Reimagining Image Management in the New Shasta Environment

CUG 2019

Harold Longley, Eric Cozzi

ecozzi@cray.com

htg@cray.com

Agenda – Image Management

- Image Management in Shasta
- Boot process in Shasta
- Package Repository Service (PRS)
- Image Management Service (IMS)
- Future work

© 2019 Cray Inc.

Image Management

- Prescriptive recipes create boot image artifacts used to boot nodes
- RESTful services for image management
  - Package Repository Service (PRS)
    - Define zypper/yum package repositories and provide the RPM content, at scale, for installing and updating software for every compute and non-compute node in the system
  - Image Management Service (IMS)
    - Build images from kiwi-ng recipes and customize images
    - Multiple Linux distributions supported
    - Uses kiwi-ng in a docker container
    - Uses Kubernetes Job workflow
  - Artifact Repository Service (ARS)
    - Store and retrieve artifacts (recipe, kernel, initrd, image root)
- Interact with these services using the REST API or Cray CLI

© 2019 Cray Inc.

Configuration Management

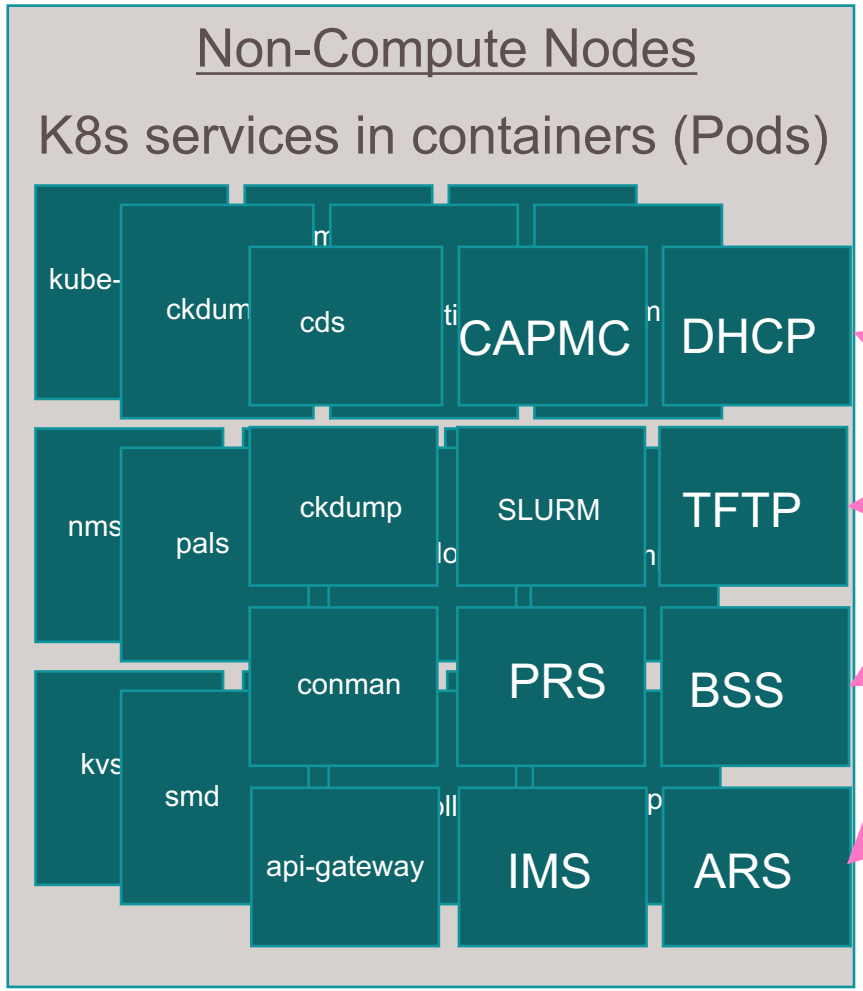
- Provides a configuration framework for Cray and customers which integrates industry-standard configuration management tooling (Ansible) with Cray services
- Flexible workflow
  - post-boot node personalization
  - post-boot re-configuration
  - pre-boot image customization
- Provides dynamic inventory plugins to target Cray nodes for config
- Provides versioned config data management which enables upgrade, rollback, and test

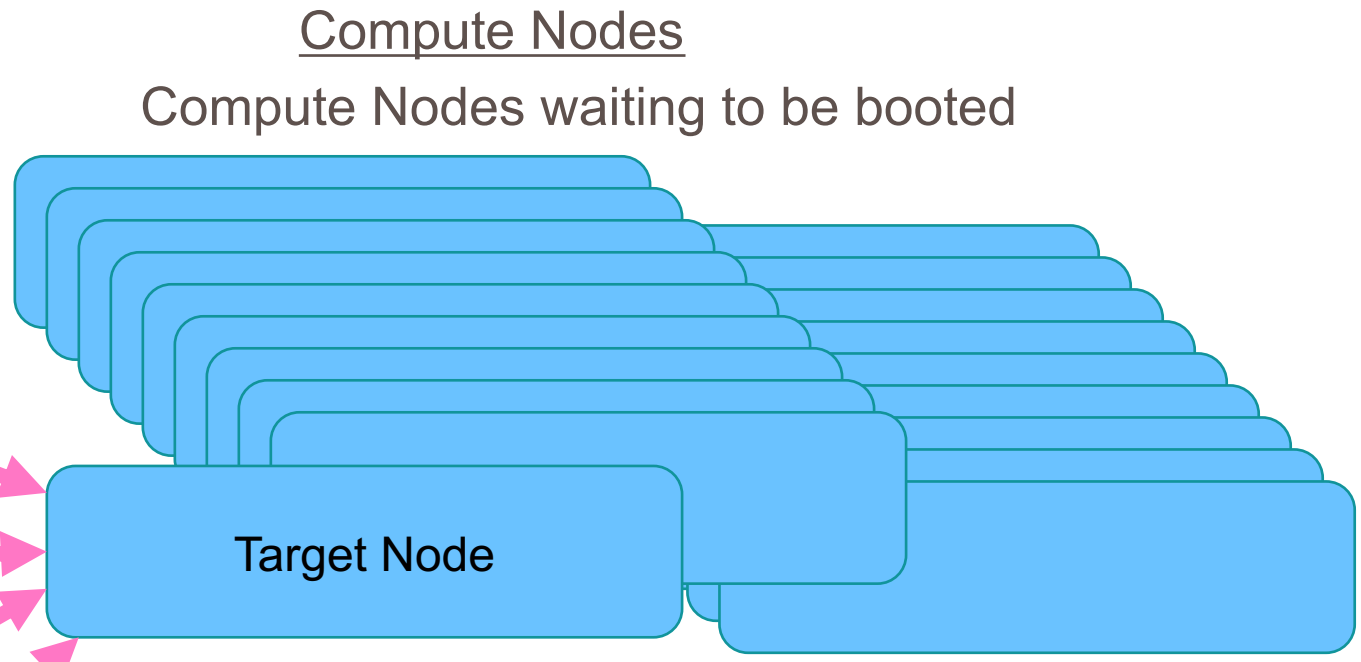
© 2019 Cray Inc.

Boot Process Flow

© 2019 Cray Inc.

