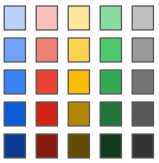
Google Cloud Platform

Infrastructure Automation with Cloud API

v 1.0

Infrastructure Automation



Agenda

1. Infrastructure Automation
2. Images
3. Metadata
4. Scripts
5. Cloud API
6. Lab
7. Review

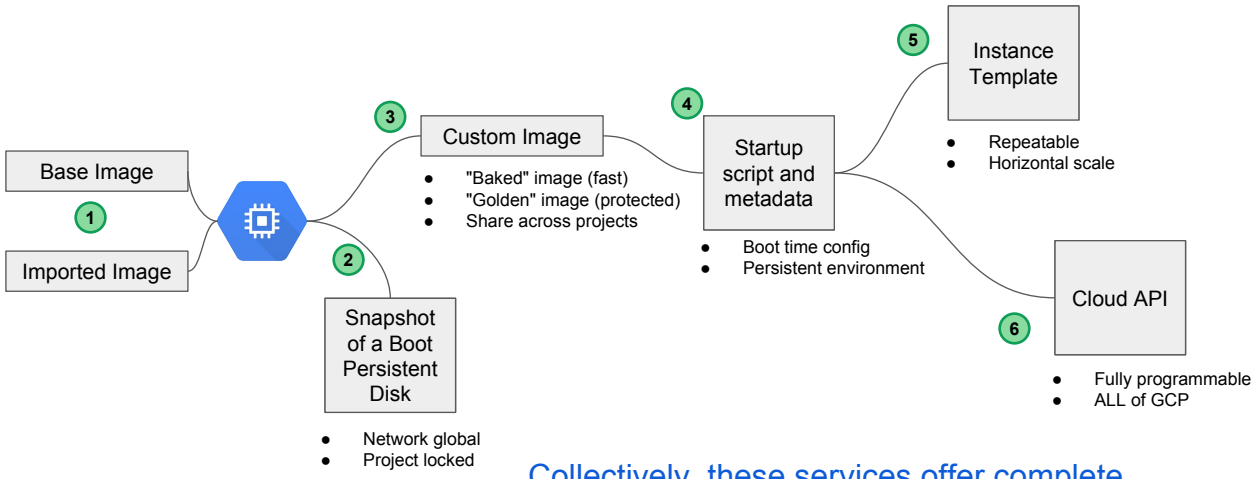
Reasons for Automating Infrastructure

- Repeatable re-deployable infrastructure
- Documented maintainable infrastructure
- Scalable solutions
- Huge architectures
- Complex systems

You know how to build infrastructure "by hand" with Console and Cloud Shell. This module and the next explore methods for automating the building of infrastructure, which is sometimes called "orchestration". The technologies covered are "force multipliers" for the methods you've already learned.

Listed above are some reasons for Automation. Other reasons include standards and policy compliance, support for internal or external audits, risk abatement, disaster recovery, survivability of the system due to loss of expertise, and many other reasons.

