Fast Provisioning with

NetBox and Ansible

Fast Provisioning with NetBox and Ansible

- Management of IT is time-consuming and fault-prone.
- Automation
  - Reduces the time for a single, repetitive task.
  - Reduces the errors in production. Normally.

- *I will always choose a lazy person to do a difficult job because a lazy person will find an easy way to do it.*

  False attribution to Bill Gates

Netbox: Source of Truth

- NetBox as the central *Source of Truth* in the IT.
  - IPAM and DCIM and ...
- Manages all entities needed in operation.
  - *https://github.com/netbox-community/netbox/labels/type%3A%20enhancement*
- Servers are combined to clusters.
- Virtual machines are assigned to cluster.

Ansible for Deployment

- From NetBox to reality you need a Configuration Management

- Modern Solutions: Saltstack or Ansible
  - I'll use Ansible in this talk
  - Saltstack will also work

Tell Ansible to use information from NetBox

> Setup an authentication token in NetBox.

> Configure the Ansible plugin

```
plugin: netbox
api_endpoint: http://[::1]
validate_certs: False
token: 0123456
config_context: True
group_by:
  - device_roles
  - platforms

compose:
  cluster: cluster.name
  interfaces: interfaces
```

Add the Ansible inventory

> Add the NetBox inventory plugin to Ansible config

```
inventory =
/etc/ansible/hosts,/usr/share/ansible/plugins/inventory
```

> Check the connection

```
# ansible-inventory --list
```

Ansible Playbook Outline

> Find the right cluster and node for new machine

> Prepare Kickstart (RHEL)

> Prepare Storage

> Create machine

> Initialize machine (kickstart process)

Find the Right Cluster for a Given Virtual Server I

```
- name: Cluster for virtual machine
  uri:
    Url: "http://[::1]/api/virtualization/virtual-machines/ \
          ?name={{ inventory_hostname }}"
    method: GET
    return_content: yes
    headers:
      Accept: application/json
      Authorization: Token {{netbox_token}}
  register: nb_vm
  delegate_to: localhost
```

Find the Right Cluster for a Given Virtual Server II

```
- name: List of hosts in the cluster
  uri:
    url: "http://[::1]/api/dcim/devices/ \
          ?cluster_id={{ nb_vm.json.results[0].cluster.id }}"
    method: GET
    return_content: yes
    headers:
      Accept: application/json
      Authorization: Token {{netbox_token}}
  register: nb_hosts
  delegate_to: localhost

- name: Select node name
  set_fact:
    host: "{{ nb_hosts.json.results[nb_hosts.json.count | \
           random | int].name }}"
```

Find the Correct Volume Group on that Node

```
- name: Get the VG name on host "{{ host }}"
  uri:
    url: "http://[::1]/api/dcim/devices/?name={{ host }}"
      method: GET
      return_content: yes
      headers:
        Accept: application/json
        Authorization: Token {{netbox_token}}
      register: nb_host
      delegate_to: localhost

- name: Set fact of VG name
  set_fact:
    vgname: "{{ nb_host.json.results[0].config_context.virtVG }}"

- name: Output
  debug:
    msg: "New machine {{ inventory_hostname }} on host {{ host }} \
          disk on {{ vgname }}"
```

Prepare Kickstart for the New Server I

```
- name: Create basic disc image
  vars:
    dd_cmd: dd if=/dev/zero \
      of=/var/lib/libvirt/images/ks-"{{ inventory_hostname }}".img \
      bs=1M count=1
  shell: "{{ dd_cmd }}"
  delegate_to: "{{ host }}"

- name: Create filesystem on ks file
  shell: "mkfs -t ext4 \
    /var/lib/libvirt/images/ks-{{ inventory_hostname }}.img"
  delegate_to: "{{ host }}"

- name: Give the filesystem the correct label
  shell: "e2label \
    /var/lib/libvirt/images/ks-{{ inventory_hostname }}.img OEMDRV"
  delegate_to: "{{ host }}"
```

