Ces slides sont disponibles en ligne !

https://lamei.re/frnog34.pdf

Industrialisation réseau pour les sysadmins

Quand le cœur de réseau n'est pas le cœur de métier

Alexis LAMEIRE

FRnOG 34

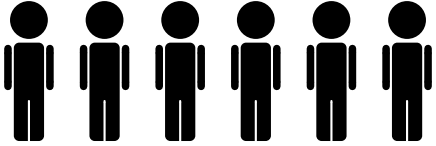
1er octobre 2021

Who am I

- Alexis Lameire
- Dompteur de bytes
- DC monkey
- Chez Weborama depuis 3 ans

Contexte

L’équipe



Les technos

Couchbase

NGINX

SALTSTACK

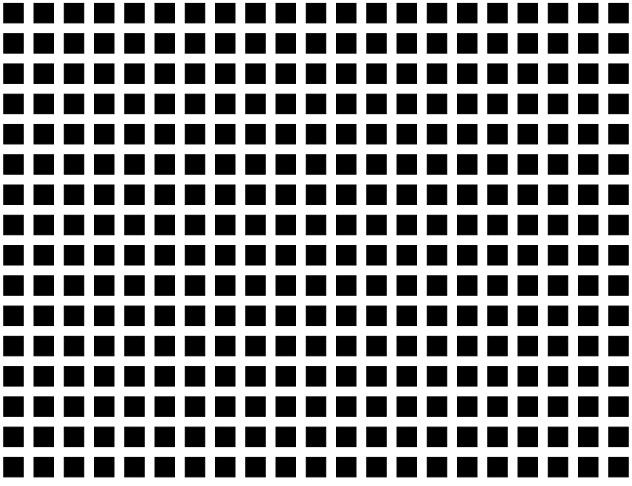
HashiCorp
Terraform

MariaDB

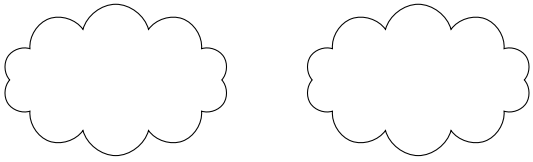
APACHE
kafka™

L'infra

L'infra



L’infra



« *Le cœur de réseau*
n'est pas le cœur de métier »

Faire et défaire, c’est toujours travailler

Le réseau pour les sysadmins : cahier des charges

- Être tolérant aux pannes, techniques… comme humaines

Le réseau pour les sysadmins : cahier des charges

- Être tolérant aux pannes, techniques… comme humaines
- Simple à administrer, simple à comprendre

Le réseau pour les sysadmins : cahier des charges

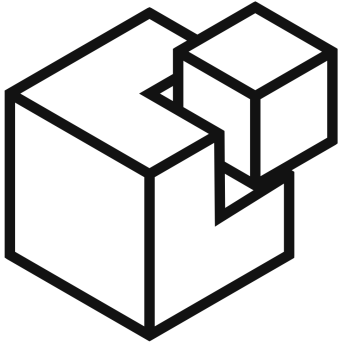
- Être tolérant aux pannes, techniques… comme humaines
- Simple à administrer, simple à comprendre
- Le moins de protocoles possibles, le plus facile à comprendre

Le réseau pour les sysadmins : cahier des charges

- Être tolérant aux pannes, techniques… comme humaines
- Simple à administrer, simple à comprendre
- Le moins de protocoles possibles, le plus facile à comprendre
- Être un bon citoyen de l'internet

« *Le meilleur outil d'automatisation est celui que l'on maîtrise* »

Mes collègues et moi utilisons…



SALTSTACK

Mais en fait, on s'en fout !

```
{% from "software/eos/map.jinja" import eos with context %}

Manage hardware config:
  netconfig.managed:
    - saltenv: {{ saltenv }}
    - replace: True
    - template_name:
      - salt://software/eos/files/basic.jinja
      - salt://software/eos/files/layer1.jinja
{% if eos.run.mlag.with_mlag %}
      - salt://software/eos/files/mlag.jinja
{% endif %}
{% if eos.run.vrf.cluster is defined %}
      - salt://software/eos/files/vrf.jinja
{% endif %}
{% if eos.run.layer2.cluster_name is defined %}
      - salt://software/eos/files/layer2.jinja
{% endif %}
{% if eos.run.layer3.types|length > 0 %}
      - salt://software/eos/files/layer3.jinja
{% endif %}
```

Orchestration VS Compliance

Orchestration

Un déploiement ⇒ plusieurs machines ⇒ une formule

Orchestration VS Compliance

Orchestration

Un déploiement ⇒ plusieurs machines ⇒ une formule

Compliance

Un état ⇒ une machine ⇒ plusieurs formules

« *Toute configuration manquante sera ajoutée par l'automatisation* »

« *Toute configuration manquante sera ajouté par l'automatisation*

Toute configuration non automatisée sera supprimée par l'automatisation »

Automatisation chez Weborama : les grands principes

Voir l'infrastructure dans son ensemble…

…Et non chaque équipement individuellement

```
eos:
  layer2:
    clusters:
      clusterName:
        members:
          - switch-01
          - switch-02
    gateways:
      PUBLIC_NETWORK:
        clusters:
          clusterName:
            MyVlan:
              id: 80
              network: 128.66.200.1/24
              addr_offset: 252
    aggr:
      clusterName:
        po100:
          [...]
```

