Cloud-init

Marc Skinner - Principal Solutions Architect

Michael Heldebrant - Solutions Architect

Red Hat

Agenda

- What is cloud-init?
- What can you do with cloud-init?
- How does it work?
- Using cloud-init enabled images
  - RHEV
  - RHOS
  - VMware
- Cloudforms leveraging cloud-init

What is cloud-init?

- Provides boot time customization for cloud and virtualization instances.
- Service runs early during boot, retrieves user data from an external provider and performs actions
- Supported user data formats:
  - Shell scripts (starts with #!)
  - Cloud config files (starts with #cloud-config)
    - Standard YAML syntax available for many common configuration operations.
  - MIME multipart archive.
    - Custom part handling also available.
- Modular and highly configurable.

What is cloud-init?

- cloud-init has modules for handling:
  - Disk configuration
  - Command execution
  - Creating users and groups
  - Package management
  - Writing content files
  - Bootstrapping Chef/Puppet
- Additional modules can be written in Python if desired.

What is cloud-init?

- Can be used to bootstrap other configuration management tools or agents.
- Widely used and broadly supported solution:
  - OpenStack
  - Amazon EC2
  - RHEV
  - VMware
- Written in Python but other implementations possible (e.g. the shells scripts used in the Cirros image).

What is cloud-init? - Data Categories

- **meta-data** is provided by the cloud platform.
- **user-data** is a chunk of arbitrary data the user provides.
- Retrieved from data source and saved to **/var/lib/cloud/**

What can you do with cloud-init?

- You may already be using it!:
  - Injects SSH keys.
  - Grows root filesystems.
- Other module support tasks such as:
  - Setting the hostname.
  - Setting the root password.
  - Setting locale and time zone.
  - Running custom scripts.

User-data Examples

- Upgrading and installing packages:

```
#cloud-config

package_upgrade: true

packages:

- git

- screen

- vim-enhanced
```


User-data Examples

- Run an arbitrary command:

```
#cloud-config

runcmd:

- rhnreg_ks --activationkey=3753...
```

- Or:

```
#!/bin/bash

rhnreg_ks --activationkey=3753...
```


User-data Examples

- Configure Puppet agent:

```
#cloud-config
puppet:
 conf:
  agent:
   server: "puppetmaster.example.org"
   certname: "%i.%f"
   cacert:
     -----BEGIN CERTIFICATE-----
     …
     -----END CERTIFICATE-----
```


User-data Examples

- Configure Chef:

```
#cloud-config
chef:
 install_type: "packages"
 force_install: false
 server_url: "https://chef.yourorg.com:4000"
 node_name: "your-node-name"
 environment: "production"
 validation_name: "yourorg-validator"
 validation_key: |
     -----BEGIN RSA PRIVATE KEY-----
     YOUR-ORGS-VALIDATION-KEY-HERE
     -----END RSA PRIVATE KEY-----
```


User-data Examples

- Configure Chef part 2:

```
run_list:
 - "recipe[apache2]"
 - "role[db]"
initial_attributes:
   apache:
     prefork:
       maxclients: 100
     keepalive: "off"
```


User-data Examples

- Including additional user-data files:

```
#include
http://config.example.com/cloud-config
http://config.dept.example.com/cloud-config
```