Boot Process Flow Needs Image Artifacts





1) CAPMC powers up node
2) Node BIOS asks PXE driver on network card to send DHCP request
3) DHCP provides TFTP server address and file name
4) TFTP provides ipxe.efi file which points to BSS iPXE boot script
5) BSS iPXE boot script indicates what is needed to boot
    1) kernel (from ARS)
    2) initrd (from ARS)
    3) Kernel parameters (including the image root from ARS)

© 2019 Cray Inc.

Package Repository Service (PRS)

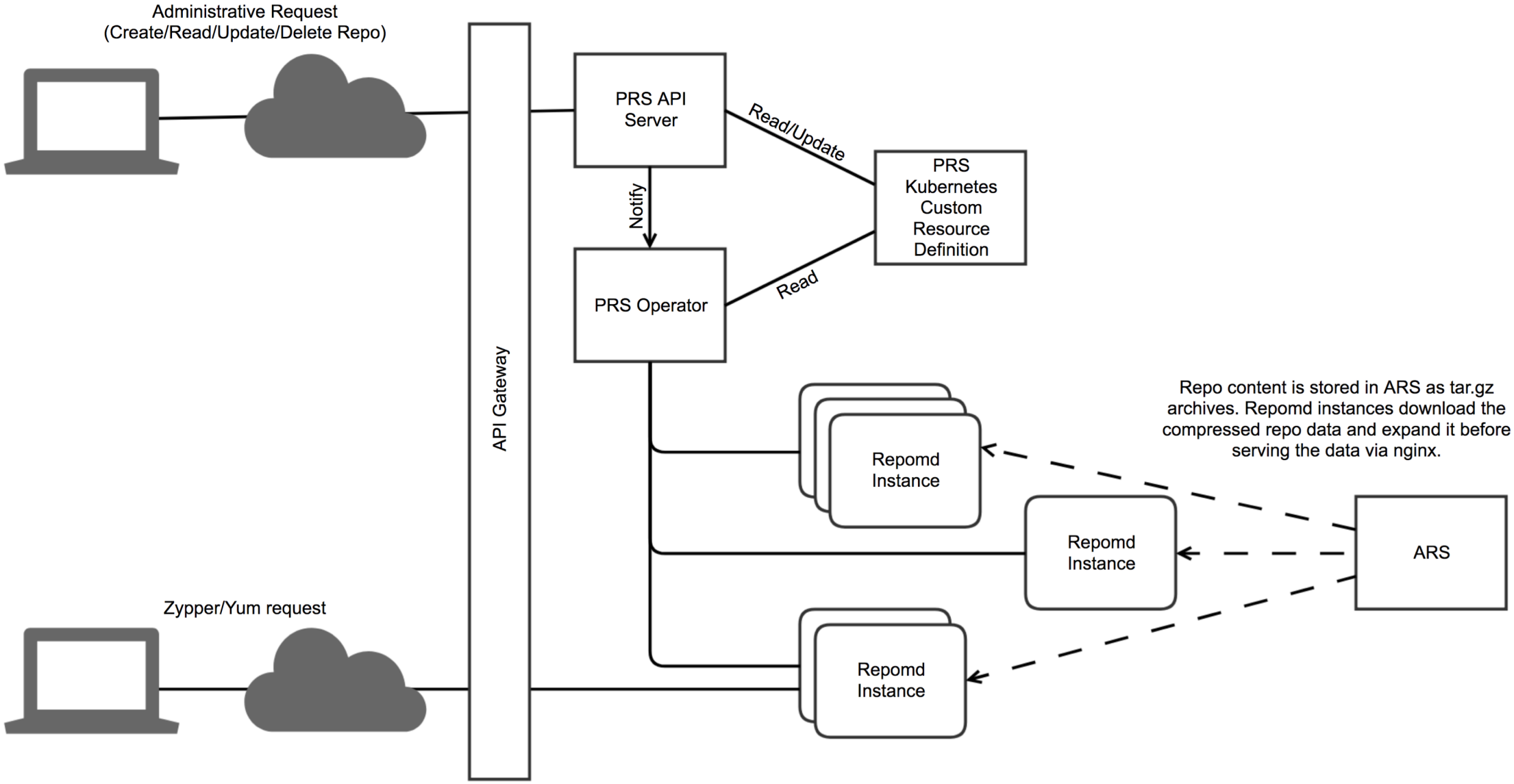
© 2019 Cray Inc.

Package Repository Service

- Package repositories are needed for:
  - Image creation
  - Image customization
  - Live Updates
    - Scaled solution for compute node access
- During the installation process several repomd package repositories are created to supply RPMs used by the system
  - Cray provided package repositories
    - Linux vendor software
      - SLES
      - CentOS
    - Cray software, such as drivers for the HSN
- Customers can use PRS to make additional package repositories available

© 2019 Cray Inc.

PRS Architecture



© 2019 Cray Inc.

PRS New Repository Demo

© 2019 Cray Inc.

PRS Create Repository Demo

- Process to create a new repository
  1. Put the RPMs in a directory
  2. Run createrepo to generate repomd index
  3. tar.gz the repository and upload to ARS
  4. Register the repository with PRS (watch “kubectl get pods” to see it)
  5. Access the repository (to confirm it was created)
  6. Register the repository with the OS
  7. Install a package (Live Update)

© 2019 Cray Inc.

Create Repository Contents

```
# REPONAME=myrepo
# mkdir "$REPONAME"
# cd "$REPONAME”
# cp $SOURCEDIR/python-httplib2-0.9.2-7.1.noarch.rpm .

# createrepo .
Spawning worker 0 with 1 pkgs
Workers Finished
Saving Primary metadata
Saving file lists metadata
Saving other metadata
# cd ..
# tar -cvzf "$REPONAME.tgz" -C "$REPONAME" .
./
./repodata/
./repodata/repomd.xml
./repodata/0c1f4b0edc5b749cda11c3ac13118aa12bfa88630634c84da84ff4f7fe4feeb9-other.xml.gz
./repodata/323d9f3d95cc8a1c3feb669e2654ed085208c2b18e1fdbda842edb06182e4af9-primary.xml.gz
./repodata/dbb2e33bf9484b546329365057 93e1419714489748 4498fbaf74eb37d7ed8965-filelists.xml.gz
./python-httplib2-0.9.2-7.1.noarch.rpm
```

© 2019 Cray Inc.

Load repository into ARS

```
# curl -s -H "Content-Type: application/json" \
>  -d '{"atype": "generic", "name": "'$REPONAME'", "version": "0.1"}' \
> https://api-gw-service-nmn.local/apis/ars/artifacts | tee out.json | python -mjson.tool
{
    "artifact_id": "05aa7d82-d160-48a1-939a-1889b4b7870b",
    "atype": "generic",
    "created": "2019-03-25T15:13:16.119884+00:00",
    "download_url": null,
    "name": "myrepo",
    "uri": null,
    "version": "0.1"
}
# ARTIFACT_ID=$(python -c 'import sys, json; print json.load(sys.stdin)["artifact_id"]' <out.json)
# curl -s -H "Content-Type: application/json" -d '{"artifact_id": "'$ARTIFACT_ID'"}' \
> https://api-gw-service-nmn.local/apis/ars/uploads | tee out.json | python -mjson.tool
{
    "artifact_id": "05aa7d82-d160-48a1-939a-1889b4b7870b",
    "created": "2019-03-25T15:13:16.175687+00:00",
    "state": "created",
    "upload_id": "87148aab-82d0-46ea-81bb-dacc126c0784",
    "upload_path": null,
    "url": null
}
# UPLOAD_ID=$(python -c 'import sys, json; print json.load(sys.stdin)["upload_id"]' <out.json)
# curl -XPUT -F "artifact=@$REPONAME.tgz" https://api-gw-service-nmn.local/apis/ars/uploads/$UPLOAD_ID
# curl -i -XDELETE https://api-gw-service-nmn.local/apis/ars/uploads/$UPLOAD_ID
```

Artifact ID in ARS for repository

Upload ID to ARS for repository

© 2019 Cray Inc.

