
 no ip redirects
 ip mtu Tunnel.MTU
 ip nhrp authentication Tunnel.1.Auth
```

If exists WAN.1.Interface

```
 tunnel source WAN.1.Interface
```

Else

```
 tunnel source WAN.0.Interface
```

End If

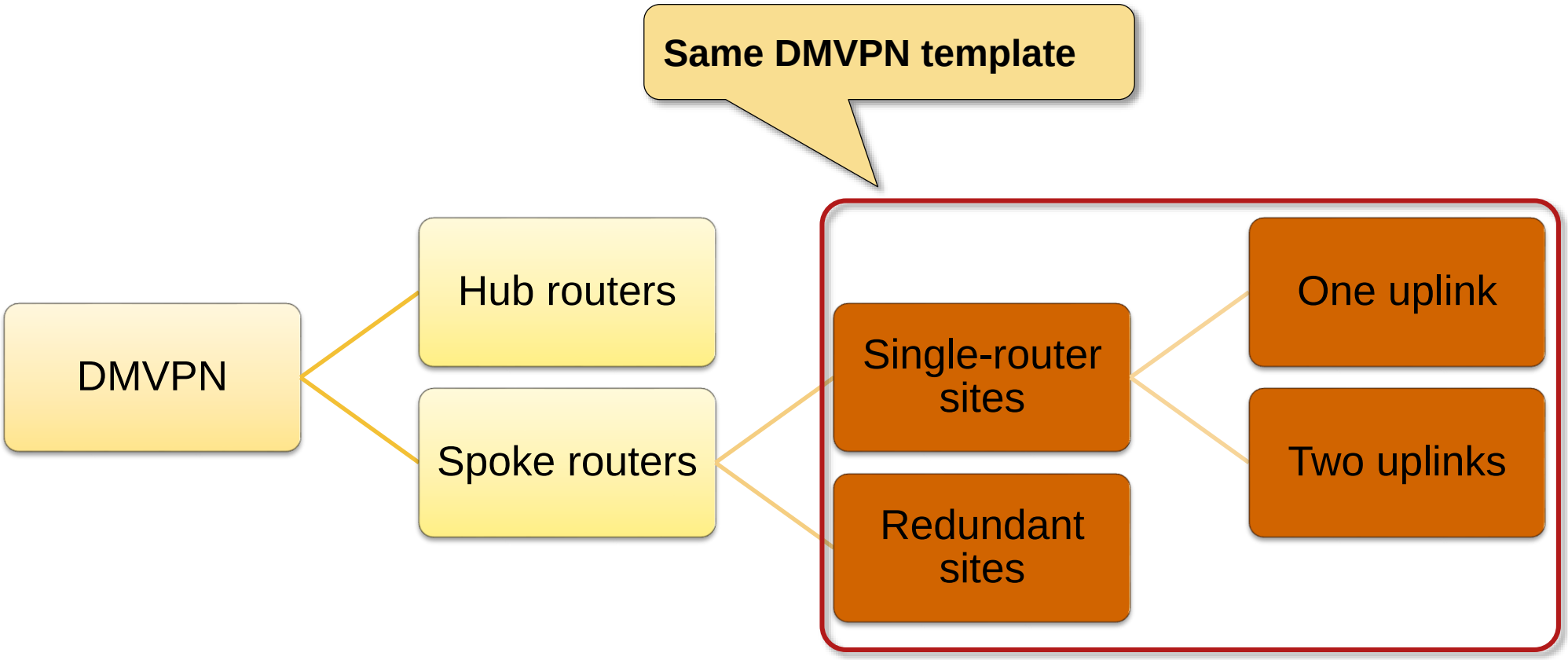
```
 tunnel mode gre multipoint
 tunnel key Tunnel.1.GRE
 …
```

End If

Tunnel	MTU		1400
	MSS		1360
	0	Auth	WanExamp
		Hub_IP	192.168.0.1
		Hub_Phy	10.0.7.17
		NHRP_ID	12345
		GRE	12345
	1	Auth	WanExamp
		Hub_IP	192.168.1.1
		Hub_Phy	10.0.7.13
		NHRP_ID	12346
		GRE	12346

WAN	0	Interface	Serial0/1
		IP	10.0.7.9
		Subnet	255.255.255.252
	1	Interface	FastEthernet1/0
		IP	DHCP
DMVPN	0	IP	192.168.0.5
	1	IP	192.168.1.5

Result: Merged All Spoke DMVPN Templates



Sample Jinja2 DMVPN Templates

Split Spoke Config Into Parts (main.conf)