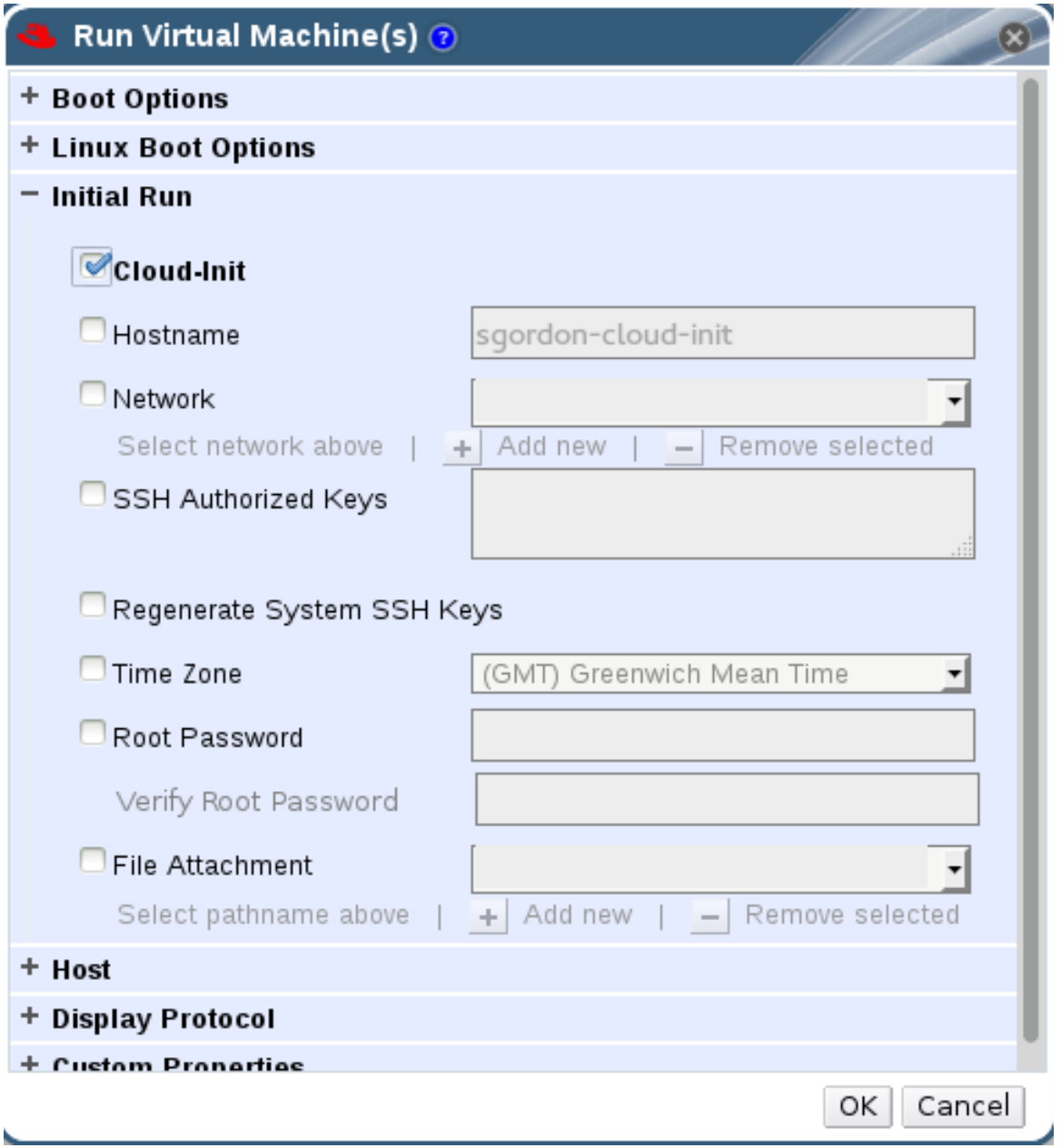

User-data Examples

- Other possibilities:
  - Additional YUM repository configuration.
  - Guest agent installation/configuration.
  - Use #include or arbitrary wget/curl command to retrieve configuration script from a central location.
  - phone_home to post objects to an arbitrary url
- More examples at:
  - http://cloudinit.readthedocs.org/en/latest/topics/examples.html

How does it work – RHEV

- cloud-init on RHEV searches for a floppy drive containing a user-data.txt file
- RHEV creates virtual floppy drive with user-data.txt file with content in a format cloud-init expects:
  - Shell script (#!)
  - Cloud-config (#cloud-config)