Register Repository with PRS and Access It

```
# ADMIN_SECRET=$(kubectl get secrets admin-client-auth -ojsonpath='{.data.client-secret}' | base64 -d)
# function get_token {
>   curl -s -d grant_type=client_credentials -d client_id=admin-client -d client_secret=$ADMIN_SECRET https://api-gw-service-nmn.local/keycloak/realms/shasta/protocol/openid-connect/token | python -c 'import sys, json; print json.load(sys.stdin)["access_token"]'
> }
# curl -s -H "Authorization: Bearer $(get_token)" -H "Content-Type: application/json" \
>  -d '{"name": "'$REPONAME'", "artifactId": "'$ARTIFACT_ID'"}' \
>  https://api-gw-service-nmn.local/apis/prs/v1/repositories | python -mjson.tool
{
    "artifactId": "05aa7d82-d160-48a1-939a-1889b4b7870b",
    "links": [
        {
            "href": "https://api-gw-service-nmn.local/apis/prs/v1/repositories/myrepo",
            "rel": "self"
        },
        {
            "href": "https://api-gw-service-nmn.local/repositories/myrepo",
            "rel": "repository"
        }
    ],
    "name": "myrepo"
}
# REPO_URL="https://api-gw-service-nmn.local/repositories/$REPONAME"
# kubectl get pods -l cray-prs=deployment
NAME                                                READY   STATUS    RESTARTS   AGE
prs-repo-fb0675daa61f437aa08340c56e4d868b-d9d4684b5-zdchv   2/2     Running   0          18s
# curl "$REPO_URL/repodata/repomd.xml"
<?xml version="1.0" encoding="UTF-8"?>
...
</repomd>
```

Artifact ID in ARS for repository

URI for repository in PRS

© 2019 Cray Inc.

Register repository with OS & install packages

```
# zypper addrepo --check --refresh --name "$REPONAME" --no-gpgcheck "$REPO_URL" "$REPONAME"
Adding repository 'myrepo'
.....................................................................[done]
Warning: GPG checking is disabled in configuration of repository 'myrepo'. Integrity and
origin of packages cannot be verified.
Repository 'myrepo' successfully added
URI         : https://api-gw-service-nmn.local/repositories/myrepo
Enabled     : Yes
GPG Check   : No
Autorefresh : Yes
Priority    : 99 (default priority)
Repository priorities are without effect. All enabled repositories share the same priority.
# zypper install -y python-httplib2-0.9.2
Retrieving repository 'myrepo' metadata
.....................................................................[done]
Building repository 'myrepo' cache
.....................................................................[done]
Loading repository data...
Reading installed packages...
Resolving package dependencies...
The following NEW package is going to be installed:
  python-httplib2
...
```

URI for repository in PRS

© 2019 Cray Inc.

Image Management Service (IMS)

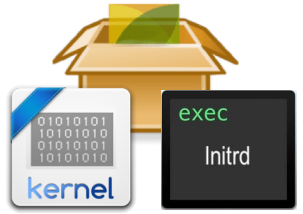
© 2019 Cray Inc.

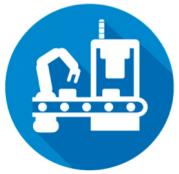
Image Management Service (IMS)

Allows administrators & authorized users to build or customize (pre-boot) images.

IMS Supports the following RESTful endpoints:

- Public Key management
  - Public keys to enable SSH access to debug and customize images
- Recipe management
  - Recipes that can be used to build an image
- Image & Image Artifact Management
  - An image can consist of multiple image artifacts including the image root, kernel and initrd
- Job Management
  - The workflow to create or customize an image via a Kubernetes Job





© 2019 Cray Inc.

kiwi-ng

- IMS uses the open source tool kiwi-ng to build images from kiwi-ng Image Descriptions
- kiwi-ng is the next-generation (or updated version) of the kiwi Appliance Builder
- An Appliance Builder?
  - An appliance is just a ready to use image for an operating system
  - Kiwi-ng can create images that boot via iPXE
- kiwi-ng supports building images of various Linux distributions
  - SLES 15
  - CentOS 7
  - Others

© 2019 Cray Inc.

kiwi-ng Terminology

- Image Description (Cray – “Recipe”)
  - XML specification to define an appliance
  - The image description is a collection of human readable files in a directory
  - The contents of the Image Description (“recipe”) is archived and stored in ARS with a link to the ARS artifact in IMS
- Image
  - The result of a kiwi-ng build process
  - Consists of the kernel, initrd, image root, and possibly other artifacts
    - IMS associates these image artifacts as a group
      - Even though each is stored in ARS as separate ARS artifacts

© 2019 Cray Inc.

