OpenShift 4 automatisch mit dem orcharchino installieren

User-Provisioned Infrastructure

Dr. Jan Bundesmann

October 6th, 2021
1. Installing OpenShift
2. CoreOS Installation on orcharhino
3. Automatic OpenShift installation
4. Summary and outlook
Agenda

1. Installing OpenShift
2. CoreOS Installation on orcharhino
3. Automatic OpenShift installation
4. Summary and outlook
Installation scheme

ATIX #ocharchino #openshift – 2
Installation variants

Automatic installation
- AWS, GCE, Azure
- vSphere, RHV
- OpenStack

User-provisioned infrastructure
- Bare-metal
- Any hypervisor

Let orcharhino help you!
User-provisioned infrastructure

hosts
1. bootstrap node (temporary)
2. master nodes
3. worker nodes

network / load balancer
- api.cluster.domain: masters
- api-int.cluster.domain: masters
- *.apps.cluster.domain: workers

deployment automatism for CoreOS
Agenda

1. Installing OpenShift
2. CoreOS Installation on orcharhino
3. Automatic OpenShift installation
4. Summary and outlook
What is CoreOS

- Immutable minimal version of Red Hat Enterprise Linux
- Installation = copy of filesystem image
- Contains container runtime (cri-o & podman)
- Adjustments though ignition
  - Environment for unattended installation
  - Partitioning
  - Creates files (systemd, network config, ...)

#ATIX #orcharhino #openshift – 5
orcharhino quickstart

- Host lifecycle management
  - Host provisioning tools
  - (Content management)
  - (Configuration management)
- Very nice templating engine
- Focus on the pets in “pets cs. cattle”
Demo time

Watch me deploy a CoreOS host!
Problems encountered

➤ Reboot loop aka. “When is CoreOS installed?”
➤ Prepare and provide ignition file
➤ Allow to supply individual ignition file
Starting CoreOS

Fedora provides (https://getfedora.org/en/coreos/download)

- ISO
- qcow2
- ova
- PXE (kernel + initramfs + rootfs)
- ...
Break the reboot loop

Start from local disk, only start PXELinux if no OS can be found

```bash
1  DEFAULT local_chain_hd0
2
3  TIMEOUT 20
4  PROMPT 0
5  ONERROR coreos
6  ONTIMEOUT coreos
7
8  LABEL local_chain_hd0
9  MENU LABEL Chainload the first hard drive (hd0)
10  COM32 chain.c32
11  APPEND hd0
12
13  LABEL coreos
14  KERNEL <%= @kernel %>
15  APPEND initrd=<%= @initrd %> ...
```
Creating ignition files

- Red Hat does not encourage users to write ignition files
- Productive use case: OpenShift 4
- OpenShift-Installer creates ignition files
- Tool to create ignition files from YAML: butane

Goal

- create generic ignition file for orcharhino deployment
- include auto-generated code as additional config
Ignition files

► JSON files

► Equivalent of kickstart, preseed, or AutoYAST files

```json
{
  "ignition": {
    "version": "3.1.0"
  },
  "passwd": {
    "users": [
      {
        "name": "core",
        "sshAuthorizedKeys": [ "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAB..." ]
      }
    ],
    "systemd": {
      "units": [
        {
          "contents": "[Unit]
Description=Tell orcharhino I'm ready
After=network-online.target

[Service]
Type=simple
ExecStart=< raw_script_command >
RestartSec=5
Restart=on-failure

[Install]
WantedBy = multi-user.target"
        }
      ]
    }
}
```

#ATIX #orcharhino #openshift – 12
Additional ignition

- Ignition understands the key `ignition.config.merge`
- Expects packed content

```ruby
<% merge_pre = "merge": [{"compression": "gzip", "source": "data:;base64," merge_post = ""}],
if (host_param('additional_ignition_packed') != "")
  additional_ignition = merge_pre + host_param('additional_ignition_packed') + merge_post
else
  additional_ignition = ""
end
<% %>
{
  "ignition": {
    "config": {
      <%= additional_ignition %>
    },
    "version": "3.1.0"
  },
...}
```

How obtain `additional_ignition_packed`?
Create additional_ignition_packed

This does not work:

```
1  cat addition.ign | gzip | base64 -w0
```

But this:

```
1  variant: openshift
2  version: 4.8.0
3  metadata:
4   name: hostname
5   labels:
6     machineconfiguration.openshift.io/role: bootstrap
7  ignition:
8   config:
9    merge:
10   - local: "./addition.ign"
```

Create packed version

```
1  podman run -v .:/data --interactive --rm quay.io/coreos/butane:v0.11.0 \
2   --files-dir /data/ -r --pretty --strict < merge.yaml \
3   | jq -r .ignition.config.merge[].source \
4   | cut -d , -f 2
```
Procedure for orcharhino

