
AWS
re:Invent

CMP214-R

EC2 Image Builder: Building virtual machine images made easy

Samartha Chandrashekar
Senior Product Manager
Amazon Web Services

AWS re:Invent

aws

Agenda

Importance of “golden” VM images

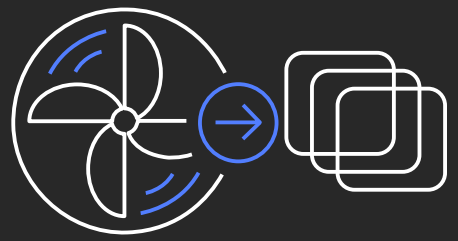
Pain points of building “golden” VM images

How EC2 Image Builder helps

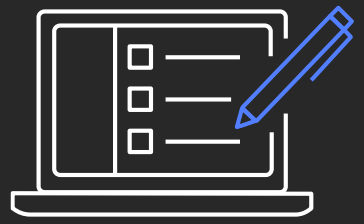
Usage workflows

Best practices

Golden VM images



Template server image. Saves time & ensure consistency



Pre-installed & pre-configured with custom software & settings

Hardened to meet IT standards

Nearly every organization has to build golden VM images

However…

Producing “golden” VM images is challenging due to the high-skill bar to build automation

Customers using existing automation/scripting tools need to host & maintain deployments

Many manually build and test new images every time software updates are available

Enterprises encounter service disruptions from not catching issues before production use

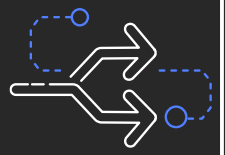
Customers asked for a one-stop shop to build golden images

Customers have asked to be able to…

Quickly and easily build automation to create golden images without writing code

Easily test images with AWS-provided and custom tests before deploying to production

Secure images with AWS-provided & custom settings to meet internal/industry standards



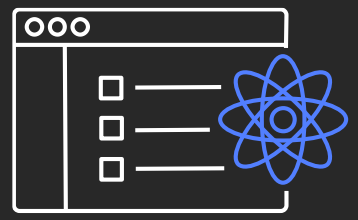
Distribute and share images easily across accounts & regions with centralized enforcement

Build images for use on AWS and on-premises

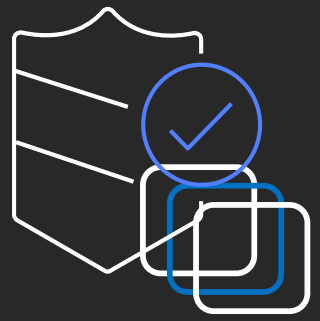
Introducing EC2 Image Builder…

EC2 Image Builder – benefits

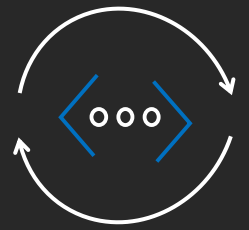
Quickly and easily automate the creation, management, and deployment of up-to-date and compliant "golden" VM images



Generate automation to build VM images with a GUI

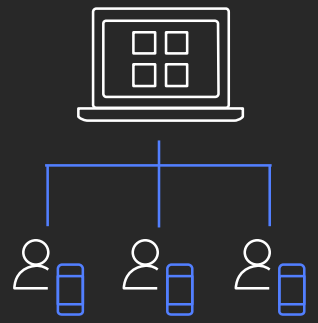


Reduce cost of building secure, compliant & up-to-date images

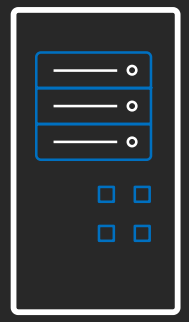


Improve service uptime by testing images before use in production

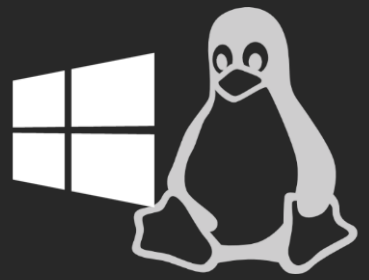
EC2 Image Builder – benefits



Enforce policies on VM image usage across AWS accounts

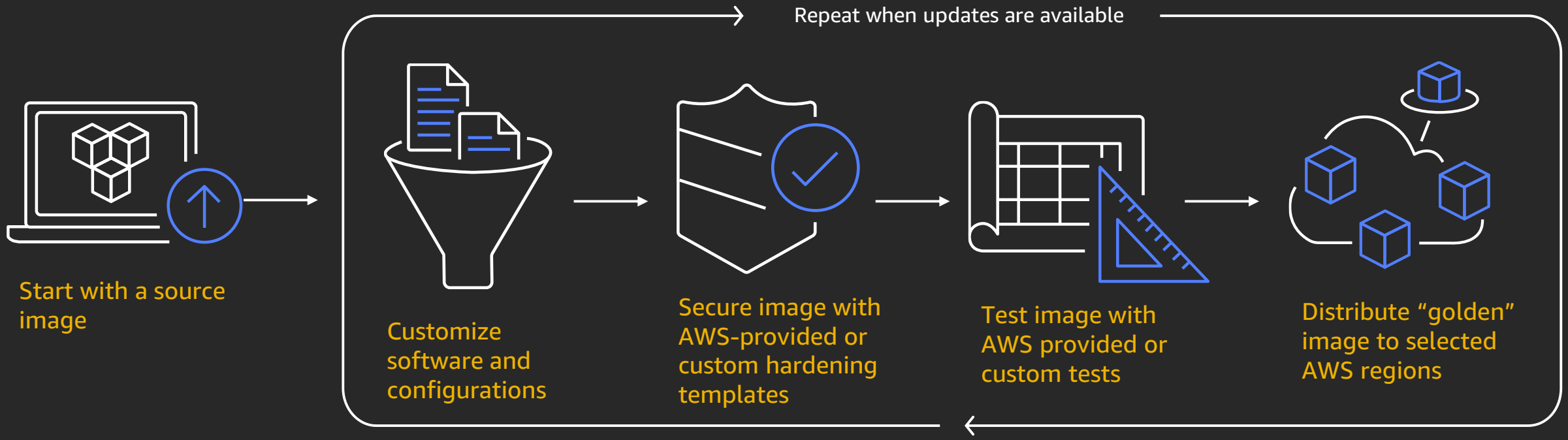


Build golden VM images for use on AWS and on-premises



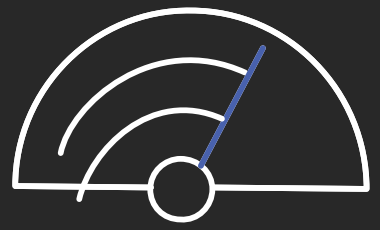
Works for both Windows and Linux

EC2 Image Builder – how it works

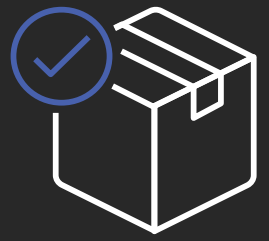


All EC2 Image Builder operations run in your AWS account

AWS-provided build components and tests

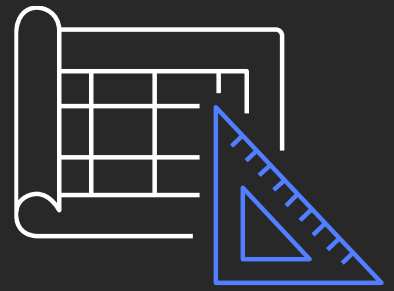


Get started quickly with AWS provided components



Commonly used software provided as “managed components”