Example kiwi-ng XML Specification

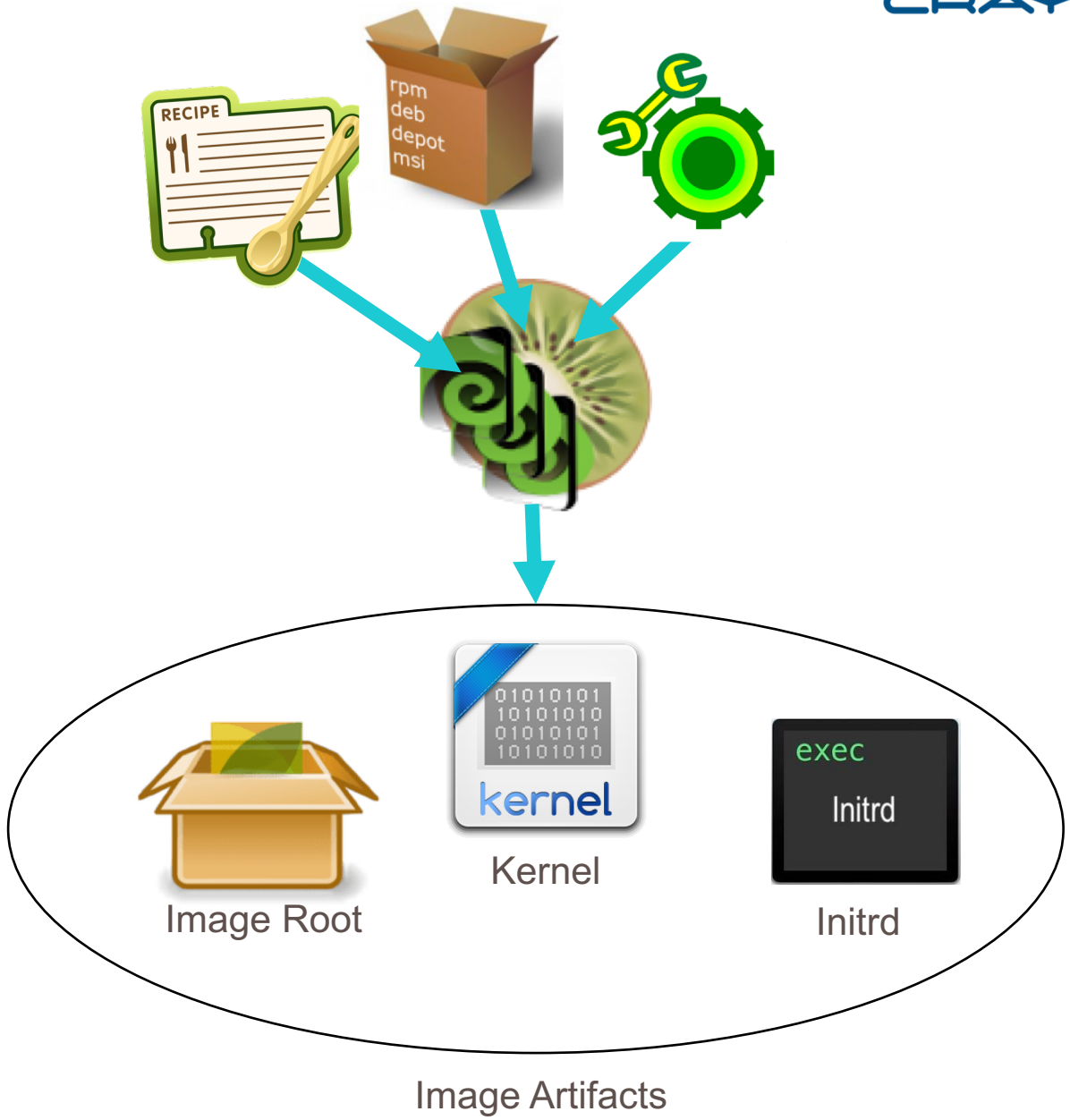
```
<?xml version="1.0" encoding="utf-8"?>
<image schemaversion="6.8" name="cray-sles15-barebones">
   …
   <description type="system">
      <author>Cray Inc</author>
      <contact>sps@cray.com</contact>
      <specification> Barebones SLES15 Recipe </specification>
   </description>
   …
   <users><user password="$1$3q2018$sPjSV98xEd0Nui0JGQXFh." home="/root" name="root" groups="root"/></users>
   <repository type="rpm-md" alias="SLES15_Module_Basesystem" imageinclude="true">
        <source path="http://api-gw-service-nmn.local/repositories/sle15-Module-Basesystem/"/>
    </repository>
    <repository type="rpm-md" alias="SLES15_Product_SLES" imageinclude="true">
        <source path="http://api-gw-service-nmn.local/repositories/sle15-Product-SLES/"/>
    </repository>
    <!-- for cray-squashfs-dracut -->
    <repository type="rpm-md" alias="DST_built_rpms" imageinclude="true">
        <source path="http://api-gw-service-nmn.local/repositories/sle15-cray-shasta/"/>
    </repository>
   <packages type="image">
      <package name="checkmedia"/>
      <package name="grub2-branding-SLE"/>
      <package name="iputils"/>
      …
   </packages>
   …
</image>
```

URIs for repositories in PRS

© 2019 Cray Inc.

Creating an Image

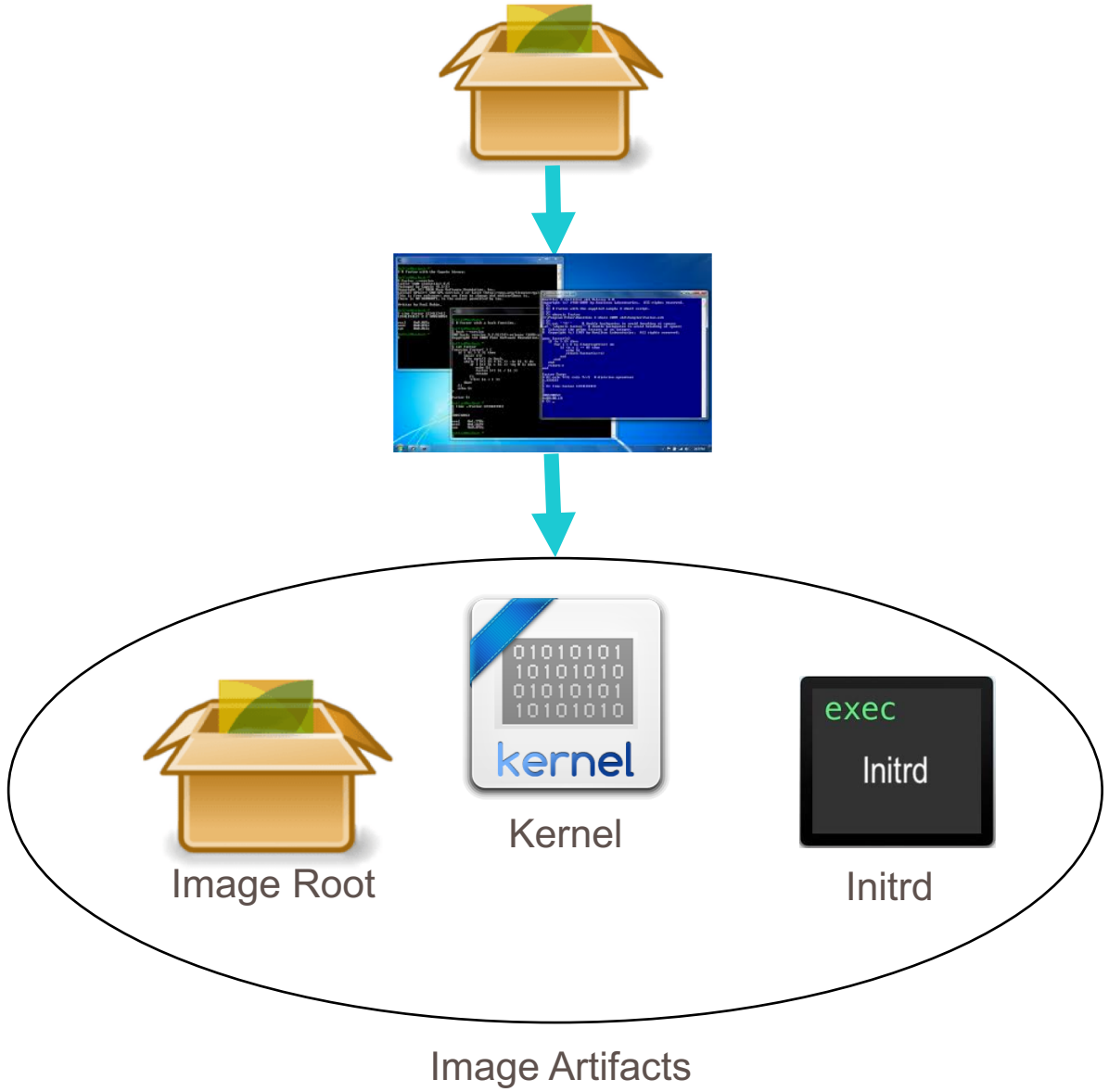
- Admin submits a “create job” to IMS
  - IMS establishes a new Kubernetes pod to build the image
  - The recipe is downloaded from ARS and passed to kiwi-ng running in the new pod
    - kiwi-ng installs the RPM packages listed in the recipe. RPMs are retrieved from repos setup by the Package Repository Service (PRS)
    - After rpms are installed, kiwi-ng runs scripts specified in the recipe on the image root
  - When kiwi-ng completes, the image artifacts are collected and stored in ARS



© 2019 Cray Inc.