```
{% include 'common_start.conf' %}
{% include 'interfaces.conf' %}
{% include 'routing.conf' %}
{% include 'ACL.conf' %}
{% include 'common_end.conf' %}
```

Common Parts of Router Configuration

```
upgrade fpd auto
version 15.0
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname {{hostname}}
!
boot-start-marker
boot-end-marker
!
logging buffered 4096
!
no aaa new-model
!
```

Interface Configuration – the Easy Part

```
interface Loopback0
 ip address {{loopback.ip}} 255.255.255.255
 !
!
interface {{LAN.interface}}
 ip address {{LAN.ip}} 255.255.255.0
 !
!
interface {{WAN.0.interface}}
 description WAN uplink
 ip vrf forwarding Internet
 ip address {{WAN.0.ip}} {{WAN.0.subnet}}
 encapsulation ppp
 no peer neighbor-route
 serial restart-delay 0
```

Default subnet mask

Won't work on Ethernet uplink

Interface Configuration – Specifying Defaults

```
interface Loopback0
 ip address {{loopback.ip}} 255.255.255.255
 !
!
interface {{LAN.interface}}
 ip address {{LAN.ip}} {{LAN.subnet|default('255.255.255.0')}}
 !
!
interface {{WAN.0.interface}}
 description WAN uplink
 ip vrf forwarding Internet
 ip address {{WAN.0.ip}} {{WAN.0.subnet|default('255.255.255.0')}}
 encapsulation ppp
 no peer neighbor-route
 serial restart-delay 0
```

Won't work on Ethernet uplink

Recognizing Serial Interfaces

```
interface {{WAN.0.interface}}
 description WAN uplink
 ip vrf forwarding Internet
 ip address {{WAN.0.ip}} {{WAN.0.subnet|default('255.255.255.0')}}
{% if WAN.0.interface > 'Serial' %}
 encapsulation ppp
 no peer neighbor-route
 serial restart-delay 0
{% endif %}
```

Adding Second WAN Uplink

```
{% if WAN.1 is defined %}
interface {{WAN.1.interface}}
 description WAN uplink
 ip vrf forwarding Internet
 ip address {{WAN.1.ip}} {{WAN.1.subnet|default('255.255.255.0')}}
 {% if WAN.1.interface > 'Serial' %}
 encapsulation ppp
 no peer neighbor-route
 serial restart-delay 0
 {% endif %}
 !
{% endif %}
```

Looping over WAN Interfaces

```
{% for intf in WAN %}
interface {{WAN[intf].interface}}
 description WAN uplink
 ip vrf forwarding Internet
 ip address {{WAN[intf].ip}} {{WAN[intf].subnet|default(…)}}
{% if WAN[intf].interface > 'Serial' %}
 encapsulation ppp
 no peer neighbor-route
 serial restart-delay 0
{% endif %}
 !
{% endfor %}
```

Fixing DHCP-based Addresses

```
{% for intf in WAN %}
interface {{WAN[intf].interface}}
 description WAN uplink
 ip vrf forwarding Internet
{% if WAN[intf].ip == 'DHCP' %}
 ip address dhcp
{% else %}
 ip address {{WAN[intf].ip}} {{WAN[intf].subnet|default(…)}}
{% endif %}
{% if WAN[intf].interface > 'Serial' %}
 encapsulation ppp
 no peer neighbor-route
 serial restart-delay 0
{% endif %}
{% endfor %}
```

Tunnel Interface Configuration