Infrastructure Automation Tools



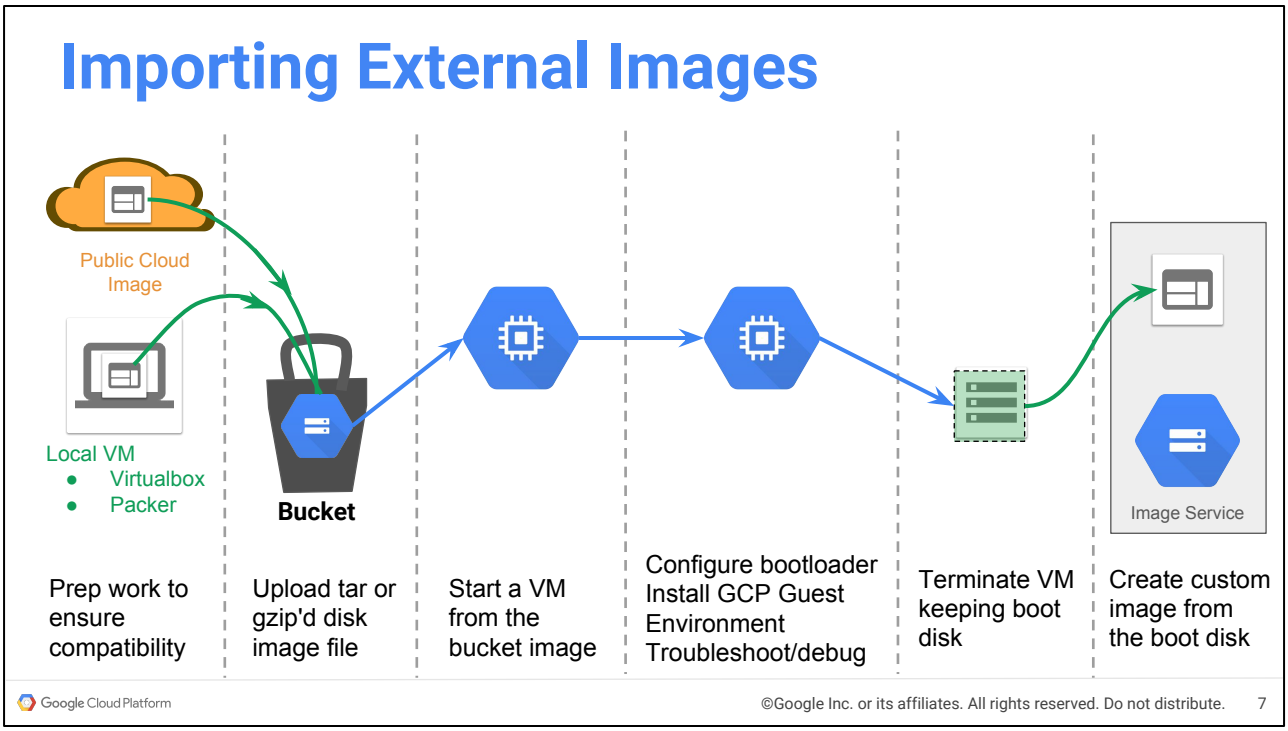
Collectively, these services offer complete control of the GCP infrastructure.

(1) You can start with a standard base image or you can import an image from another domain (on prem or another cloud). Google base images are maintained, updated, and patched. Imported images are useful for synchronizing with an existing system.
You can use the image to create a running VM. And you can connect to the VM and change its configuration, install software, and so forth.
(2) After the VM is tailored to your liking, you can create a Snapshot of its boot disk. Snapshots are global resources. You can use them to reconstitute a boot disk in any region in any network in your project. However, you can't share them between projects.
(3) A custom image can be shared between projects. And there are tools for managing those shared images with your image users. When we say "custom image" we generally mean that the OS and system settings are customized. A "baked" image is one with pre-installed and preconfigured software. Baked images are generally faster to become operational than other methods of installing software during or soon after boot. A "Golden image" is one with all the settings "just right", ready for sharing. Another reason to bake an application into an image is to lock the application so it is harder to make changes to its configuration.
(4) A startup script and metadata is one method to implement boot time customization and software installation. The benefit is being able to change configurations (including which software to install) on the fly. It is ideal for passing parameters that can only be known when the VM is being created and not before. The metadata provides a persistent environment that survives the termination of any individual VM. So you can use metadata to maintain system-level infrastructure data and state information.

(5) Any kind of image, startup scripts, and metadata can be used in an Instance Template. Instance Templates give you a system-documented repeatable way to make identical VMs. When used with a Managed Instance Group and an Autoscale they provide horizontal scaling.
(6) All of these tools and methods get you consistent, reliable, and automatic VM creation. But what they don't do is provide automation over the rest of the GCP infrastructure; they don't create load balancers, vpn connections, or networks. One solution is to install the Cloud SDK on a VM, authorize the VM to use parts of the Google Cloud APIs, and write programs that automate infrastructure creation. Remember that everything you've learned to do with the Console and the gcloud and gsutil commands was implemented by calling the Cloud API. You could write programs that call the Cloud API to programmatically create and manage infrastructure.

Agenda

1. Infrastructure Automation
2. Images
3. Metadata
4. Scripts
5. Cloud API
6. Lab
7. Review



Importing an external image:
"There is significant work involved. You might not want to do it every day."