Customizing an Image

- Admin submits a “customize job” to IMS
  - IMS establishes a new Kubernetes pod to customize the image
  - The existing image is downloaded from ARS and uncompressed
  - An SSH environment is established where the admin can access the image root and make any required changes
  - When the admin is done, the image artifacts are collected and stored in ARS as new artifacts



© 2019 Cray Inc.

Recipe Upload & SSH Key Demo

© 2019 Cray Inc.

Recipe Upload & SSH Key Demo

1. Upload image recipe archive to ARS
2. Associate ARS recipe artifact with IMS
3. Upload admin's public SSH Key to the IMS service
   - Could be special key only used for IMS work

© 2019 Cray Inc.

Upload Image Recipe Archive to ARS

```
# curl -X POST -H 'Content-Type: application/json' -d '{"atype": "generic", "name": "cray-sles15-barebones.tgz", \
>   "version": "0.1"}' https://$SMS_HOST/apis/ars/artifacts | python -m json.tool
{
    "artifact_id": "36693b15-abe7-49e6-a323-b42e309616b0",
    "atype": "generic",
    "created": "2019-03-26T15:29:18.556578+00:00",
    "download_url": null,
    "name": "cray-sles15-barebones.tgz",
    "uri": null,
    "version": "0.1"
}
# export POST_DATA=$(cat <<EOF
> { "artifact_id": "36693b15-abe7-49e6-a323-b42e309616b0" }
> EOF
> )
# curl -X POST -H 'Content-Type: application/json' -d "$POST_DATA" https://$SMS_HOST/apis/ars/uploads \
>   | python -m json.tool
{
    "artifact_id": "36693b15-abe7-49e6-a323-b42e309616b0",
    "created": "2019-03-26T15:29:52.521573+00:00",
    "state": "created",
    "upload_id": "70c0d42b-6b11-4cea-88b6-dfca03b0a600",
    "upload_path": null,
    "url": null
}
# curl -X PUT -F "artifact=@cray-sles15-barebones.tgz" \
>   https://$SMS_HOST/apis/ars/uploads/70c0d42b-6b11-4cea-88b6-dfca03b0a600
```

Artifact ID in ARS for recipe

Upload ID in ARS for recipe

© 2019 Cray Inc.

Associate ARS recipe artifact with IMS

```
# export RECIPE_TYPE=kiwi-ng
# export LINUX_DISTRIBUTION=sles15
# export RECIPE_NAME="cray-sles15-barebones"
# export ARS_ARTIFACT_ID=36693b15-abe7-49e6-a323-b42e309616b0

# export POST_DATA=$(cat <<EOF
> { "name": "$RECIPE_NAME",
>   "recipe_type": "$RECIPE_TYPE",
>   "linux_distribution": "$LINUX_DISTRIBUTION",
>   "artifact_id": "$ARS_ARTIFACT_ID" }
> EOF
> )

# curl -X POST -H 'Content-Type: application/json' \
>   -d "$POST_DATA" https://$SMS_HOST/apis/ims/recipes | \
>   python -m json.tool
{
    "artifact_id": "36693b15-abe7-49e6-a323-b42e309616b0",
    "created": "2019-03-26T15:31:17.208560+00:00",
    "id": "8e7f14e8-7465-496a-9111-e8797dd3d8f7",
    "linux_distribution": "sles15",
    "name": "cray-sles15-barebones",
    "recipe_type": "kiwi-ng"
}
```

Artifact ID in ARS for recipe

Artifact ID in ARS for recipe

Recipe ID in IMS

© 2019 Cray Inc.

Upload admin's SSH Key to the IMS service

```
# export PUB_KEY=$(cat ~/.ssh/id_rsa.pub)

# export POST_DATA=$(cat <<EOF
> { "name": "my public key",
>   "public_key": "$PUB_KEY" }
> EOF
> )

# curl -X POST -H "Content-Type: application/json" \
>    --data "$POST_DATA" https://$SMS_HOST/apis/ims/public-keys | python -m json.tool
{
    "created": "2019-03-26T15:32:19.927346+00:00",
    "id": "920fa63e-e326-4750-8ff9-7b7f102a8543",
    "name": "my public key",
    "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDE7ADUHS/WjMwTHBHqF4JoQCxCLXMxnIbYHl621Wgc3zAPr/nnP9TTn+3dEZGF
L4B88ZQXsJnZlPSgYGmAOczhWCvr9rcbc+KXAu7XU6LdWIrxMgzKuySnCZEso5LulxV4rU6pAI58JFafdwWX/t74dHIp
0K/K4V6Y39CLOoH5ZLls842G+JVFgjSfdnYSOFx/z1bYsXEtlIRNejZaPhTzy5teg919PUul0gUAGAXZWi2lZL67+qk+
DYjvJKz4LJa6sUTiNJJcLg0tp8ebKIl5Jm963VZydUZAOsrOwSnGUS6+gfJf8gWGJtlmutoqhzzEHe4LxrRhD0xB0xvC
5BfZ "
}
```