Using cloud-init enabled images (RHEV)



Using cloud-init enabled images (RHEV)

- Install the **rhel-guest-image-6** package from RH common:
  - `# yum install rhel-guest-image-6`
- Upload the image to an export domain:
  - `# engine-image-uploader upload --export-domain DefaultExport /usr/share/rhel-guest-image-6/rhel-guest-image-6-6.5-20140116.1-1/`
- Import the template into RHEV-M
- Create a virtual machine from the template.
- Click “Run Once”.
- Click “Initial Run” and then click “cloud-init”.

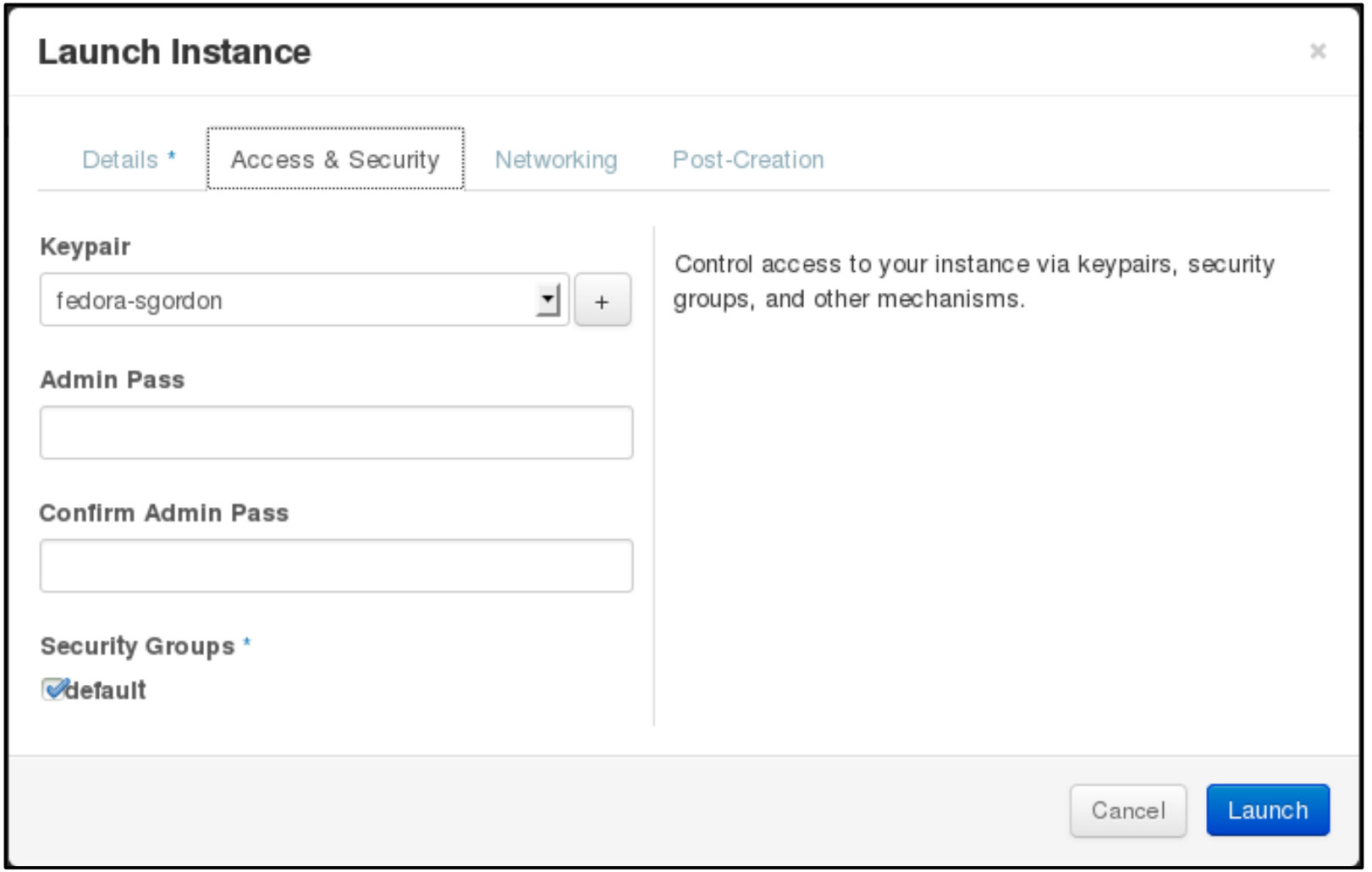
Using cloud-init enabled images (RHEV)

```
<vm>
  <payloads>
    <payload type="floppy">
      <file name="user-data.txt">
        <content>
        #!/bin/bash
        echo Testing... &gt;&gt; /root/testing.txt
        </content>
      </file>
    </payload>
  </payloads>
</vm>
```