Required prep work:
https://cloud.google.com/compute/docs/images/import-existing-image

GCP Guest Environment (Linux example)

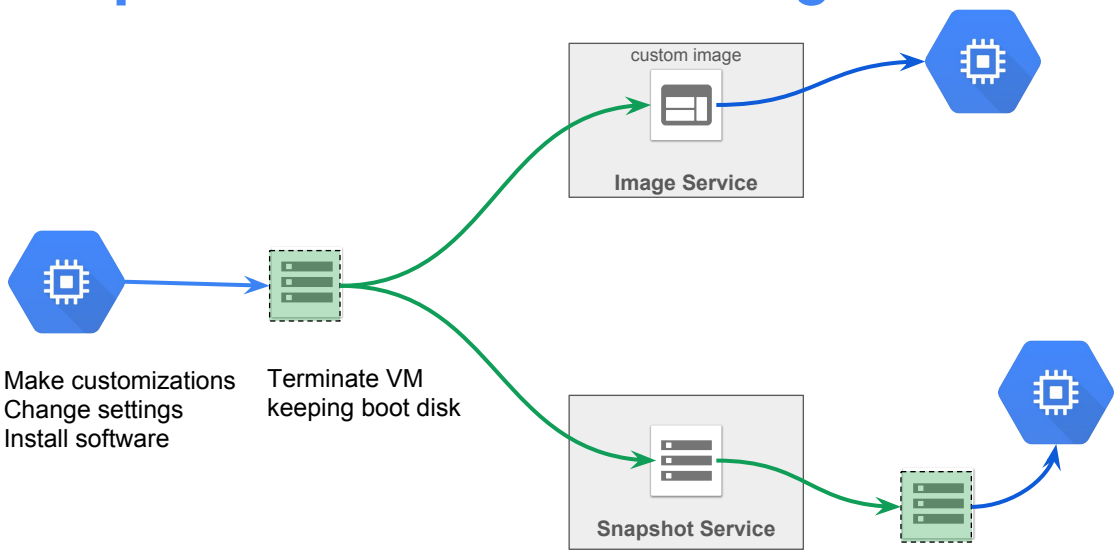
- Accounts daemon to setup and manage user accounts, and to enable SSH key based authentication.
- Clock skew daemon to keep the system clock in sync after VM start and stop events.
- Disk expand scripts to expand the VM root partition for boot disks with CentOS/RHEL 6 and 7 operating systems.
- Instance setup scripts to execute VM configuration scripts during boot.
- IP forwarding daemon that integrates network load balancing with forwarding rule changes into the guest.
- Metadata scripts to run user provided scripts at VM startup and shutdown.
- Network setup service to enable multiple network interfaces on boot.

https://cloud.google.com/compute/docs/images/import-existing-image

Migrating VMs to GCP: https://cloud.google.com/migrate/

Developing Packer images for GCP:
https://www.packer.io/docs/builders/googlecompute.html

Snapshots and Custom Images



To create a VM from a Snapshot backup of a Persistent Boot Disk, you need to first create a new disk from the snapshot. Then you can use that disk to boot a new VM instance. Persistent Disks are Zonal resources. However, Snapshots are Global resources. So you can use a snapshot to create a copy of the boot disk in any region, and also, in any network in your project. However, you cannot share the snapshot outside of your project. Note that when you create a new disk in a different region or network that egress charges apply to the data.

There are many benefits to creating an image from a VM's Persistent Boot Disk.

- A VM can be generated directly from the custom image without first manually restoring to a disk.
- The custom image can be used in Instance Templates
- The custom image can be shared across projects

https://cloud.google.com/compute/docs/instances/windows/creating-windows-os-image
On some versions of Windows you need to run a preparation program that updates drivers and some system services: GCESysprep

Managing custom images

- Share custom images using IAM roles
  - Share between projects: `--image-project` tag
  - Export to GCS bucket as tar file: `gcimagebundle`
- Image family
  - Points at latest version of an image
  - Reduces script/automation updates as versions change
- Deprecated -> Obsolete -> Deleted
  - Deprecated - warning that image is not supported and may end
  - Obsolete - existing users can continue to use it, but no new users
  - You can only delete custom images in projects you own

https://cloud.google.com/compute/docs/images/create-delete-deprecate-private-images

Deploy Docker Containers to VMs Alpha

- Alternative to a native GCP image
- Develop a Docker image and deploy it to container on VM
  - Containers abstract application code from OS-specifics
  - Containers reduce maintenance issues due to OS dependencies
  - Containers increase portability
- For applications that want the benefits of Docker Containers but the resource control of GCE / VMs
  - Both GKE and GAE Flexible Environment can run Docker containers, however both manage the resources for you