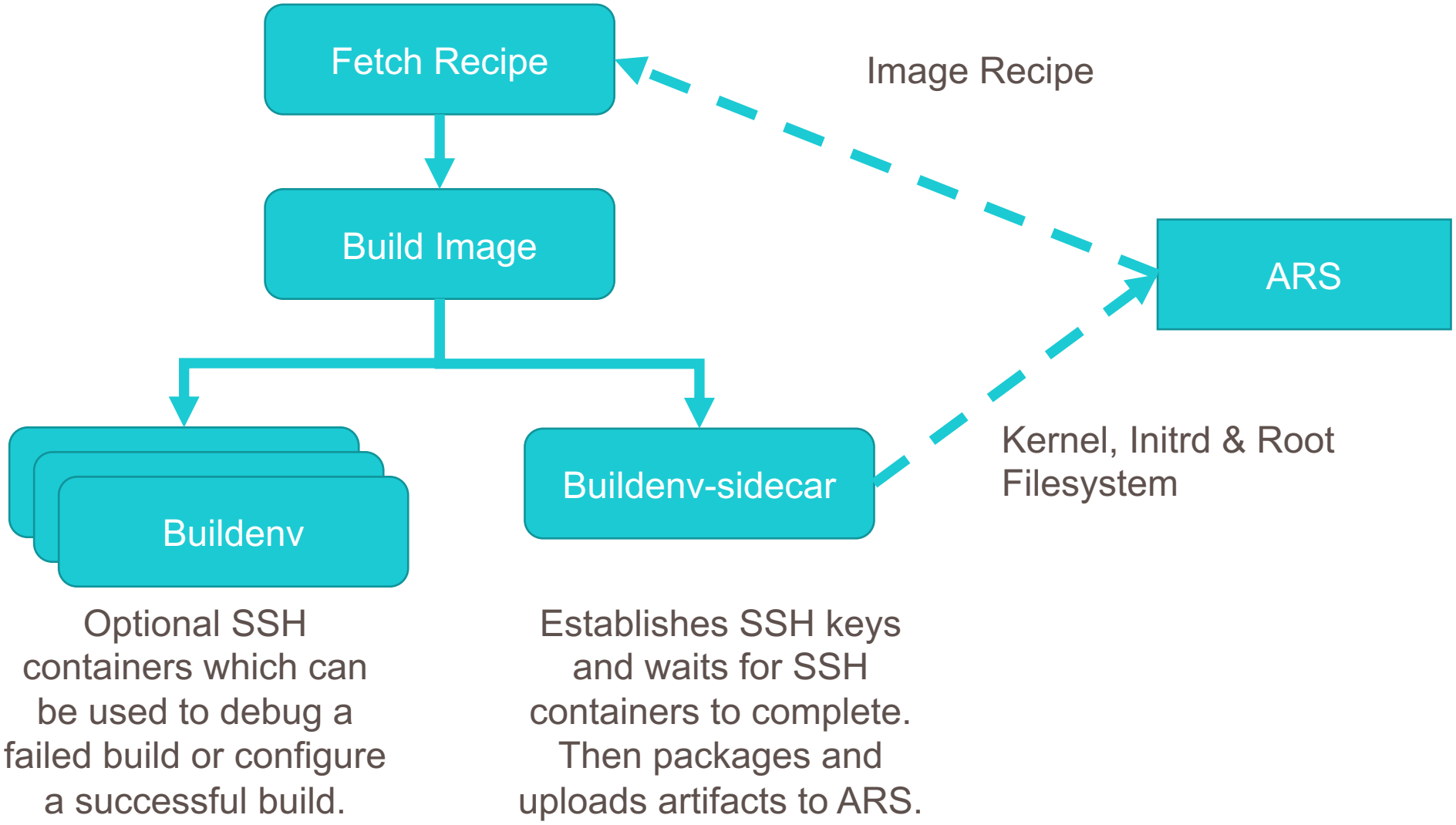
Public SSH Key ID in IMS

© 2019 Cray Inc.

Create Image Demo

© 2019 Cray Inc.

IMS – Create Image



© 2019 Cray Inc.

Create Image Demo

1. Submit a "create job" request to the IMS service
2. Describe Kubernetes job created by IMS
3. View logs from Kubernetes pod building our image
   - Fetching the recipe
   - Building the image
   - Uploading Artifacts

© 2019 Cray Inc.

Submit a "create job" Request to IMS

```
# export POST_DATA=$(cat <<EOF
> { "job_type": "create",
>   "artifact_id": "8e7f14e8-7465-496a-9111-e8797dd3d8f7",
>   "public_key_id": "920fa63e-e326-4750-8ff9-7b7f102a8543",
>   "enable_debug": "False",
>   "image_root_archive_name": "cray-sles15-barebones-image",
>   "kernel_file_name": "vmlinuz-4.12.14-23-default",
>   "initrd_file_name": "initramfs-cray.img" }
> EOF
> )
# curl -X POST -H 'Content-Type: application/json' -d "$POST_DATA" https://$SMS_HOST/apis/ims/jobs | python -m
json.tool
{
    "artifact_id": "8e7f14e8-7465-496a-9111-e8797dd3d8f7",
    "build_env_size": 10,
    "created": "2019-03-26T15:34:05.203819+00:00",
    "enable_debug": false,
    "id": "a176a7ef-7119-4196-b488-a1c94ca179c9",
    "image_root_archive_name": "cray-sles15-barebones-image",
    "initrd_file_name": "initramfs-cray.img",
    "job_type": "create",
    "kernel_file_name": "vmlinuz-4.12.14-23-default",
    "kubernetes_configmap": "cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-configmap",
    "kubernetes_job": "cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create",
    "kubernetes_service": "cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-service",
    "public_key_id": "920fa63e-e326-4750-8ff9-7b7f102a8543",
    "resultant_image_id": null,
    "ssh_port": 0,
    "status": "creating"
}
```

Recipe ID in IMS

Public SSH ID in IMS

© 2019 Cray Inc.

Describe Kubernetes Job Created by IMS

```
# kubectl describe job cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create
Name:           cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create
Namespace:      default
Selector:       controller-uid=97271ff5-4fdc-11e9-9e6f-fa163eecf85e
Labels:         app=ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create
                controller-uid=97271ff5-4fdc-11e9-9e6f-fa163eecf85e
                job-name=cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create
Annotations:    <none>
Parallelism:    1
Completions:    1
Start Time:     Tue, 26 Mar 2019 15:34:06 +0000
Pods Statuses:  1 Running / 0 Succeeded / 0 Failed
Pod Template:
  Labels:           app=ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create
                    controller-uid=97271ff5-4fdc-11e9-9e6f-fa163eecf85e
                    job-name=cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create
  Service Account:  ims-service-get-nodeport
  Init Containers:

...

Events:
  Type    Reason            Age   From            Message
  ----    ------            ----  ----            -------
  Normal  SuccessfulCreate  13s   job-controller  Created pod: cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create-jzxdm
```

© 2019 Cray Inc.

Logs from Fetching the Recipe

```
# kubectl logs -f cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create-jzxdm -c fetch-recipe
+ curl -f --cacert /etc/cray/ca/certificate_authority.crt https://api-
gateway.default.svc.cluster.local/apis/ars/assets/artifacts/generic/cray-sles15-barebones.tgz
+ tar zxv -C /mnt/recipe
...
./
./images.sh
./config.xml
./root/
./config.sh
./root/usr/
./root/.kiwi_grub_config.trigger
./root/etc/
./root/etc/udev/
./root/etc/dracut.conf.d/
./root/etc/sysconfig/
./root/etc/zypp/
./root/etc/motd
./root/etc/zypp/zypp.conf
./root/etc/sysconfig/network/
./root/etc/sysconfig/network/ifcfg-lan0
./root/etc/dracut.conf.d/02-kiwi.conf
./root/etc/dracut.conf.d/99-debug.conf
./root/etc/udev/rules.d/
./root/etc/udev/rules.d/70-persistent-net.rules
./root/usr/lib/
./root/usr/lib/systemd/
./root/usr/lib/systemd/system/
./root/usr/lib/systemd/system/grub_config.service
```

© 2019 Cray Inc.