Security hardening primitives for STIG available as inbox components

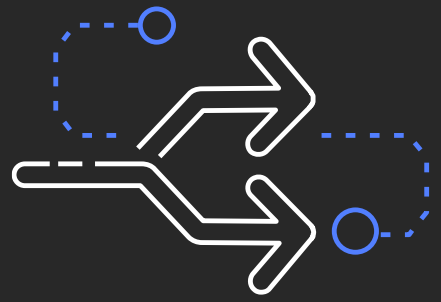


AWS provides tests to validate your images after they are built

Declarative document to define custom components

Phases

build & validate, test



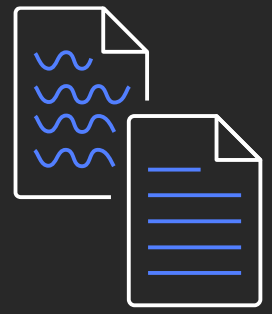
Each step in each phase uses action modules & returns an exit code

ExecuteBinary, ExecuteBash, ExecutePowerShell

Reboot, UpdateOS

S3Upload, S3Download

SetRegistry (Windows-only)

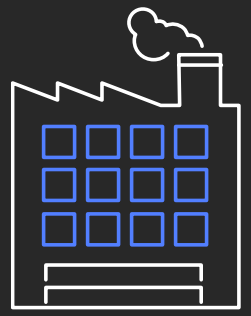


Files for each execution

detailedOutput.json
document.yaml
console.log
application.log

Integration with AWS License Manager

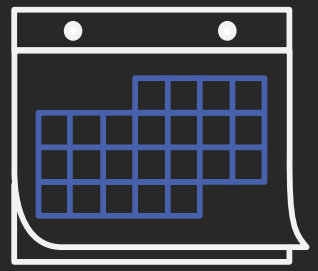
Associate license configurations to AMIs



Use AWS License Manger to centrally mange licenses and their usage

Create and mange license configurations with AWS License Manager

Keeping images patched and up-to-date



Rebuild images with latest changes on a schedule

You can also manually trigger rebuilds

Include latest updates to the OS, build components, and tests based on semver based triggers

Shared responsibility for security

Ensuring the security of custom AMIs is a shared responsibility between AWS and the customer

You own the security posture of images produced

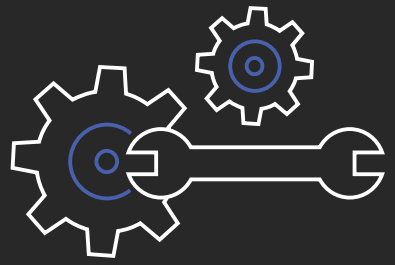
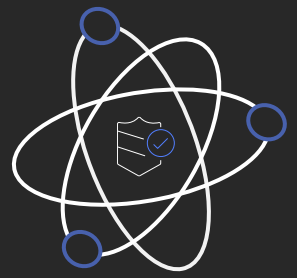


Image Builder enforces cleanup after building images (Linux: cleanup script run; Windows: Sysprep)



We recommend that you validate the security posture (AWS Inspector can help)

AWS does not guarantee images produced with Image Builder to be compliant with regulatory criteria

Identity and Access Management

Service Linked Role (SLR)

Grants permissions to EC2 Image Builder on your behalf. When you create your first Image Builder resource, an SLR is created for you.

IAM permissions to Instance role

IAM Role attached to the EC2 instance used to install build components, run tests, and write troubleshooting logs to Amazon S3 needs requisite permissions

Just enough permissions

EC2 Image Builder concepts

Pipeline

Recipe

Build component

Test

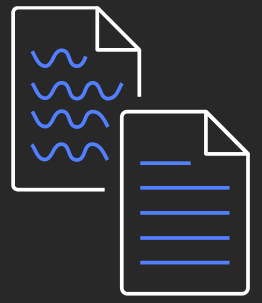
Build schedule

Managed Image

EC2 Image Builder workflow & demonstration

Debugging and troubleshooting

EC2 Image Builder tracks and displays progress for each step



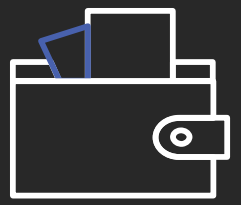
Logs exported to Amazon S3

Run arbitrary commands & scripts with AWS Systems Manager RunCommand

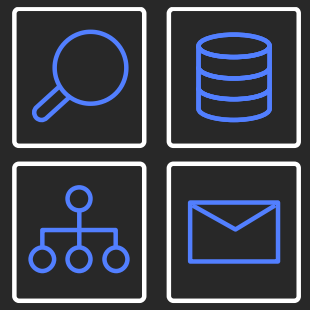
Example of a log file for debugging

```
2019-11-29 23:31:57 Info Document TOE_2019-11-29_23-31-57_UTC-0_6f164c8e-1300-11ea-83b0-0e7e644ebf61/windows-throw-
exception_123794003928538027489912422 5.ps1
2019-11-29 23:31:57 Info Phase build
2019-11-29 23:31:57 Info Step HappyStep
2019-11-29 23:31:59 Info Command execution completed successfully
2019-11-29 23:31:59 Info Stdout: Directory: C:\
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----       11/14/2018   6:43 AM                EFI
d-----       11/29/2019  11:31 PM                imagebuilder
d-----        9/15/2018   7:12 AM                PerfLogs
d-r---       11/14/2018   4:08 PM                Program Files
d-----       11/29/2019  11:28 PM                TOE_2019-11-29_23-28-10_UTC-0_e7ca00c4-12ff-11ea-a640-0e7e644ebf61
d-----       11/29/2019  11:31 PM                TOE_2019-11-29_23-31-57_UTC-0_6f164c8e-1300-11ea-83b0-0e7e644ebf61

2019-11-29 23:31:59 Info Stderr:
2019-11-29 23:31:59 Info ExitCode 0
2019-11-29 23:31:59 Info Step UnhappyStep
2019-11-29 23:32:02 Info Command execution resulted in an error
2019-11-29 23:32:02 Info Stdout: This is standard output from Write-Host
2019-11-29 23:32:02 Info Stderr: This is an exception from a PowerShell throw command
At C:\Windows\TEMP\AWSTOE836058097\script-702966940.ps1:2 char:1
+ throw 'This is an exception from a PowerShell throw command'
+ CategoryInfo          : OperationStopped: (This is an exce...l throw command:String) [], RuntimeException
+ FullyQualifiedErrorId : This is an exception from a PowerShell throw command

2019-11-29 23:32:02 Info ExitCode 1
```

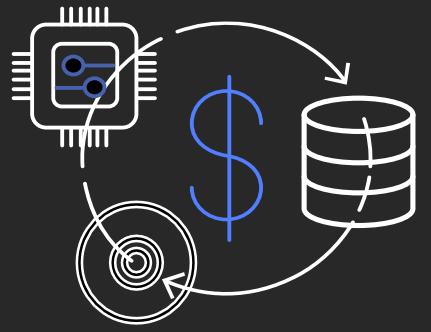
Pricing



No cost

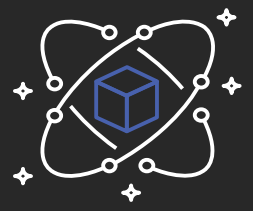


All operations run in your AWS account



Pay for the resources used in your account (e.g. EC2 instance usage, Amazon S3 usage, Systems Manager Advance, AWS Inspector, etc.)