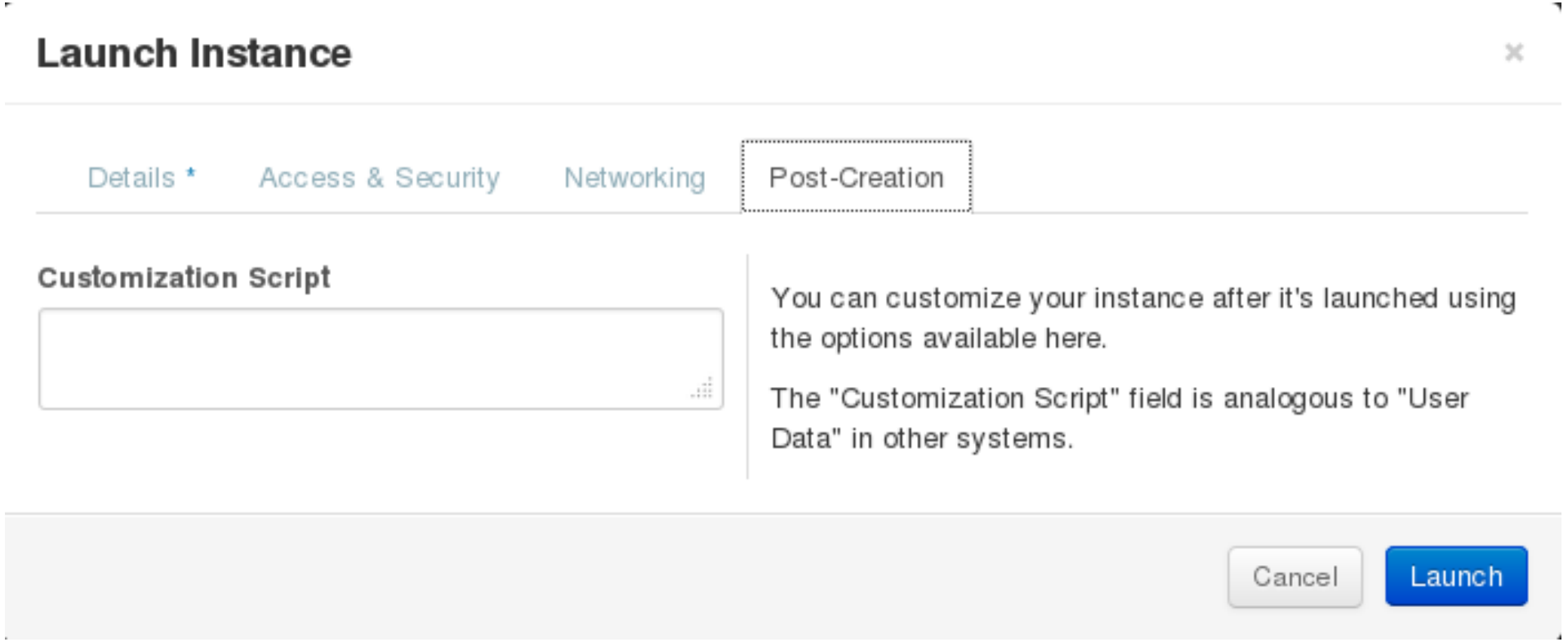

How does it work - OpenStack / EC2

- Accesses metadata service at
  - http://169.254.169.254/latest/meta-data
  - http://169.254.169.254/latest/user-data
- NAT rules on your network controller make this work.
- Service provided by **nova-api** (accessed via per-router **neutron-metadata-proxy** when using Neutron).

Using cloud-init enabled images (RHOS)



Using cloud-init enabled images (RHOS)



How does it work - OpenStack/EC2 Data Source

```
$ curl http://169.254.169.254/latest/meta-data
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
hostname
instance-action
instance-id
instance-type
kernel-id
local-hostname
