Генератор конфига из NOC. На примере Dlink DES-3200

В ожидании прихода большого количества коммутаторов, которые в кратчайшие сроки нужно будет настроить, было принято решение автоматизировать их начальную настройку. А именно - создать на них пользовательские vlan-ы, vlan для управления, назначить адреса и прописать маршрут по умолчанию и т.д.

Для этого нужно:

1. Созданы Managed Object Profile для данной группы коммутаторов
2. В Address Space Management должен присутствовать к которому относятся адреса коммутаторов
3. Создать Selector для данной групы коммутаторов
4. Создать в VC VCDomain для данной группы коммутаторов. И добавить требуемые vlan. Для vlan по которому будет выполняться коммутатором добавить description "Mvlan"
5. В Managed Objects добавляем коммутаторы, назначаем им VCDomain, в атрибутах прописываем платформу.

Сам снипет имеет такой вид

```
{% load python %}
{% var cmd internal %}
{% var Client str %}
{% var po str %}
{% var id str %}
{% var speed str %}
{% python %}
from noc.inv.models import *
from noc.sa.models import *
from noc.ip.models import *
from noc.vc.models import *
from noc.lib.ip import IP
from noc.lib.text import split_alnum
if "3200-10" in context['object'].platform:
    uplink = "9-10"
    user_port = "1-8"
    all_ports = "1-10"
if "3200-28" in context['object'].platform:
    uplink = "25-28"
    user_port = "1-24"
    all_ports = "1-28"
if "DES-1228/ME" in context['object'].platform:
    uplink = "25-28"
    user_port = "1-24"
    all_ports = "1-28"
if "DGS-3120-24SC" in context['object'].platform:
    uplink = "9-24"
    user_port = "1-8"
    all_ports = "1-24"
ip = context['object'].address
ip_prefix = Prefix.objects.filter(address=Address.objects.get(address=ip))
ip_prefix = str(ip_prefix[0]).split(':')[1]
ip_address = ip + str(ip_prefix[-3:])
gw = str(ip_prefix).split('/')[0]
gw = str(gw).split('.')
gateway = gw[0]+ "." +gw[1]+ "." + gw[2]+ "." + str(int(gw[3])+1)

context["cmd"]="\n"
context["cmd"]+="config snmp system_location " + str(context['object'].object_profile) + "\n"
context["cmd"]+="config command_prompt " + context['object'].name + "\n"
vcdomain = str(context['object'].vc_domain)
vlans = VC.objects.filter(vc_domain=VCDomain.objects.get(name=vcdomain))
for vl in vlans:
    if "MVlan" in str(vl.description):
        context["cmd"]+="config vlan " + vl.name + " add untagged " + uplink + "\n"
        context["cmd"]+="config ipif System ipaddress " + ip_address + "\n"
        context["cmd"]+="config ipif System vlan " + vl.name + "\n"
        context["cmd"]+="create iproute default " + gateway + " 1 primary\n"
    if "default" in vl.name:
        context["cmd"]+="config vlan default delete " + all_ports + "\n"
    else:
        context["cmd"]+="create vlan " + vl.name + " tag " + str(vl.l1) + "\n"
        context["cmd"]+="config vlan " + vl.name + " add tagged " + uplink + "\n"
{% endpython %}
{{cmd}}
```

Что делаем:

строки 14-29 - наследие более старых скриптов, определяем аплинки, абонентские порты и диапазон всех портов.

строка 30 - определяем адрес данного коммутатора

строка 31-36 определяем ip адрес коммутатора, его маску, шлюз по умолчанию

строка 41-42 - определяем к какому VCDomain относится данный коммутатор и какие vlan принадлежат ему.

строки 38-40, 43-53 - генерируем набор команд для данного коммутатора.

Возможно такой подход идеологически не верен, но позволяет быстро создать набор команд необходимых для начальной настройки.