Summary

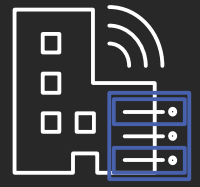


Produce automation to build images with ease

No need to write and maintain code to build automation
GUI wizard to create image building pipelines

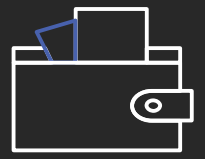
Improve security and uptime

Keep images secure and up-to-date
Capture and reuse security settings
Run tests to catch issues before deploying to production



Hybrid use cases

Produce AMIs for use on AWS
Generate on-prem VM images



No cost

Runs in customer account
Pay for resources used in your account

Thank you!

Samartha Chandrashekar

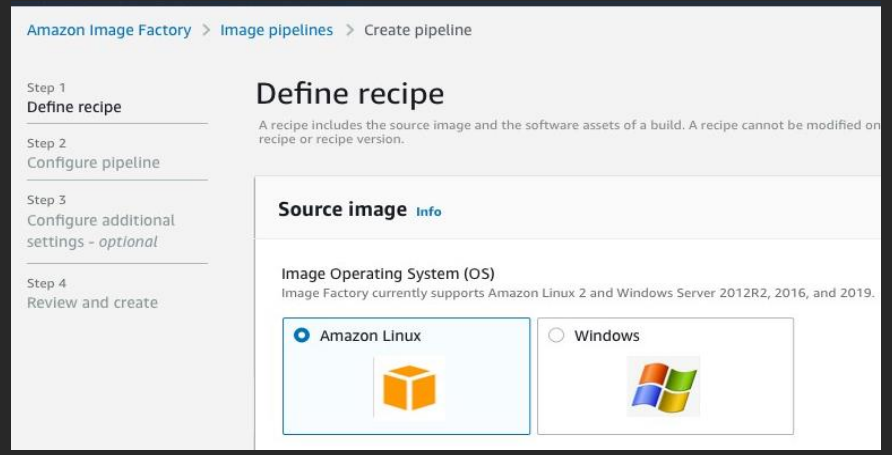
samarthc@amazon.com

AWS re:Invent

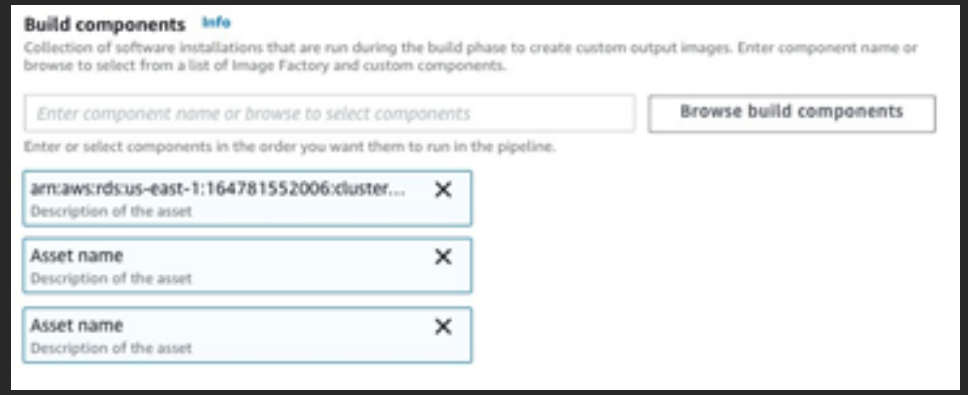
aws

EC2 Image Builder workflow

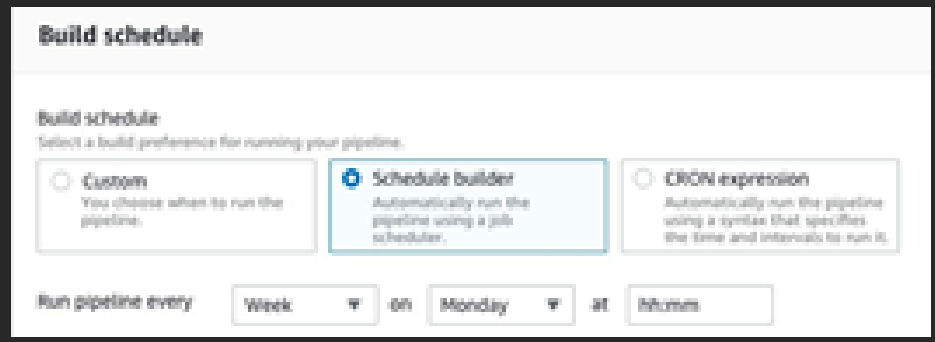
1) Start with a known good VM image



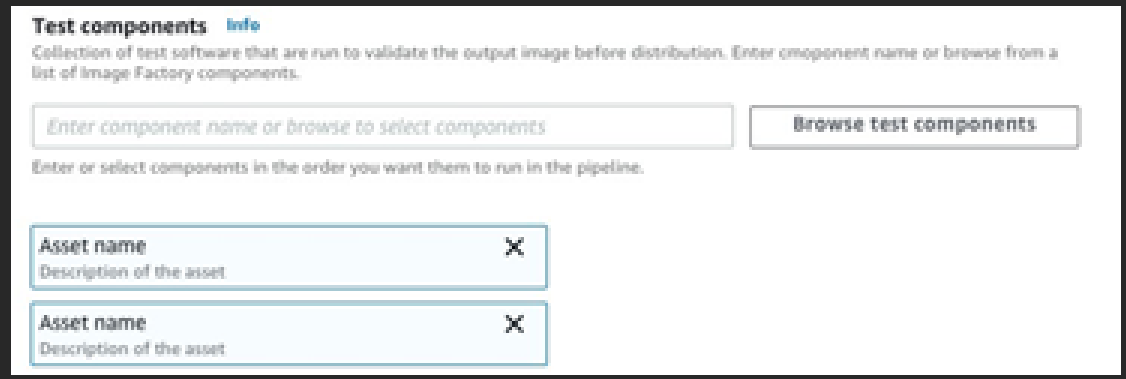
2) Select components to install and run



3) Select how frequently images are rebuilt with pending updates

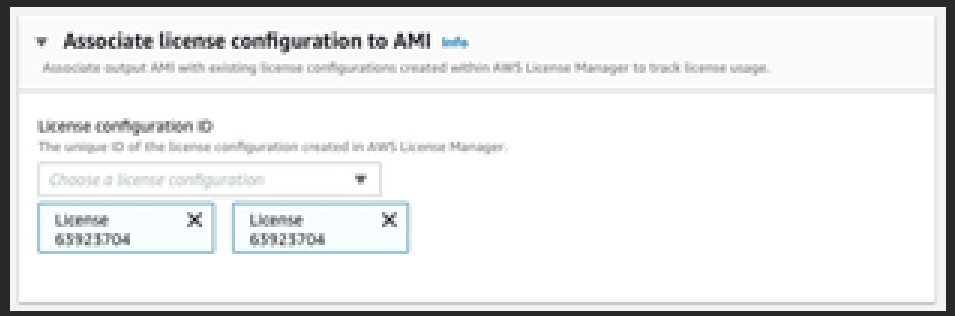


4) Select from a list of AWS and custom tests to run

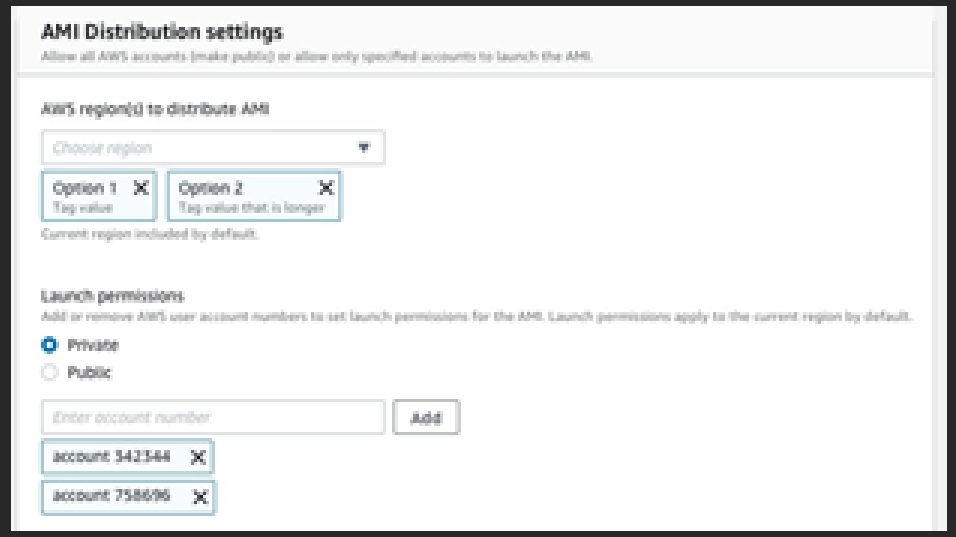


EC2 Image Builder workflow

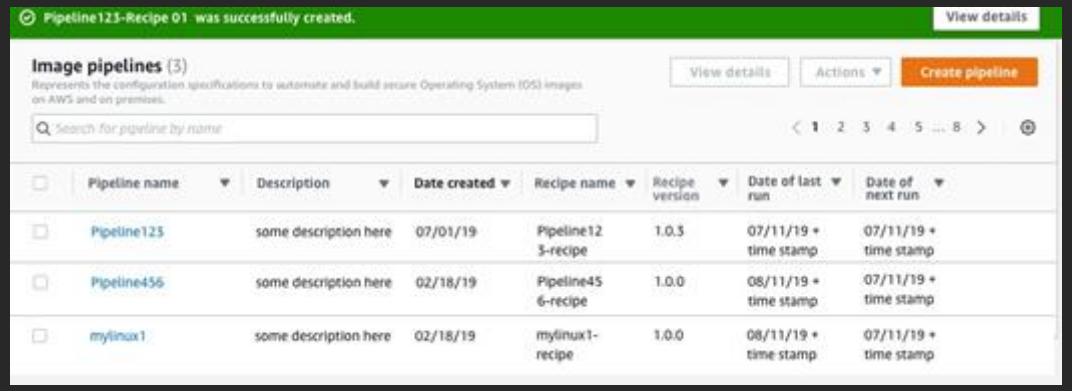
5) Select license configurations to attach to images



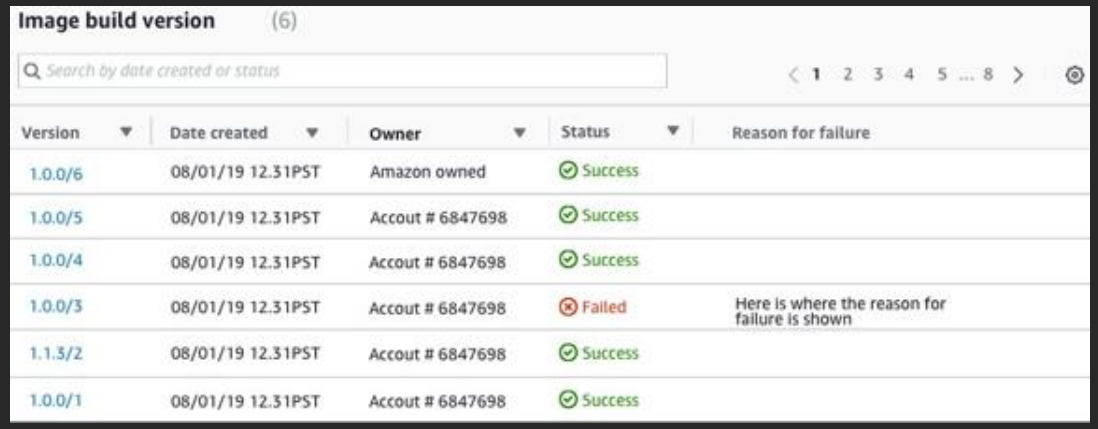
6) Select AWS regions for distribution



7) Review automated pipeline



Done! Images are produced by automated pipeline



Limits

Name	Description	Limit
Concurrent builds	Maximum concurrent builds that can be in progress in this account in current region	100 builds per account per region