local-ipv4
placement/
public-hostname
public-ipv4
public-keys/
ramdisk-id
Reservation-id

$ curl http://169.254.169.254/latest/user-data
#!/bin/bash
echo 'Extra user data here'
```


Using cloud-init enabled images (RHOS)

- Install the **rhel-guest-image-6** package from RH common:
  - `# yum install rhel-guest-image-6`
- Upload the image to glance:
  - `# glance image-create --name rhel65-image --disk-format=qcow2 –container-format=bare --is-public=True –file=/usr/share/rhel-guest-image-6/rhel-guest-image-6-6.5-20140116.1-1.qcow2`
- Launch an instance based on the image.

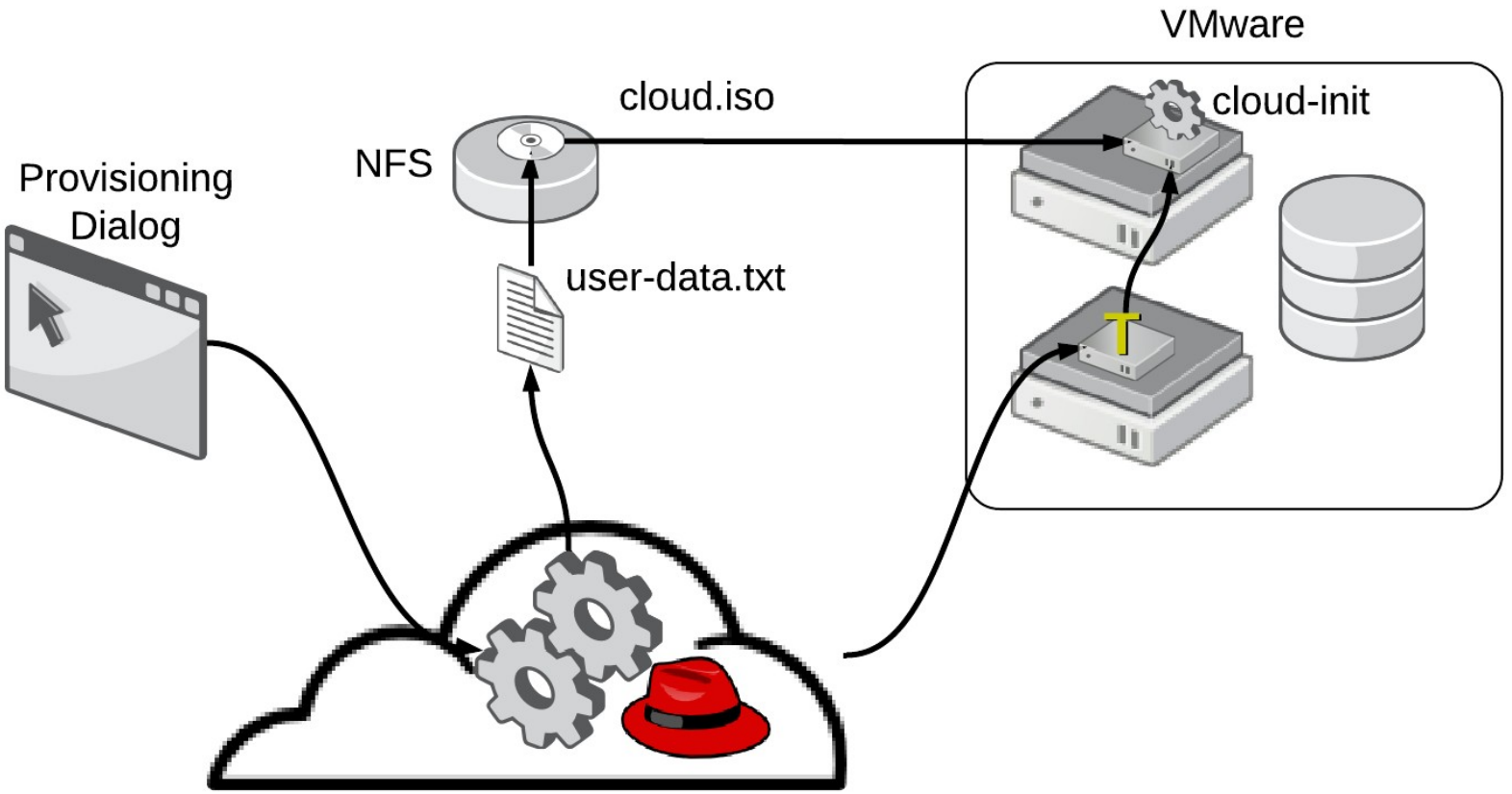
Using cloud-init enabled images (RHOS)