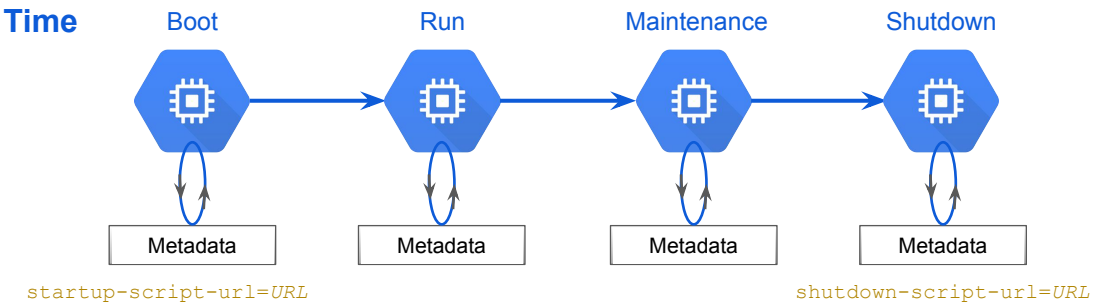
https://cloud.google.com/compute/docs/instance-groups/deploying-docker-containers

You would more commonly deploy Docker containers to GKE and allow Kubernetes/GKE to manage the scaling of the cluster and resources.
Docker containers can also be deployed onto the Google App Engine (GAE) Flexible environment. Containers are covered in detail in the Containers module.

Agenda

1. Infrastructure Automation
2. Images
3. Metadata
4. Scripts
5. Cloud API
6. Lab
7. Review

Metadata and Scripts



VMs have a special relationship with metadata, because the metadata is the standard place for persistent state information that survives the creation and termination of a VM.

During Boot, the Metadata tells the VM where to find the startup script.
When that Startup script is running, it can access additional metadata and get current information about its environment.

Also, scripts and applications can access metadata while the VM is running. And an application can watch a metadata object and receive notification when that object changes. So it can be used to coordinate actions between VMs in a system. (Note that Cloud Pub/Sub might be the better tool for this).

When a maintenance event is about to occur, such as a live migration, there is a (not guaranteed) 30 to 60 second warning. An application can register to watch a special value to get notification when the VM that the application is running on is about to undergo a live migration event. Then the application can take appropriate action.

Finally, when a VM is shutting down, it uses Metadata to find the shutdown script. In Managed Instance Groups the shutdown script might drain the connection from the Load Balancer to the VM to minimize user impact to the loss of the VM. Other shutdown activities: write logs, save state.

Metadata

- project metadata - visible to all VMs in project
- VM metadata - private to the VM
  - Instance hostname
  - SSH keys
  - Project ID
  - Service account information and token
- key:value pairs
  - directories containing key:value pairs
  - Can return: specific value for a key, all keys in a directory, or a recursive list of keys

https://cloud.google.com/compute/docs/storing-retrieving-metadata#default

Query metadata

- Console, CloudShell, or API
- From VM, use wget or curl:
  - http://metadata.google.internal/computeMetadata/v1/
  - http://metadata/computeMetadata/v1/
  - http://<ip-address>/computeMetadata/v1/
- Using `gcloud`
  - `gcloud compute instances create <INSTANCE> \ --metadata servertype=client color=blue`
  - `gcloud compute project-info add-metadata \ --project <PROJECT> --metadata foo=bar baz=bat`