Components	Maximum EC2 Image Builder components that you can create in an account in current region	1000 components per account per region
Component size	Maximum size of the data field of an EC2 Image Builder component	16 KiB
Image pipelines	Maximum EC2 Image Builder image pipelines that you can create in an account in current region	75 image pipelines per account per region
Image recipes	Maximum EC2 Image Builder image recipes that you can create in an account in current region	1000 image recipes per account per region
Build components & tests per recipe	Maximum build components that can be associated with a single image recipe	20 components per image
Infrastructure configurations	Maximum EC2 Image Builder infrastructure configurations in an account in current region.	1000 configurations per account per region
Distribution configurations	Maximum EC2 Image Builder distribution configurations in an account in the current region.	1000 configurations per account per region

YAML-based Document for build components & test

3 possible phases - build, validate, test

Each step in each phase uses **action modules** that return an exit code

- **ExecuteBinary**
- **ExecuteBash**
- **ExecutePowerShell**
- **Reboot**
- **UpdateOS**
- **S3Upload**
- **S3Download**
- **SetRegistry** (Windows only)

Output files for each execution

- **detailedOutput.json** describes details on the orchestration
- **document.yaml** is sent to the application for the execution
- **console.log** contains stdout and stderr information captured during the execution
- **application.log** contains logs generated by debugging executions