Logs from Building the Image

```
# kubectl logs -f cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create-jzxdm -c build-image
+ kiwi-ng --logfile=/mnt/image/kiwi.log --type tbz system build --description /mnt/recipe --target
/mnt/image
[ INFO    ]: 15:34:13 | Loading XML description
[ INFO    ]: 15:34:13 | --> loaded /mnt/recipe/config.xml
[ INFO    ]: 15:34:13 | --> Selected build type: tbz
[ INFO    ]: 15:34:13 | Preparing new root system
[ INFO    ]: 15:34:13 | Setup root directory: /mnt/image/build/image-root
...
[ INFO    ]: 15:36:34 | Calling config.sh script
[ INFO    ]: 15:36:35 | Cleaning up SystemPrepare instance
[ WARNING ]: 15:36:35 | Path /mnt/image/build/image-root/sys not a mountpoint
[ WARNING ]: 15:36:35 | Path /mnt/image/build/image-root/proc not a mountpoint
[ WARNING ]: 15:36:35 | Failed to remove directory hierarchy /mnt/image/build/image-root/var/cache/kiwi:
[ INFO    ]: 15:36:35 | Calling images.sh script
[ INFO    ]: 15:37:37 | Creating system image
[ INFO    ]: 15:37:37 | Creating XZ compressed tar archive
[ INFO    ]: 15:39:42 | --> Creating archive checksum
[ INFO    ]: 15:40:14 | Export rpm packages metadata
[ INFO    ]: 15:40:15 | Export rpm verification metadata
[ INFO    ]: 15:40:42 | Result files:
[ INFO    ]: 15:40:42 | --> image_packages: /mnt/image/cray-sles15-barebones.x86_64-1.0.1.packages
[ INFO    ]: 15:40:42 | --> image_verified: /mnt/image/cray-sles15-barebones.x86_64-1.0.1.verified
[ INFO    ]: 15:40:42 | --> root_archive: /mnt/image/cray-sles15-barebones.x86_64-1.0.1.tar.xz
[ INFO    ]: 15:40:42 | --> root_archive_md5: /mnt/image/cray-sles15-barebones.x86_64-1.0.1.md5
+ rc=0
+ '[' 0 -ne 0 ']'
```

© 2019 Cray Inc.

Logs from Uploading Image Artifacts

```
# kubectl logs -f cray-ims-a176a7ef-7119-4196-b488-a1c94ca179c9-create-jzxdm -c buildenv-sidecar
Copying SMS CA Public Certificate to target image root
+ echo 'Copying SMS CA Public Certificate to target image root'
+ mkdir -p /mnt/image/build/image-root/etc/cray
+ cp -r /etc/cray/ca /mnt/image/build/image-root/etc/cray/
+ mksquashfs /mnt/image/build/image-root /mnt/image/cray-sles15-barebones-image.sqsh
Parallel mksquashfs: Using 4 processors
Creating 4.0 filesystem on /mnt/image/cray-sles15-barebones-image.sqsh, block size 131072.
[=============================================================/] 23479/23479 100%

Exportable Squashfs 4.0 filesystem, gzip compressed, data block size 131072
        compressed data, compressed metadata, compressed fragments, compressed xattrs
        duplicates are removed

...

+ python -m ims_python_helper image upload_artifacts cray-sles15-barebones-image a176a7ef-7119-4196-b488-a1c94ca179c9 -r /mnt/image/cray-sles15-barebones-image.sqsh -k /mnt/image/build/image-root/boot/vmlinuz-4.12.14-23-default -i /mnt/image/build/image-root/boot/initramfs-cray.img
```

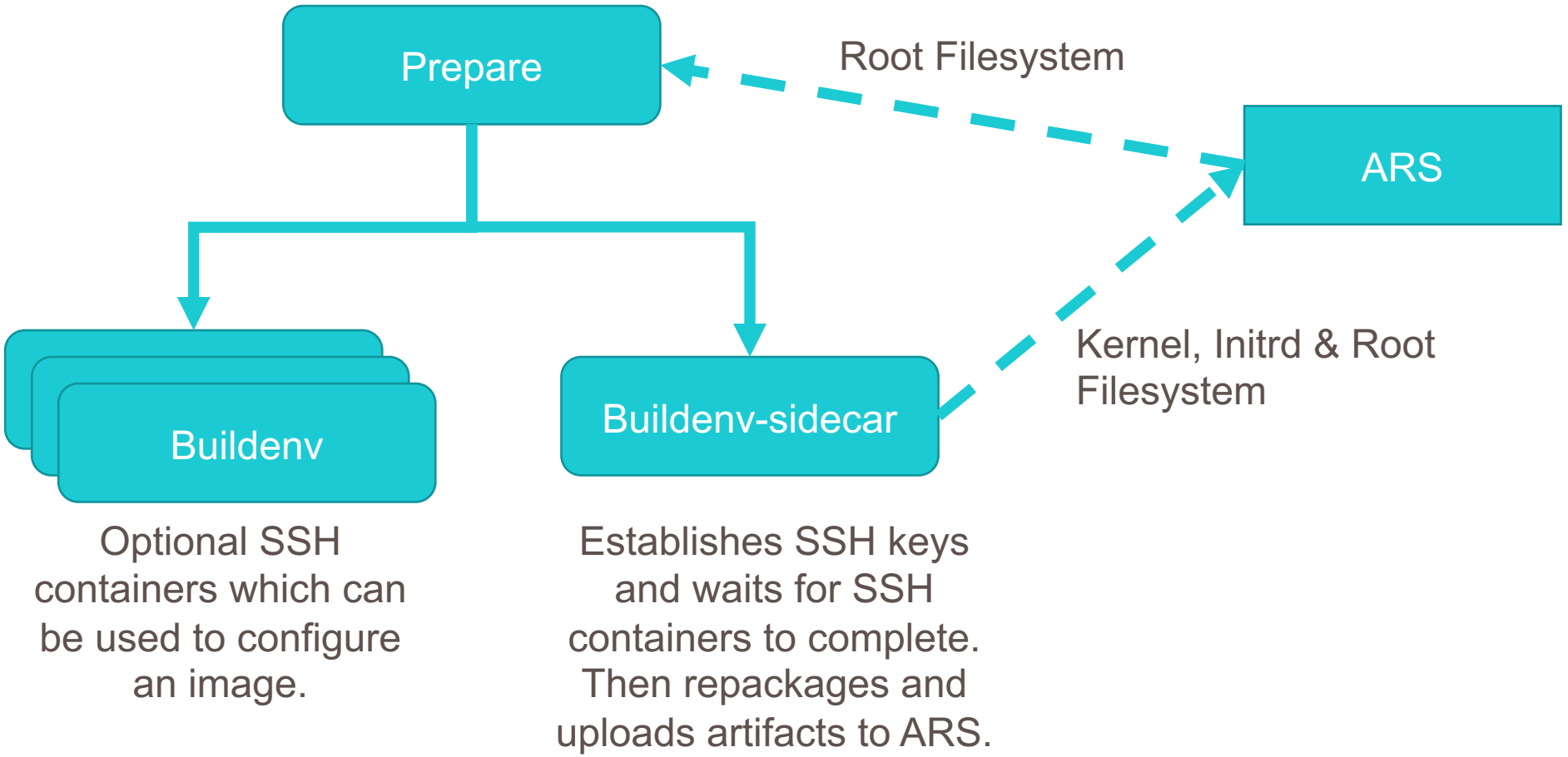
Artifact ID in ARS for new image root

© 2019 Cray Inc.

Customize Image Demo

© 2019 Cray Inc.

IMS – Customize Image



© 2019 Cray Inc.