```
interface Tunnel0
 ip address {{DMVPN.0.ip}} 255.255.255.0
 no ip redirects
 ip mtu {{tunnel.mtu}}
 ip nhrp authentication {{tunnel.0.auth}}
 ip nhrp map 192.168.0.1 10.0.7.17
 ip nhrp map multicast 10.0.7.17
 ip nhrp network-id {{tunnel.0.nhrp_id}}
 ip nhrp holdtime 60
 ip nhrp nhs 192.168.0.1
 ip nhrp registration timeout 30
 ip tcp adjust-mss {{tunnel.mss}}
 tunnel source {{WAN.0.interface}}
 tunnel mode gre multipoint
 tunnel key {{tunnel.0.gre}}
```

Need lookup from hub router parameters

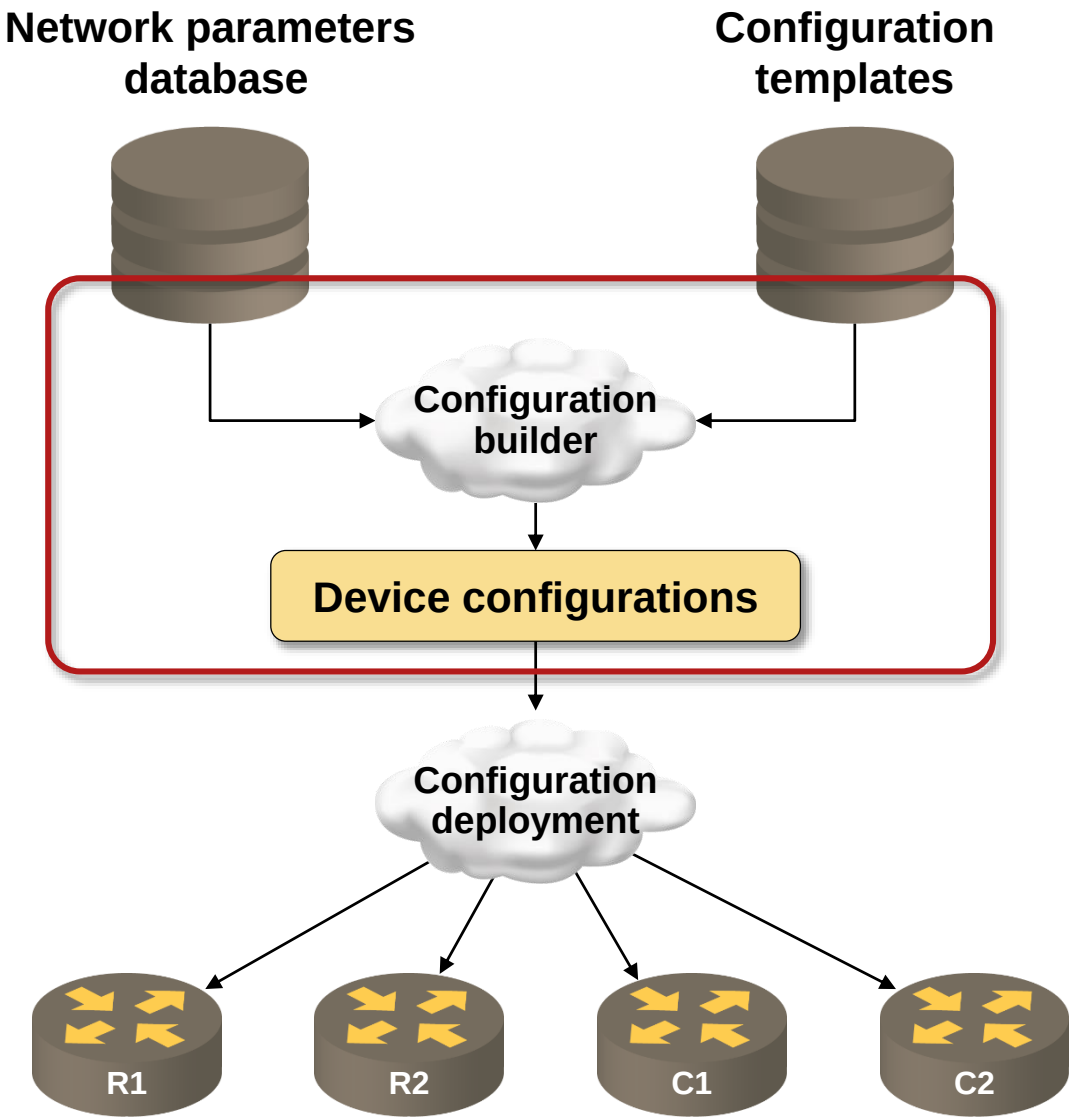
Referencing Variables from Other Nodes