Actions performed by on-instance agent

ExecuteBinary to execute binaries with command line arguments

ExecuteBash to run bash scripts with inline shell code/commands

ExecutePowerShell to run PowerShell scripts with inline shell code/commands

Reboot to reboot the instance with a configurable delay

UpdateOS to install Windows and Linux updates

S3Upload to upload a file from a source file/folder to an Amazon S3 location

S3Download to download an S3 object or **KeyPrefix** to a local destination path

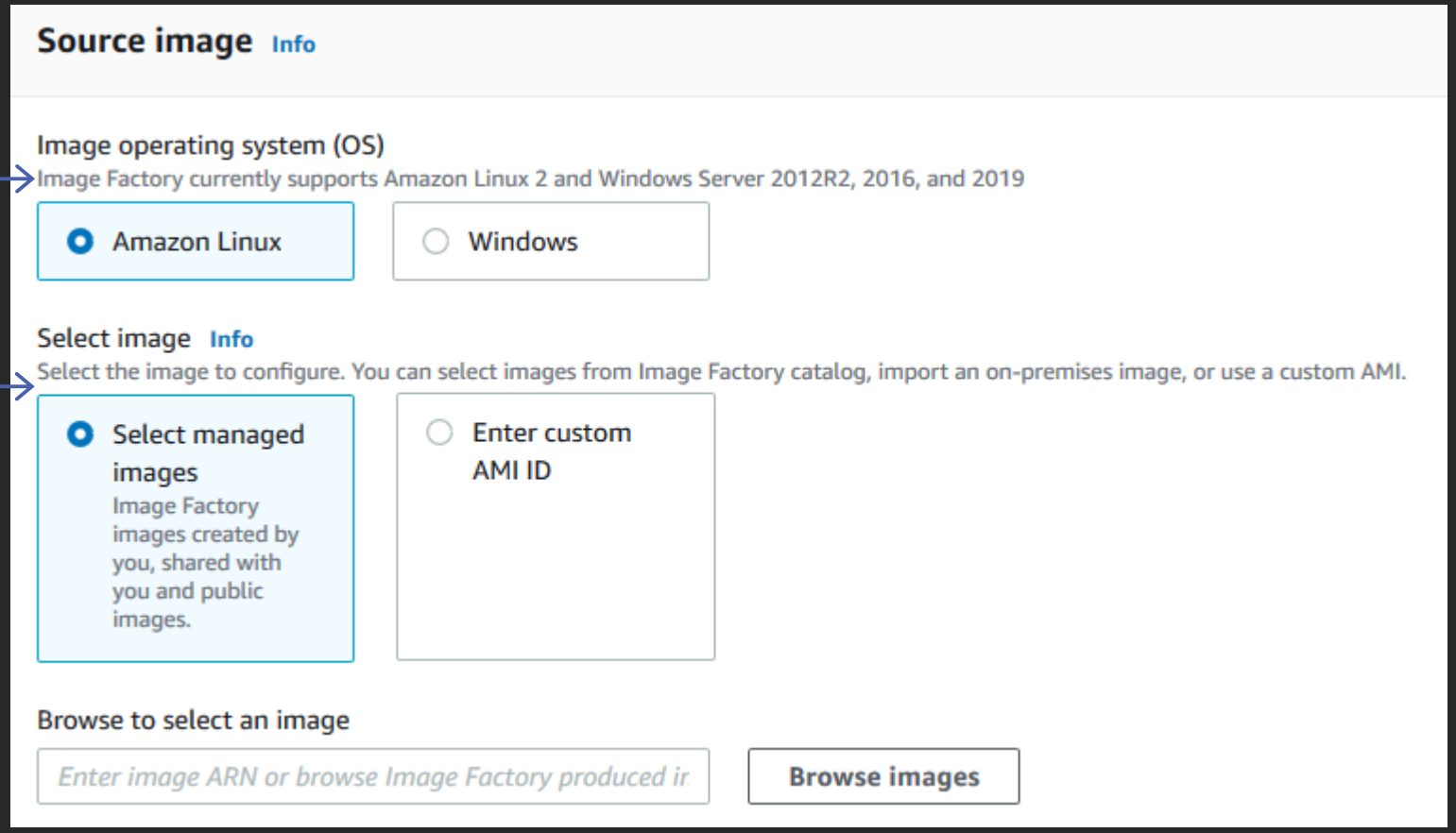
SetRegistry (Windows only) accepts inputs and sets a specified registry key

Start with a known good VM image

Select OS

Select image source

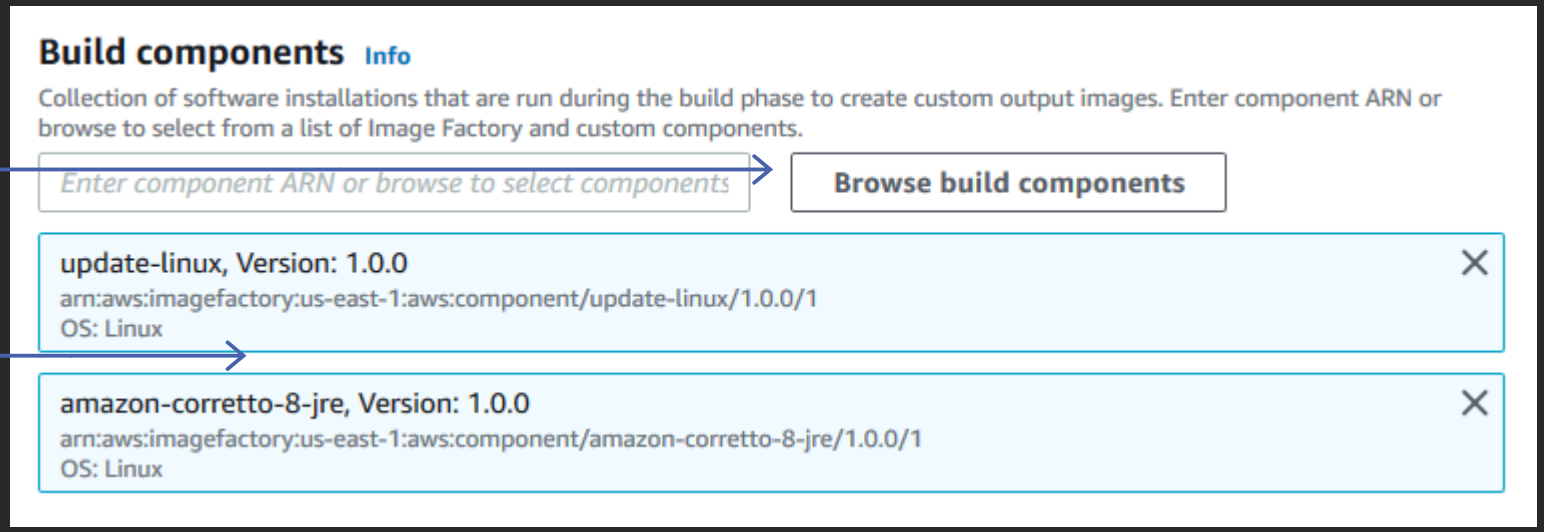
(either Image Builder managed image kept up-to date OR existing AMI)



Select software, hardening scripts, and configurations to install, run, and apply