Prepare Kickstart for the New Server II

```
- name: Mount ks filesytem
  shell: "mount -o loop \
    /var/lib/libvirt/images/ks-{{ inventory_hostname }}.img /mnt/ks/"
  delegate_to: "{{ host }}"

- name: Copy kickstart file from template
  template:
    src: ks-template.j2
    dest: "/mnt/ks/ks.cfg"
  delegate_to: "{{ host }}"

- name: Umount ks file
  shell: "umount /mnt/ks"
    delegate_to: "{{ host }}"
```

The Kickstart Template

```
(…)
ipv6={{ primary_ip6 }}/64 --activate -
(…)
network  --hostname={{ inventory_hostname }}.sys4.de
(…)

part pv.453 --fstype="lvmpv" --ondisk=vda \
  --size={{ disk * 1024 - 1024 -1024 }}
volgroup cl_{{ inventory_hostname }} --pesize=4096 pv.453

logvol swap --fstype="swap" --size=1024 -name=swap \
  --vgname=cl_{{ inventory_hostname }}
logvol /var --fstype="ext4" --size=4096 --name=var \
  --vgname=cl_{{ inventory_hostname }}
logvol / --fstype="ext4" --size={{ 13*1024 }} --name=root \
  --vgname=cl_{{ inventory_hostname }}
```

Create New Virtual Machine

```
- name: Create VM
  virt:
    command: define
    xml: "{{ lookup('template', 'c8-template.j2') }}"
    autostart: no
  delegate_to: "{{ host }}"
```

New RHEL Template

```
<domain type='kvm'>
  <name>{{ inventory_hostname }}</name>
  <uuid>{{ (9999999999999999999 | random | string + \
           (lookup('pipe', 'date +%s%N'))) | to_uuid() }} </uuid>

  <memory unit='MiB'>{{ memory }}</memory>
  <currentMemory unit='MiB'>{{ memory }}</currentMemory>
  <vcpu placement='static'>{{ vcpus }}</vcpu>

  <devices>
    <disk type='block' device='disk'>
      <source dev='/dev/{{ vgname }}/{{ inventory_hostname }}'/>
    </disk>
    <disk type='file' device='disk'>
      <source \
file='/var/lib/libvirt/images/ks{{ inventory_hostname }}.img'/>
    </disk>
  </devices>
</domain>
```

Start the New Machine

```
- name: Start new VM
  virt:
    name: "{{ inventory_hostname }}"
    command: start
  delegate_to: "{{ host }}"

- name: Wait until pings
  command: ping -c1 "{{ primary_ip6 }}"
  register: r
  delegate_to: localhost
  until: r.rc == 0
  delay: 30
  retries: 40

- name: Set boot from HD
  shell: "virt-xml {{ inventory_hostname }} --edit --boot hd"
  delegate_to: "{{ host }}"
```

Reboot the New Machine

```
- name: Wait until host went down. Kickstart takes it time.
  command: ping -c1 "{{ primary_ip6 }}"
  register: r
  delegate_to: localhost
  until: r.rc != 0
  delay: 30
  retries: 40
  ignore_errors: yes

- name: Reboot virtual machine
  virt:
    name: "{{ inventory_hostname }}"
    command: start
  delegate_to: "{{ host }}"
```

Further Reading

- I am going to write a blog article about this
- Detailed templates will be included there
- https://sys4.de/blog
  - There are a lot of German blogs, but mine are mostly English

Bonus Track: Add new Server to Monitoring

```
- name: Create host in Zabbix
  local_action:
    module: zabbix_host
    server_url: https://mon.sys4.de
    login_user: user
    login_password: XXXXXX
    host_name: '{{ inventory_hostname }}'
    host_groups:
      - Linux servers
      - sys4
    status: enabled
    interfaces:
      - type: 2
        ip: '{{ ansible_host }}'
        useip: 1
        main: 1
    link_templates:
      - Template SNMP OS Linux AES
  when: tags is defined and '"Monitoring" in tags'
```

Thank You for Your Attention

> Contact

ms@sys4.de

https://sys4.de