- Two templates rendered by orcharhino for CoreOS: PXELinux and Provisioning
- Additional config can be added through a host parameter
- With some minor additions this allows for a one-click installation of CoreOS
Agenda

1. Installing OpenShift
2. CoreOS Installation on orcharhino
3. Automatic OpenShift installation
4. Summary and outlook

#ATIX #orcharhino #openshift
### Showcase

<table>
<thead>
<tr>
<th>Pipeline</th>
<th>Needs</th>
<th>Jobs</th>
<th>Tests</th>
<th>Status</th>
<th>Job ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>0</td>
<td>Only Once</td>
<td></td>
<td>setup basics</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>passed</td>
<td>#111922</td>
<td>docker</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>manual</td>
</tr>
<tr>
<td></td>
<td>Vms</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>deploy_vms</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>passed</td>
<td>#111923</td>
<td>docker</td>
</tr>
<tr>
<td></td>
<td>Orcharhino</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>install_orcharhino</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>passed</td>
<td>#111925</td>
<td>docker</td>
</tr>
<tr>
<td></td>
<td>Openshift</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>install_openshift</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>passed</td>
<td>#111926</td>
<td>docker</td>
</tr>
</tbody>
</table>

ATIX #orcharhino #openshift – 16
Installing OpenShift – Overview

1. Network: Configure load balancer and set DNS entries
2. Obtain live image
3. Create deployment parameters on orcharhino
   - operating system entry
   - templates
   - installation medium
   - compute profile
   - host group
4. Create packed ignition files
5. Start OpenShift nodes in correct order
   - bootstrap
   - control plane
   - worker nodes
Installing OpenShift – first steps

- HAproxy entries for api, api-int, and *.apps
- DNS entries through nsupdate for api, api-int, apps and *.apps
- Download CoreOS ISO file and unpack it.
Installing OpenShift – CoreOS as operating system

- hosts: orcharhinos
  tasks:
  - name: define default variables for orcharhino communication
    set_fact: ...
  - name: Create dummy ptable for coreos
    theforeman.foreman.partition_table:
      <<: *orcharhino_connection
      name: coreos
      layout: '# content required'
      os_family: Redhat
  - name: "Create Operating System CoreOS"
    theforeman.foreman.operatingsystem:
      <<: *orcharhino_connection
      name: CoreOS
      major: "4"
      minor: "6"
      os_family: Redhat
      state: present
      password_hash: MD5
      architectures:
      - x86_64
      ptables:
      - coreos
- name: Create template from ignition file
  theforeman.foreman.provisioning_template:
    <<: *orcharhino_connection
    template: '{{ lookup("file", "files/ignition.json.erb") }}'
    operating_systems: ["CoreOS"]

- name: Create PXELinux template
  theforeman.foreman.provisioning_template:
    <<: *orcharhino_connection
    template: '{{ lookup("file", "files/pxelinux.erb") }}'
    operating_systems: ["CoreOS"]

- name: "Associate ignition file with CoreOS"
  theforeman.foreman.os_default_template:
    <<: *orcharhino_connection
    operating_system: "CoreOS"
    template_kind: "provision"
    provisioning_template: "Ignition File for CoreOS"

- name: "Associate ignition file with CoreOS"
  theforeman.foreman.os_default_template:
    <<: *orcharhino_connection
    operating_system: "CoreOS"
    template_kind: "PXELinux"
    provisioning_template: "CoreOS PXELinux for OpenShift"
Installing OpenShift – Installation medium

- name: create installation medium in orcharhino
  theforeman.foreman.installation_medium: <<: *orcharhino_connection
    name: coreos
    operating_systems:
    - CoreOS
    os_family: Redhat
    path: http://or.openshift.atix-training.de/pub/installation_media/coreos/iso
Installing OpenShift – Compute profile

- name: "Create compute profile for OCP"
  theforeman.foreman.compute_profile:
    <<: *orchrhino_connection
    name: ocp
    compute_attributes:
      - compute_resource: libvirt
        vm_attributes:
          cpus: "4"
          memory: "17179869184"
          ...

- Depends on compute resource (i.e. hypervisor)
- Error message does not indicate insufficient resources
Installing OpenShift – Host group

- name: "Create hostgroup for OCP"
  theforeman.foreman.hostgroup:
    <<: *orcharhino_connection
    name: ocp
    compute_profile: ocp
    architecture: x86_64
    operating_system: CoreOS
    parameters:
      - name: additional_ignition_packed
        value: ""
  ...