https://cloud.google.com/compute/docs/storing-retrieving-metadata#querying

Detect when metadata has changed

- Detect `using wait_for_change` parameter
  - ```
    curl \
    "http://metadata/computeMetadata/v1/instance/\
    tags?wait_for_change=true" \
    -H "Metadata-Flavor: Google"
    ```
- `timeout_sec` - returns if value changed after n seconds
- Entity tags - HTTP ETag *(used for webcache validation)*
  - If metadata ETag differs from local version then the latest value is returned immediately
  - Instances can use ETags to validate if they have the latest value

Use hanging GET request to wait for metadata changes and programmatically react by including the `wait_for_chage` parameter.

ETags do not work on directory listings or service account tokens.
    ```

Handle maintenance event

- Retrieve scheduling options (availability policies)
  - `/scheduling` directory
  - `maintenance-event` attribute
  - Notifies when a maintenance event is *about* to occur
  - Value changes 60 seconds before a transparent maintenance event starts
- Query periodically to trigger application code prior to a transparent maintenance event
  - Example: backup, logging, save state

https://cloud.google.com/compute/docs/storing-retrieving-metadata#maintenanceevents

Agenda

1. Infrastructure Automation
2. Images
3. Metadata
4. Scripts
5. Cloud API
6. Lab
7. Review

Startup and shutdown scripts

- Startup scripts
  - Linux
    - Runs during VM boot, last step in the boot process
    - Any language valid on the VM, but bash is most common
  - Windows
    - Runs before boot or after boot
    - cmd or bat, but Powershell is most common
- Shutdown scripts
  - Best effort run triggered by terminate or restart
- Both startup and shutdown scripts are commonly specified when the VM is created, but can also be added after

https://cloud.google.com/compute/docs/startupscript

Linux startup scripts

- Script typically specified at instance creation
  - Runs at boot time; can be rerun after boot
  - `gcloud compute instances create <instance> --metadata startup-script-url=<url>`
  - `startup-script-url` metadata key for script in GCS bucket
    - Service account needs permission and scope to read
  - Upload directly to metadata using `startup-script` key
  - Upload a script from a local file
- Logs accessible on instance, can be used for troubleshooting: `/var/log/startupscript.log`
-

Windows startup scripts

- Scripts can be hosted in metadata
- Windows Server requires scripts in GCS to be public

	Runs during sysprep, before boot	Runs after sysprep completes and on every subsequent boot
cmd	`sysprep-specialize-script-cmd`	`windows-startup-script-cmd`
bat	`sysprep-specialize-script-bat`	`windows-startup-script-bat`
Powershell	`sysprep-specialize-script-ps1`	`windows-startup-script-ps1`

Startup Scripts

- Startup scripts
  - Can be rerun after boot
- Common use cases include:
  - Software installation
  - Operating system updates
  - Turn on services

Notes:

For more information on startup scripts, see: https://cloud.google.com/compute/docs/startupscript.

Shutdown scripts

- Best effort run on terminate or restart
- `~90 seconds on most machine-types`
- `~30 seconds on preemptible VMs`
- `gcloud compute instances create <instance> --metadata shutdown-script-url=<url>`

https://cloud.google.com/compute/docs/shutdownscript

Agenda

1. Infrastructure Automation
2. Images
3. Metadata
4. Scripts
5. Cloud API
6. Lab
7. Review

Cloud SDK

- Cloud Client Libraries
  - Go, Java, Python, Node.js, PHP, Ruby, C#
- Installation
  - Download: https://cloud.google.com/sdk/downloads
  - Extract
  - Setup paths and reporting: `./google-cloud-sdk/install.sh (or .bat)`
  - Initialize the SDK: `gcloud init`
- Authorization
  - `gcloud auth activate-service-account --key-file [KEY_FILE]`

Google API Libraries: https://developers.google.com/products/