Customize Image Demo

1. Submit "customize job" request to IMS
2. Describe Kubernetes job created by IMS
3. View logs from Kubernetes pod setting up customize environment
   - Fetch the image from ARS
   - Expand the image archive and make available via SSH environment
4. Admin accesses SSH environment and makes any modifications required
   - Admin signals when their changes are complete
5. View logs from Kubernetes pod customize and upload image artifacts

© 2019 Cray Inc.

Submit a "customize job" Request to IMS

Artifact ID in ARS for old image root

Public SSH ID in IMS

```
# export POST_DATA=$(cat <<EOF
> { "job_type": "customize",
>   "artifact_id": "a13fefa1-cf41-4672-8149-4f45f8a1b928",
>   "public_key_id": "920fa63e-e326-4750-8ff9-7b7f102a8543",
>   "enable_debug": "False",
>   "image_root_archive_name": "cray-sles15-barebones-image-customized",
>   "kernel_file_name": "vmlinuz-4.12.14-23-default",
>   "initrd_file_name": "initramfs-cray.img" }
> EOF
> )
# curl -X POST -H 'Content-Type: application/json' -d "$POST_DATA" https://$SMS_HOST/apis/ims/jobs
| python -m json.tool
{
    "artifact_id": "a13fefa1-cf41-4672-8149-4f45f8a1b928",
    "build_env_size": 10,
    "created": "2019-03-26T15:52:57.777381+00:00",
    "enable_debug": false,
    "id": "025a6bd8-ef0b-4f6e-8ab9-748c9060c232",
    "image_root_archive_name": "cray-sles15-barebones-image-customized",
    "initrd_file_name": "initramfs-cray.img",
    "job_type": "customize",
    "kernel_file_name": "vmlinuz-4.12.14-23-default",
    "kubernetes_configmap": "cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-configmap",
    "kubernetes_job": "cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize",
    "kubernetes_service": "cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-service",
    "public_key_id": "920fa63e-e326-4750-8ff9-7b7f102a8543",
    "resultant_image_id": null,
    "ssh_port": 0,
    "status": "creating"
}
```

© 2019 Cray Inc.

Describe Kubernetes Job Created by IMS

```
# kubectl describe job cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize
Name:           cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize
Namespace:      default
Selector:       controller-uid=3a7157b8-4fdf-11e9-8f45-fa163e0206aa
Labels:         app=ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize
                controller-uid=3a7157b8-4fdf-11e9-8f45-fa163e0206aa
                job-name=cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize
Annotations:    <none>
Parallelism:    1
Completions:    1
Start Time:     Tue, 26 Mar 2019 15:52:58 +0000
Pods Statuses:  1 Running / 0 Succeeded / 0 Failed
Pod Template:
  Labels:           app=ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize
                    controller-uid=3a7157b8-4fdf-11e9-8f45-fa163e0206aa
                    job-name=cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize
  Service Account:  ims-service-get-nodeport
  Init Containers:
...
Events:
  Type    Reason             Age   From             Message
  ----    ------             ----  ----             -------
  Normal  SuccessfulCreate   21s   job-controller   Created pod: cray-ims-025a6bd8-ef0b-4f6e-
8ab9-748c9060c232-customize-k5xwj
```

© 2019 Cray Inc.

Logs from Downloading Image & Setting Up SSH Environment

```
# kubectl logs -f cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj -c prepare
############################################################################## 100.0%
Parallel unsquashfs: Using 4 processors
25198 inodes (24589 blocks) to write
[=============================================================|] 24589/24589 100%
created 19418 files
created 3599 directories
created 1110 symlinks
created 0 devices
created 0 fifos

# kubectl logs -f cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj -c buildenv-sidecar
Running user shell for customize action
Image customization build environment is ready.
Use the following command to signal that image customization is complete in this container:
EXIT_COMMAND='touch /mnt/image/complete'

To access the build environment via ssh, connect to the Management plane using the following
configuration with the matching private key:

Use the following SSH PORT to connect to the image management environment.

PORT=25697

Example:
$ ssh -p 25697 [-i <private-key>] root@<sms-management-cluster>
```

© 2019 Cray Inc.

SSH to Customize the Image

```
# ssh -p 25697 root@sms.shasta.cray.com
root@cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj:/ # cd /mnt/image/image-root
root@cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj:image-root # ls
bin  boot  dev  etc  home  image  lib  lib64  mnt  opt  proc  root  run  sbin  selinux  srv  sys  tmp  us
r  var
root@cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj:image-root # cd etc
root@cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj:etc # echo "This file was modified by
ecozzi" >> motd
root@cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj:etc # cat motd
##############################################
# Welcome                                    #
##############################################

You have logged into a Cray Barebones image.

This node is running SLES 15 SuSE Linux System.

Please contact your IT system admin for any support requests.

This file was modified by ecozzi

root@cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj:etc # cd /mnt/image
root@cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj:image]# touch complete
Connection to sms.shasta.cray.com closed.
```

Signal end of customization

© 2019 Cray Inc.

Logs from Uploading Image Artifacts

```
# kubectl logs -f cray-ims-025a6bd8-ef0b-4f6e-8ab9-748c9060c232-customize-k5xwj -c buildenv-sidecar
Copying SMS CA Public Certificate to target image root
+ echo 'Copying SMS CA Public Certificate to target image root'
+ mkdir -p /mnt/image/build/image-root/etc/cray
+ cp -r /etc/cray/ca /mnt/image/build/image-root/etc/cray/
+ mksquashfs /mnt/image/build/image-root /mnt/image/cray-sles15-barebones-image.sqsh
Parallel mksquashfs: Using 4 processors
Creating 4.0 filesystem on /mnt/image/cray-sles15-barebones-image.sqsh, block size 131072.
[=============================================================/] 23479/23479 100%

Exportable Squashfs 4.0 filesystem, gzip compressed, data block size 131072
        compressed data, compressed metadata, compressed fragments, compressed xattrs
        duplicates are removed

...

+ python -m ims_python_helper image upload_artifacts cray-sles15-barebones-image-customized 025a6bd8-
ef0b-4f6e-8ab9-748c9060c232 -r /mnt/image/cray-sles15-barebones-image-customized.sqsh -k
/mnt/image/image-root/boot/vmlinuz-4.12.14-23-default -i /mnt/image/image-root/boot/initramfs-cray.img
```

Artifact ID in ARS for new image root

© 2019 Cray Inc.

Future Work

© 2019 Cray Inc.

Future Work

- Integration of IMS, PRS, and ARS into Cray CLI framework
- Provide CLI with higher level functions
  - One action for creating a recipe will do all calls to ARS and IMS
  - One action for creating a repository will do all calls to ARS and PRS
  - Etc.
- Live Updates
  - Automatic configuration so booted node can install rpms from PRS repomd instances
- Image customization (pre-boot configuration)
  - IMS workflow to build recipe into image automatically does configuration via Ansible
  - Configuration workflow can customize a pre-built image via Ansible

© 2019 Cray Inc.

SAFE HARBOR STATEMENT

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts.

These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.

© 2019 Cray Inc.

QUESTIONS?

ecozzi@cray.com
htg@cray.com

cray.com
@cray_inc
linkedin.com/company/cray-inc-/