- Basic ingredient for one-click deployment.
- additional_ignition_packed as host parameter
Installing OpenShift – obtain openshift installer

- hosts: orcharhinos
  tasks:
  - name: "Create folder structure for openshift installation"
    ...
  - name: "Download OpenShift installer to recently created folder"
    get_url:
      dest: /opt/openshift-installer/openshift-install-linux.tar.gz
    ...
  - name: "unpack installer"
    unarchive:
      src: /opt/openshift-installer/openshift-install-linux.tar.gz
    ...
  - name: "Delete config folder"
    file:
    ...
  - name: "Recreate config folder"
    file:
    ...

▶ Deletion and recreation of config folder for reinstallation

#ATIX #orcharhino #openshift – 24
Installing OpenShift – Run openshift installer

```yaml
- name: "create config file"
  template:
    dest: /opt/openshift-installer/cluster/install-config.yaml

- name: "create manifests"
  command:
    cmd: ./openshift-install create manifests --dir=cluster

- name: "turn master non-schedulable"
  lineinfile:
    line: " mastersSchedulable: false"
    path: /opt/openshift-installer/cluster/manifests/cluster-scheduler-02-config.yml
    regexp: " mastersSchedulable: true"

- name: "create ignition files"
  command:
    cmd: ./openshift-install create ignition-configs --dir=cluster
```

- Config file contains credentials for quay.io
- Adjustments happen before ignition files are created
- Pack ignition files using butane and store them as variables
Installing OpenShift – Creating bootstrap node

- name: "create bootstrap node"
  theforeman.foreman.host:
    <<: *orcharhino_connection
    name: "bootstrap...."
    hostgroup: ocp
    parameters:
      - name: additional_ignition_packed
        parameter_type: string
        value: "{{ bootstrap_ignition }}"
      ...

- name: "start bootstrap node"
  theforeman.foreman.host_power:
    <<: *orcharhino_connection
    name: "bootstrap.openshift.atix-training.de"
    state: on
Installing OpenShift – Control plane nodes

```yaml
- name: "create master nodes"
  theforeman.foreman.host:
    <<: *orcharhino_connection
  name: "master{{ item }}..."
  hostgroup: ocp
  parameters:
    - name: additional_ignition_packed
      parameter_type: string
      value: "{{ master_ignition }}"
  ...

  loop:
    - 0
    - 1
    - 2

- name: "wait for bootstrap to complete and API become available"
  wait_for:
    host: "api-int.openshift.atix-training.de"
    port: 6443
    delay: 10

- name: "start master nodes"
  theforeman.foreman.host_power:
    <<: *orcharhino_connection
    name: "master{{ item }}.openshift.atix-training.de"
    state: on

...
Installing OpenShift — worker nodes

```
- name: "create worker nodes"
  theforeman.foreman.host:
    <<: *orcharhino_connection
    name: "worker{{ item }}..."
  hostgroup: ocp
  parameters:
    - name: additional_ignition_packed
      parameter_type: string
      value: "{{ worker_ignition }}"
...

- name: "start worker nodes"
  theforeman.foreman.host_power:
    <<: *orcharhino_connection
    name: "worker{{ item }}.openshift.atix-training.de"
    state: on
  loop:
    - 0
    - 1
    - 2
```
Installing OpenShift – trusting infrastructure

```
- name: wait for bootstrapper signing requests to appear
  command: >
    oc get csr
    -o go-template='{{range .items}}{{if not .status}}{{. metadata.name}}{{"\n"}}{{end}}{{end}}{{end}}'
  register: bootstrap_csrs
  retries: 30
  delay: 10
  until: bootstrap_csrs.stdout_lines | list | count > 2
  environment:
    KUBECONFIG: /opt/openshift-installer/cluster/auth/kubeconfig

- name: approve signing requests
  command: "oc adm certificate approve {{ item }}"
  loop: "{{ bootstrap_csrs.stdout_lines }}"
  environment:
    KUBECONFIG: /opt/openshift-installer/cluster/auth/kubeconfig
```
Installing OpenShift – Credentials

- name: `fetch cluster credentials`
  fetch:
    dest: "./local/{{ item }}"
    src: "/opt/openshift-installer/cluster/auth/{{ item }}"
  loop:
    - kubeconfig
    - kubeadmin-password
    ...

#ATIX #orcharhino #openshift – 30
Agenda

1. Installing OpenShift
2. CoreOS Installation on orcharhino
3. Automatic OpenShift installation
4. Summary and outlook
Summary and outlook

- RedHat extend immutable design pattern to all levels of OpenShift
- Automatic installation requires a deployment method for CoreOS
- orcharhino can render ignition files to bootstrap CoreOS
- Complete automation of OpenShift installation through orcharhino uses
  - openshift-installer
  - butane
  - Ansible with Foreman Ansible Modules
- Special network requirements to be implemented
  - orcharhino could act as a local mirror for airgapped installations
  - Static ip addresses can be provided through ignition