Rendre facile ce qui est complexe

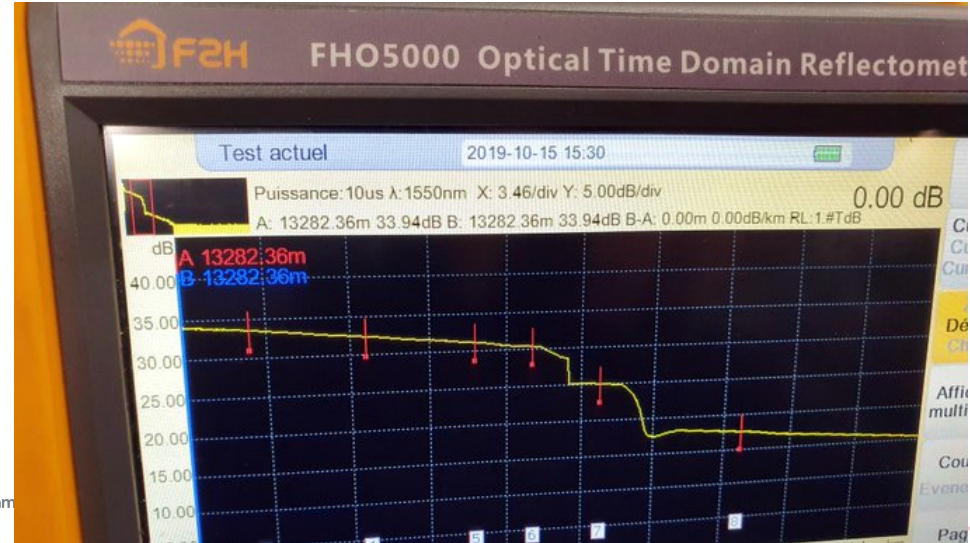
```
[...]
clusterName:
  MyVlan:
    id: 42
    network: 128.66.200.1/24
    addr_offset: 252
    redistribute: True
    redundancy_type: varp
[...]
```

Permettre les évolutions

```
{% set http_ip = ['128.66.0.66', '128.66.42.17']
layer5:
  clusters:
    MyCluster:
      eth25.100:
        rule: MY-ACL
    rules:
      My-ACL:
{% for ip in http_ip %}
        - action: permit
          proto: tcp
          daddr: {{ ip }}
          dport:
            - 80
            - 443
{% endfor %}
        - action: deny
          log: True
```

RETEX

Le cross connect farceur !



La DFZ pour les nuls

```
[...]
        common:
          te:
            SOME-AS-FIX:
              source_asn: 200350
              priority: low
            OTHER-AS-FIX:
              source_asn: 4580
              priority: low
          clusters:
            some_tier_one:
              members:
                - rt-provider: xe1/2/3
                - my_router: eth1
              subnet: 128.66.42.0/30
              remote_asn: 666
              type: ebgp-public-transit
              priority_out:
                - SOME-AS-FIX
                - OTHER-AS-FIX
[...]
```

```
{% if 'priority_out' in cluster -%}
{%   for rule in cluster.priority_out -%}
route-map {{ out_priority_route_map }} permit {{ loop.index * 10 }}
{%-     if 'source_asn' in layer3.vrfs[vrf_name].common.te[rule] %}
  match as-path {{ rule }}
{%-     endif %}
  set local-preference {{
    run.clusters_priorities[cluster.type]
    [layer3.vrfs[vrf_name].common.te[rule].priority] }}
exit

{%     if 'source_asn' in layer3.vrfs[vrf_name].common.te[rule] -%}
ip as-path access-list {{ rule }} permit _{{
  layer3.vrfs[vrf_name].common.te[rule].source_asn }}$ any
ip as-path access-list {{ rule }} deny .* any
{%     endif -%}
{%-   endfor %}

route-map {{ out_priority_route_map }} permit {{ (cluster.priority_out|length + 1)*10 }}
exit
{%- endif %}
```

Jinja 2 n’est PAS un bon moteur de template

```
{%     set cluster_members = cluster.members|map('list')|map('first')|list -%}
```

Jinja 2 n’est PAS un bon moteur de template

```
{% set i = 10 %}
{% if i == 10 %}
{%   set i = 11 %}
{% endif %}
{{ i }} # i = 10
```

Jinja hack : les variables

```
{{ eos.run.layer2.setdefault('vrrp_priority', 100) }}
or
{{ eos.run.layer2.update({
    'redundancy_types': eos.run.layer2.redundancy_types|unique
   })
}}
```

Des fois, il faut changer de langage

```
{%- set password = salt['eos_utils.genpwd'](
      username, eos.id, basic.usernames[username].password) %}
[...]
username {{ username }} role {{ role }} secret sha512 {{ password }}
```

Des fois, il faut changer de langage

```
from passlib.hash import sha512_crypt
import hashlib
import base64

def genpwd(username, router_id, password):
    '''
    gendPwd : generate a strong arista ,password

    the salt part of the password is generated using base64 with static informations
    '''
    sourceHashString = username + '@' + router_id
    hashedString = hashlib.sha512(sourceHashString).digest()
    b64salt = base64.b64encode(hashedString)[:16].replace('+', '.')

    return sha512_crypt.encrypt(password,rounds=5000,salt=b64salt)
```

Résultats

Questions ?

(Au fait, on recrute…)