Select components to be installed and applied

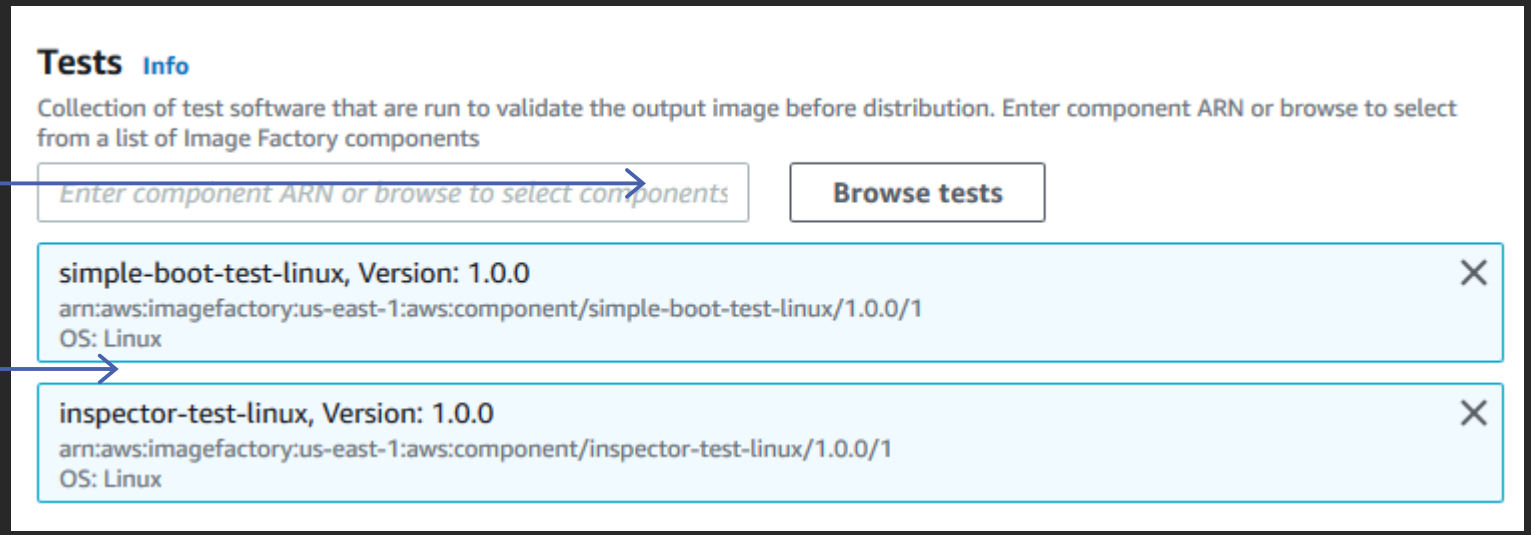
View selected components here



Select tests to run

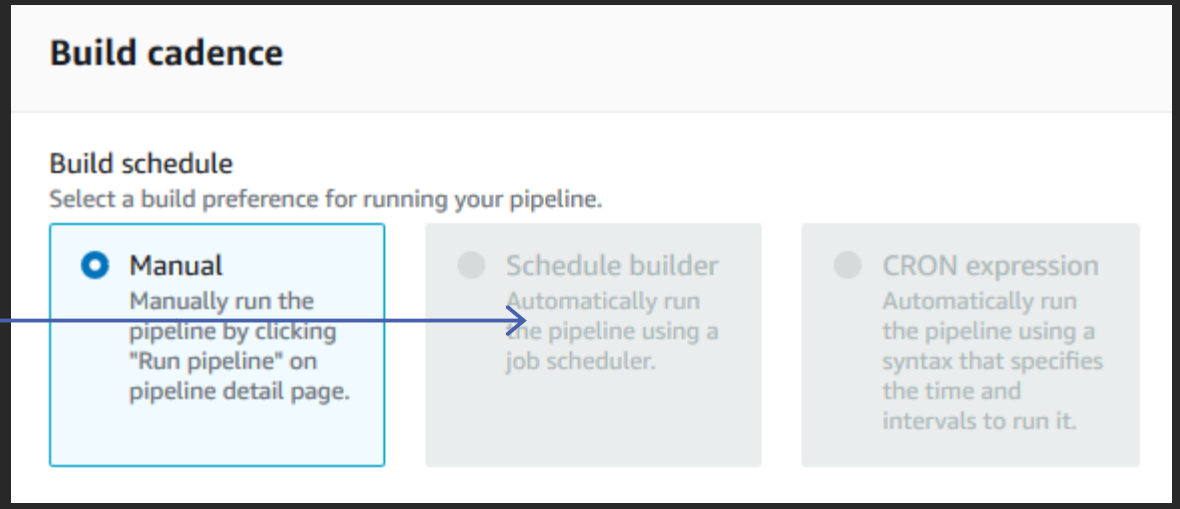
Select AWS provided tests to be run

View selected tests here



Select how frequently images are rebuilt with pending updates

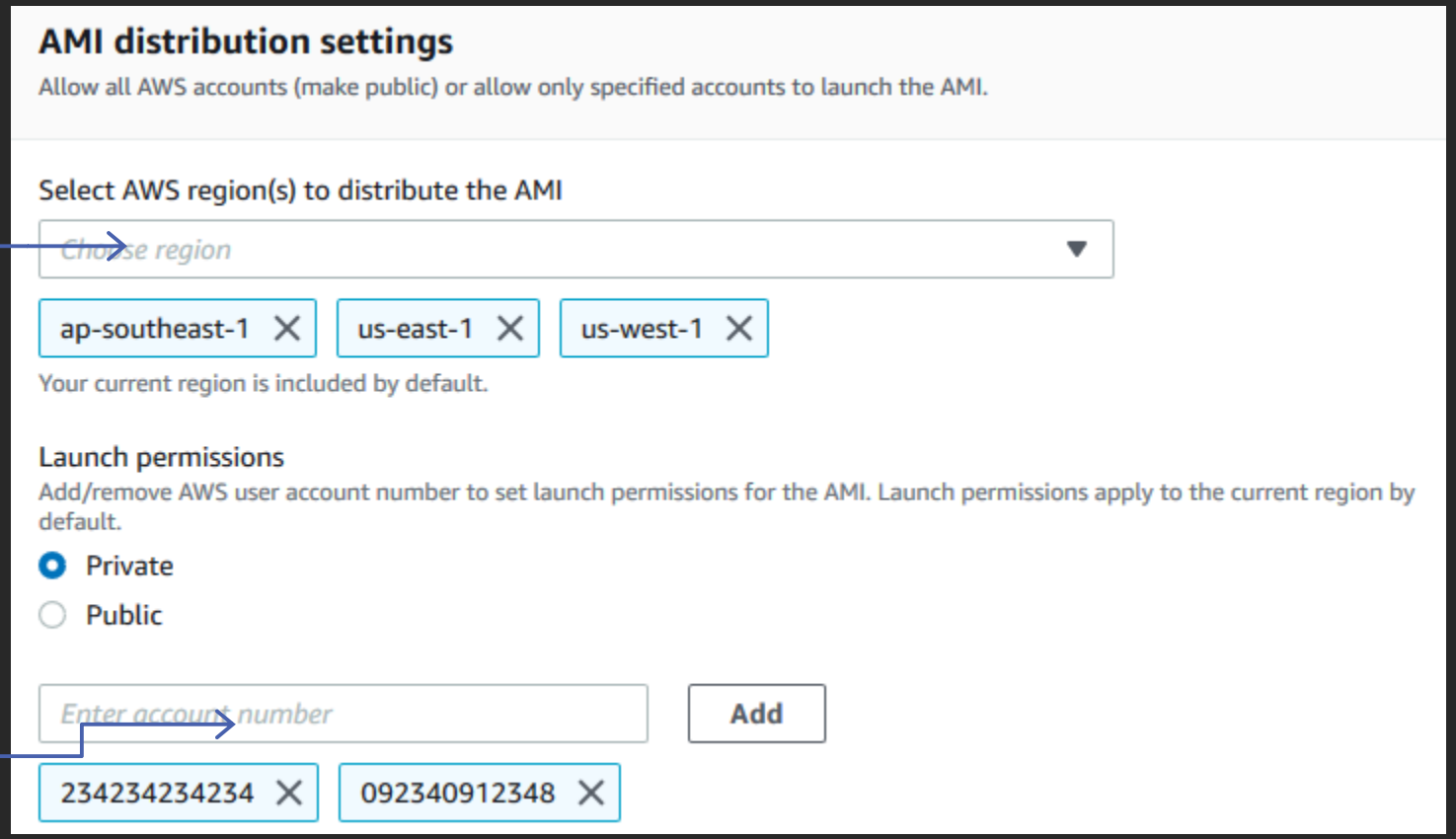
Set up a schedule to automatically produce patched and up-to-date images