Technically, the Cloud SDK is one library among the *many* libraries associated with Google products, including Maps, gMail, Drive, and so forth.

https://cloud.google.com/sdk/docs/initializing

Cloud SDK Components

- List components
  - `gcloud components list`
- Add components
  - `gcloud components install [COMPONENT]`
    - Developers: Cloud Pub/Sub and Cloud Datastore Emulators
- Update components
  - `gcloud components update`
- Remove components
  - `gcloud components remove [COMPONENT]`
    - Command line tools - you may want to remove them for security

The command line tools are components. So are all the Beta commands. For developers, there is a Cloud Pub/Sub Emulator and a Cloud Datastore Emulator.

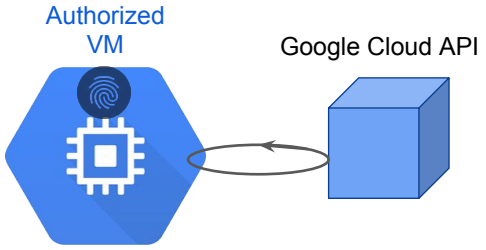
Command line tool integration

gcloud	gsutil	bq	kubectl
Compute Engine	Cloud Storage	Cloud Logging	
Container Engine	Big Query	Deployment Manager	
Cloud DNS	Cloud SQL	Resource Manager	
Cloud IAM	Cloud Dataproc	Source Repositories	

Windows Powershell `cmdlets`

This is the top of the gcloud reference documentation. Most functions are in the gcloud tool. Only, storage (gsutil), Big Query (bq), and Kubernetes (kubectl) are separate.
https://cloud.google.com/sdk/gcloud/reference/

VM as an Automation Engine



- A VM is authorized with a service account to use the Google Cloud API
- Software on the VM uses the API to create infrastructure on your behalf

This isn't a coding class. Developer Track classes cover application development including interaction with the Cloud API.

However, using a VM to create infrastructure is a very powerful tool.

So… without having you code… you will be learning how to setup and operate such an infrastructure automation tool in the lab.

Agenda

1. Infrastructure Automation
2. Images
3. Metadata
4. Scripts
5. Cloud API
6. Lab
7. Review

LAB #1 Automate with Cloud API

In this lab you will experiment with using the Google Cloud SDK for infrastructure automation. You will actually create a VM and authorize it to use the Google Cloud API to create VMs and install software on them.

You won't actually write the code. You will use an open source project that has the code already written.

Lab #1 Overview

In this lab you will open access to the **Google Cloud API** to software on a VM and use it to automate infrastructure creation. You will do everything *except* write the automation code.

You will customize and authorize a VM to use the Google Cloud API. You will use open source software to create a Hadoop "clustermaker". And you'll take a snapshot of the boot persistent disk that will enable you to migrate the solution to other networks.

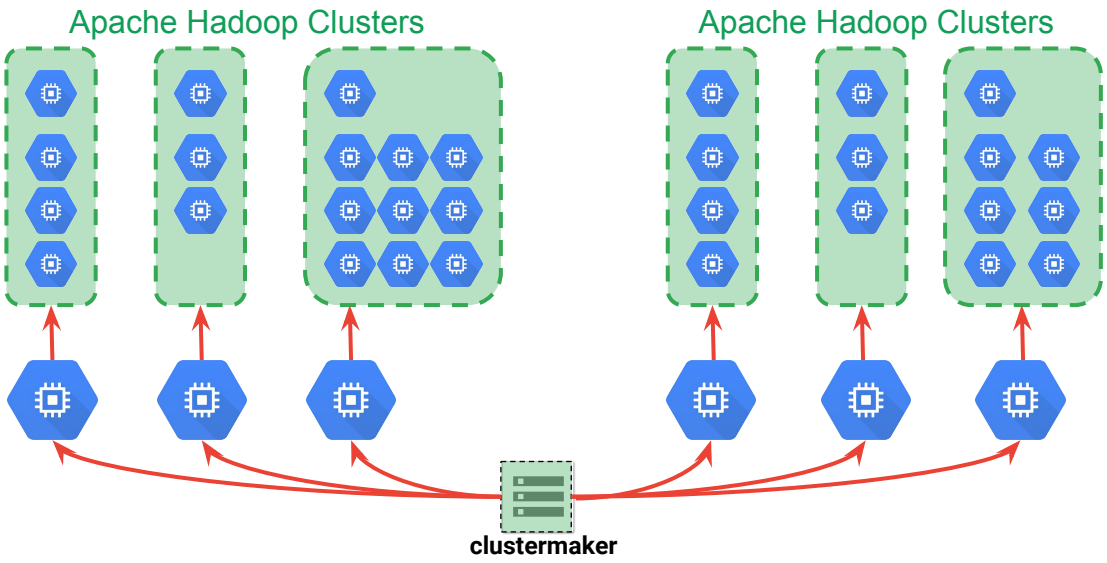
Lab#1: Review

- In this lab you created a customized VM.
- You authorized it to use the Google Cloud API for automation. You used open source software that uses the Google Cloud API to provision Hadoop clusters.
- You took a snapshot of the boot persistent disk, and used that snapshot to create a new VM in a different network.
- By the end of this lab, you were able to deploy Apache Hadoop clusters of any size in any region within GCP in minutes.

This purpose of this lab is to give you experience and develop knowledge and skills with a number of infrastructure features of GCP that relate to VMs.

This VM is performing infrastructure automation similar to what might be done with a 3rd party orchestration tool such as Ansible, Chef, Puppet, or Terraform.
You might use Cloud Dataproc for Hadoop service rather than something like this.

Lab#1: Hadoop IaaS Solution



Agenda

1. Infrastructure Automation
2. Images
3. Metadata
4. Scripts
5. Cloud API
6. Lab
7. Review

More...

- Startup scripts
  - https://cloud.google.com/compute/docs/startupscript
- Shutdown scripts
  - https://cloud.google.com/compute/docs/shutdownscript
- Storing and retrieving metadata
  - https://cloud.google.com/compute/docs/storing-retrieving-metadata


cloud.google.com