- Can also use the command line client:
  - `$ nova boot --image rhel-6.5 --flavor 1 --user-data mydata.file`

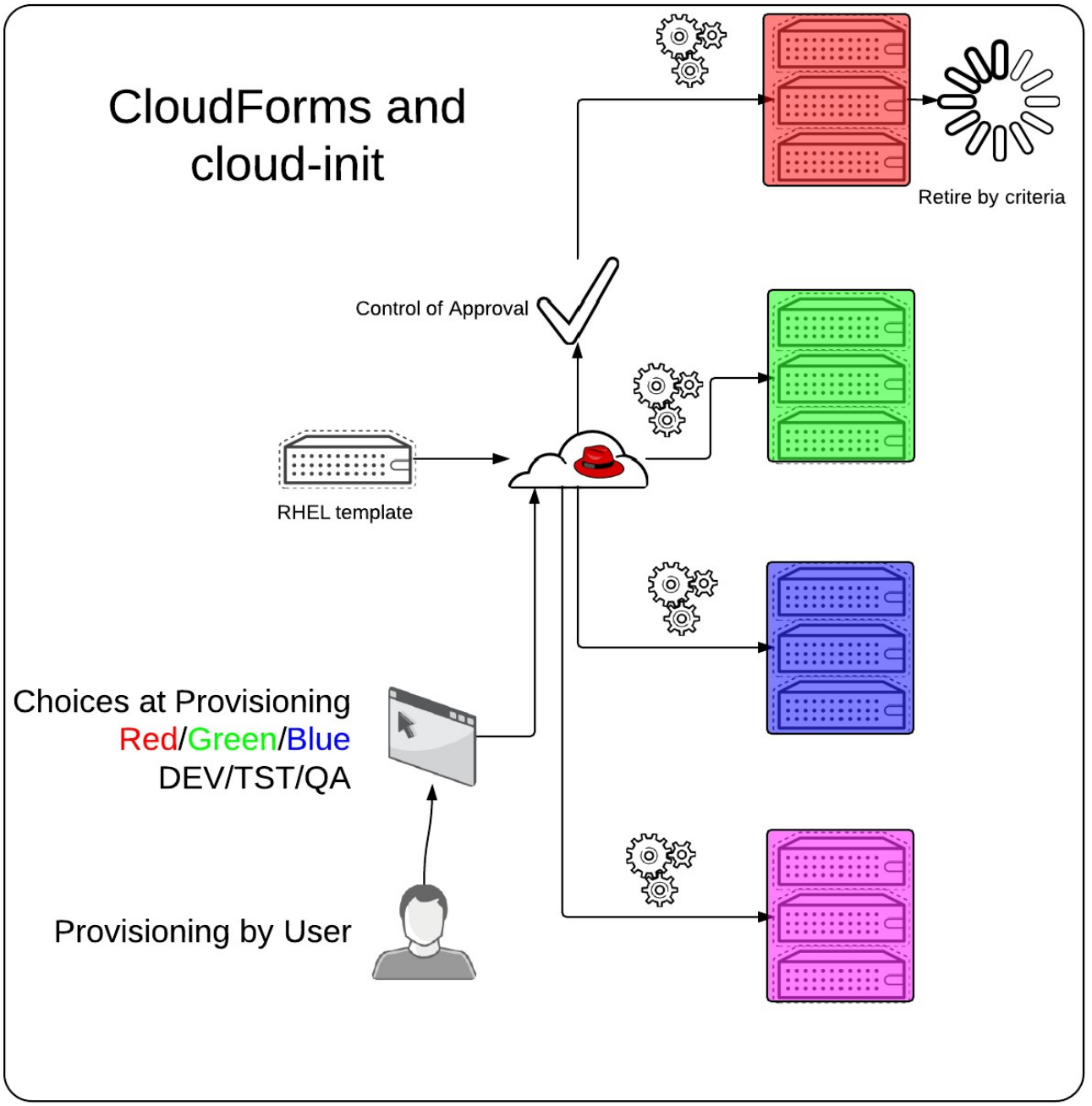
How does it work – VMware

- Requires control of power and cdrom drive to the vm and access to the iso domain or the vSphere client to mount the iso
- cloud-init on vSphere searches for a CDROM with volume id CDROM on boot
- guest in vSphere using cloud-init reads the iso mounted containing the user-data.txt file:
  - Shell script (#!)
  - Cloud-config (#cloud-config)

CloudForms: clone from template with cloud-init



Using cloud-init enabled images (VMware)



Getting cloud-init enabled images

- cloud-init package:
  - Included in Red Hat Common channel for RHEL.
    - 0.7.2 in EL6
    - 0.7.4 in EL7
  - Included in Fedora.
- Baked into many cloud images:
  - Red Hat Enterprise Linux
  - Fedora
- Easily added to custom images for most common distributions.

Adding cloud-init to your own images (RHEV)

- Install the cloud-init package
- Configure /etc/cloud/cloud.cfg
  - Allow root logins
    - disable_root: 0
  - Add additional modules to customize behavior
    - cloud-final-modules:
    - - package-update-upgrade-install
- Remove the following to templatize
  - /etc/udev/rules.d/70-persistent-*
  - /etc/ssh/ssh_host*
  - /etc/sysconfig/rhn/systemid

Using cloud-init enabled images

- Gotchas:
  - RHEV injects SSH key into root by default, which is disabled for SSH in the RHEL image (BZ # 1063518).
  - Free form text field for user data in UI does not capture “Enter” or “Shift + Enter”, need to paste multi-line data in (BZ # 1064567).

Debugging

- **/var/log/cloud-init.log** in the guest contains (very) verbose output from the run.
- **/var/lib/cloud/** contains the data retrieved from the metadata service on config drive.
- Run can be simulated/repeated from inside the guest:
  - `$ cloud-init [-h] [--version] [--file FILES] [--debug] [--force] {query,init,modules,single}`

Further Information

- Upstream Documentation
- How we use cloud-init in OpenStack Heat