```
interface Tunnel0
 ip address {{DMVPN.0.ip}} 255.255.255.0
 no ip redirects
 ip mtu {{tunnel.mtu}}
 ip nhrp authentication {{tunnel.0.auth}}
 ip nhrp map {{ hostvars[tunnel.0.hub_router].DMVPN.0.ip }} →
       {{ hostvars[tunnel.0.hub_router].WAN.0.ip }}
 ip nhrp map multicast →
       {{ hostvars[tunnel.0.hub_router].WAN.0.ip }}
 ip nhrp network-id {{tunnel.0.nhrp_id}}
 ip nhrp holdtime 60
 ip nhrp nhs {{ hostvars[tunnel.0.hub_router].DMVPN.0.ip }}
```

Automating Configuration Generation with Ansible

ipSpace

The Goal



Debugging the Data Structure

```
---
- name: Dump variables for each host
  hosts: all
  connection: local
  gather_facts: no

  tasks:
    - name: display host variables
      debug: var=hostvars
```

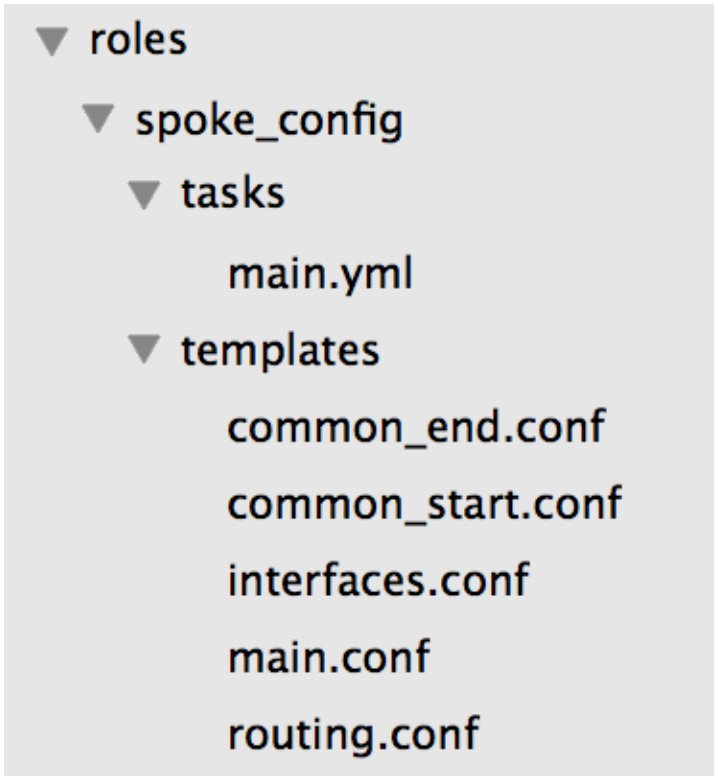
Generating the Configurations

```
---
- name: Creating build directories for each host
  hosts: all
  connection: local
  gather_facts: no

  tasks:
    - name: remove host config file
      file: path={{ build_dir }}/{{ inventory_hostname }}.conf →
            state=absent

- name: Generate templates for each device by roles
  hosts: spokes
  connection: local
  gather_facts: no
  roles:
    - spoke_config
```

Role-Specific Playbooks and Templates




```
---
- name: template building
  template: src=main.conf dest= →
        {{ build_dir }}/ →
        {{inventory_hostname}}.conf
```

Questions?

Paperwork issues

- Follow-up email
- Please fill in the evaluation form
- Recording available within 24 hours
- PDF materials always available for download
- Discount for future webinars – register @ my.ipspace.net
- Upgrade to yearly subscription
- Help me spread the word!

Send them to ip@ipSpace.net or @ioshints