Select AWS Regions for distribution and permissions

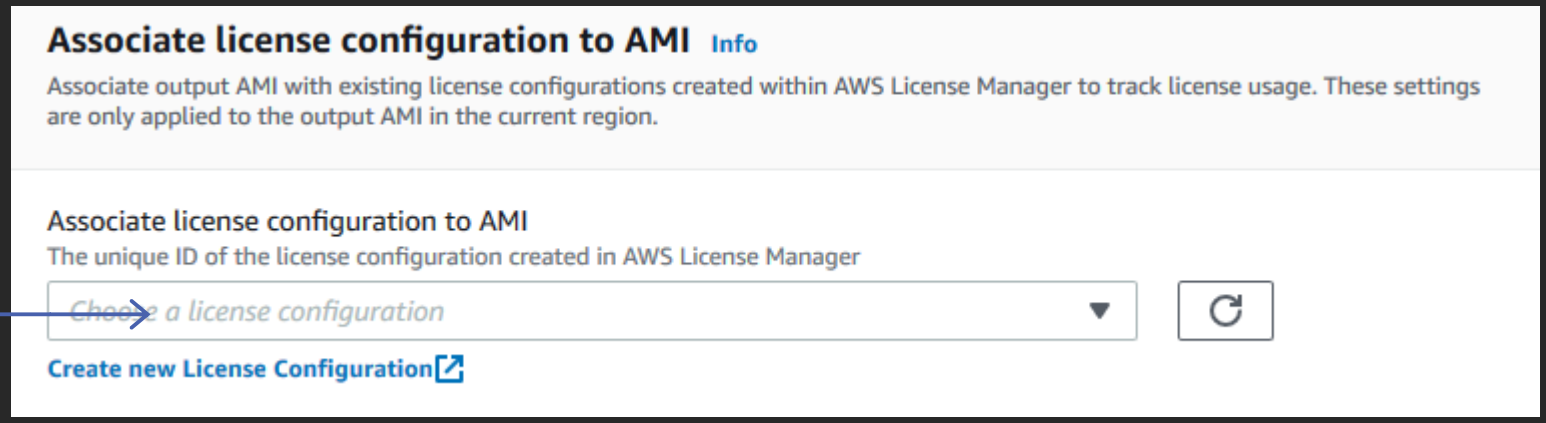
Select AWS Regions to distribute images

Select account IDs that can launch images

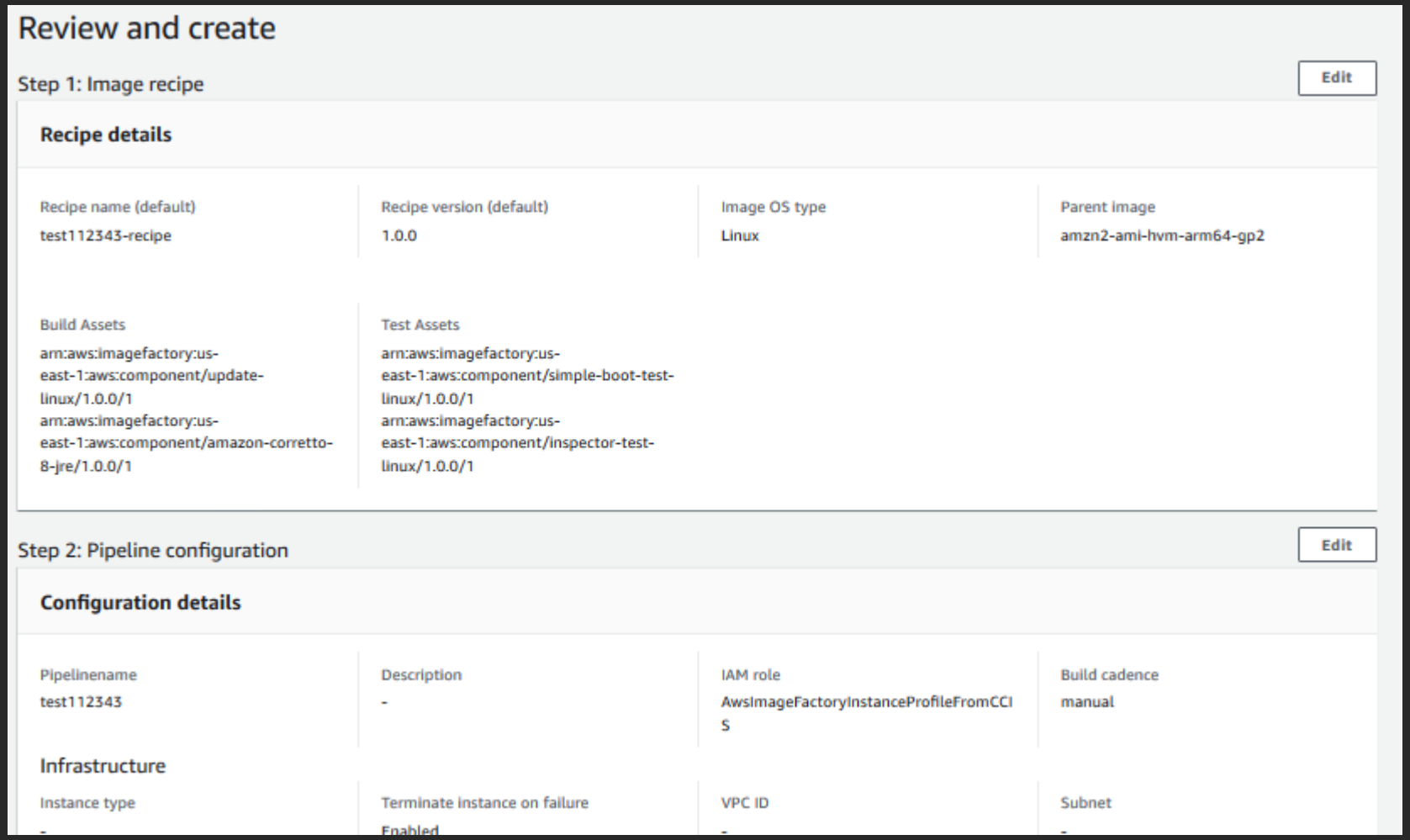


Select license configurations to attach to images

Select and apply license configurations

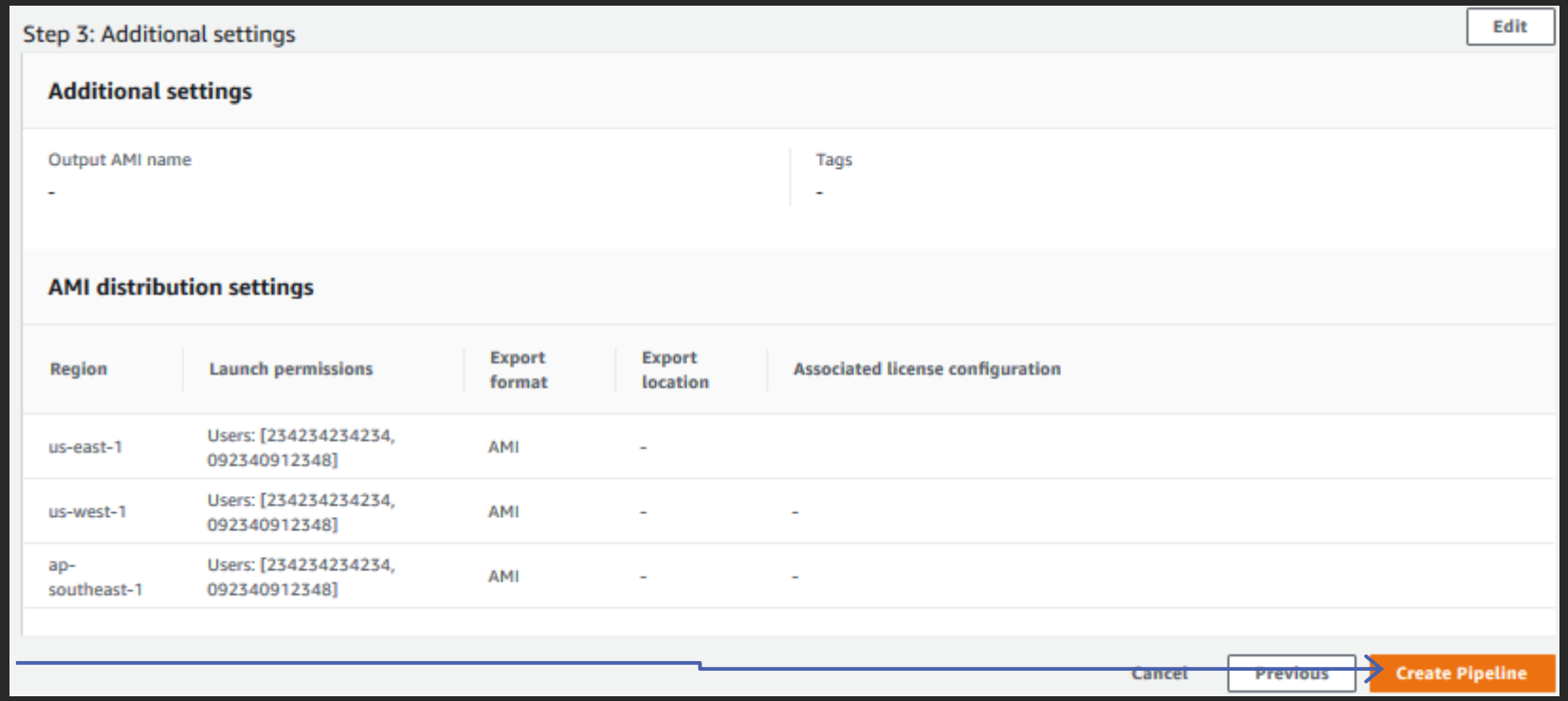


Review pipeline details

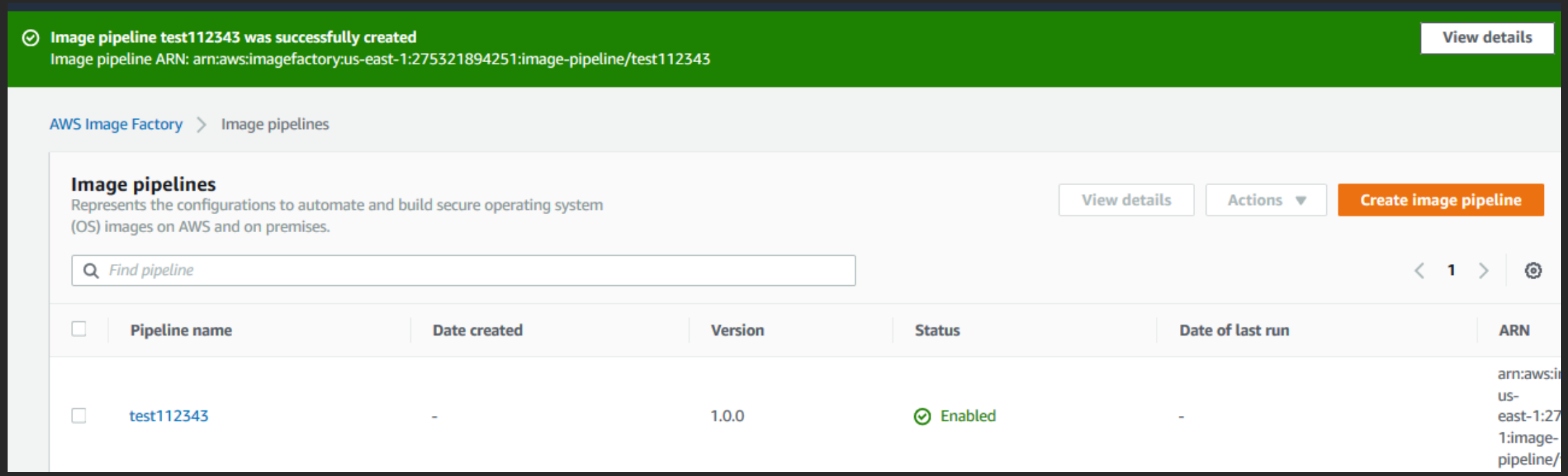


Create pipeline

Create pipeline after reviewing settings



Pipeline is now running!



Pipeline is producing images

Output images

Filter images

Image name	Version	Platform	Date created	ARN
amzn2-ami-hvm-arm64-gp2	2019.10.31	Linux	2019-11-07 | 23:51:07.636Z	arn:aws:imagefactory:us-east-1:aws:image/amzn2-ami-hvm-arm64-gp2/2019.10.31
amzn2-ami-hvm-x86_64-gp2	2019.10.31	Linux	2019-11-07 | 23:51:10.660Z	arn:aws:imagefactory:us-east-1:aws:image/amzn2-ami-hvm-x86-64-gp2/2019.10.31
amzn2-ami-minimal-hvm-arm64-ebs	2019.10.31	Linux	2019-11-07 | 23:51:13.356Z	arn:aws:imagefactory:us-east-1:aws:image/amzn2-ami-minimal-hvm-arm64-ebs/2019.10.31
amzn2-ami-minimal-hvm-x86_64-ebs	2019.10.31	Linux	2019-11-07 | 23:51:15.353Z	arn:aws:imagefactory:us-east-1:aws:image/amzn2-ami-minimal-hvm-x86-64-ebs/2019.10.31