Best Practices to Prepare your Amazon WorkSpaces for Linux Images

Amazon WorkSpaces for Linux

February 2020

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

Abstract

This whitepaper outlines a set of best practices for the image preparation of Amazon WorkSpaces for Linux. The paper covers Well-Architected Principles applied to image design, the make-up of an Amazon WorkSpace, the Bundle and Image process for Amazon WorkSpaces, and methodologies for Image management. The paper addresses best practices for operating system updates, remote protocol configuration, application deployment, and desktop experience configuration of MATE.

This whitepaper will help you in your role as a desktop configuration or security engineer.

Introduction

Amazon WorkSpaces is a managed desktop computing service in the cloud. Amazon WorkSpaces removes the burden of procuring or deploying hardware and delivers a desktop experience. Administrators can provision WorkSpaces with a few clicks on the AWS Management Console, using the AWS Command Line Interface (CLI), or by using the range of Application Program Interfaces (APIs). With Amazon WorkSpaces, you can launch a desktop within minutes, establish a connection, and access your desktop software on-premises or through an external network securely, reliably, and quickly.

For your WorkSpaces users, you can choose between multiple hardware configurations and operating system types. You can launch a WorkSpace that runs Amazon Linux 2, which is bundled with the Amazon Linux WorkSpaces Desktop, Firefox, Evolution, Pidgin, and Libre Office. The Amazon Linux WorkSpaces Desktop uses the MATE Desktop Environment. The MATE Desktop Environment is the continuation of GNOME 2. It provides an intuitive desktop environment using desktop metaphors for Linux operating systems.

If we consider typical Amazon Linux WorkSpace use cases, there are a number of benefits to preparing a customized Amazon WorkSpace ready for deployment directly to users. For a development environment, the ability to pre-configure the desktop provides a ready identification of different WorkSpace sessions for production, development and test environments. For productivity users, pre-configuration allows for a standardized environment that is protected from accidental change. Organizations that need to provide their users with a mix of Windows and Linux environments can create a unified operations and configuration model with a single set of tools (such as Puppet or Chef) and processes and a consistent interaction experience that meets the needs of the entire user community.

Consider the following question when preparing to manage Amazon Linux WorkSpaces:

- How will you deploy applications to the image?
- How will you keep the base operating system and applications current for security and functional updates?
- How will you maintain the user configuration in order to enable best time-to-value, or a recoverable state should errors occur?

The following sections provide details about image management for Amazon WorkSpaces for Linux, explain the principles and deployment methodologies, and explain the options and features that are available for configuration.

Well-Architected Principles for Image Management

The Well-Architected Framework helps build secure, high performing, resilient, and efficient infrastructure for applications. Based on five pillars — Operational Excellence, Security, Reliability, Performance Efficiency, and Cost Optimization — the Framework provides a consistent approach for customers and partners to evaluate architectures and implement designs that will scale over time.

You should consider the application of the Well-Architected pillars when designing your image management methodology and preparing your images. Appendix A provides a summary checklist to help you review your image management methodology.

Design for Automation where possible

Design to deliver the least possible manual intervention in the process in order to enable repeatability and scale.

Design for Cost Optimization

Cost optimization with Amazon WorkSpaces cannot only come from the pay-as-you-go model, or the deployment of the WorkSpaces Cost Optimizer.

Your preparation of the Amazon WorkSpaces image for supportability and productivity helps drive cost efficiencies through reduced support requests and improved user productivity. All things being equal between AWS Regions with regards to data locality, with regards to user experience and security, review the costs for workspaces per Region to select the most cost effective location to deploy you WorkSpaces.

Design for Efficiency

Minimize the resources needed to deliver an image. This is not necessarily a recommendation to deploy one image with all possible configurations contained within it. While this is frugal in numbers of images, you will be deploying and maintaining resources for users who may not be using them.

When minimizing the number of images, consider where use cases overlap, and consider abstraction of applications from images to be as efficient a service as possible for delivering new images, updated images, and rebuilt images.

Design for Flexibility

A user's persona is the representation their application needs, their access rights' needs, and their compute needs. Your organization's requirements may need a number of personas to match user needs, which can require multiple images in order to optimize delivery time, or accommodate conflicting resource requirements.

Create a consistent deployment mechanism that can handle multiple scenarios, and can scale with the same mechanism customized using tagged use case and profile identifiers.

Design for Productivity

You can create a WorkSpace and allow individuals to customize that environment as they see fit. This is a valid operating model. However, there is value in having a preconfigured, validated deployment so that when users first connect, they can be productive as soon as possible, and minimize the time needed for application deployment and additional customization.

Amazon WorkSpaces are available across a number of Regions. Virtual Desktop best practice is to locate the user environment in the same Region as the data that the user is accessing. Always consider the user to have an Amazon WorkSpace in a Region with the most responsive latency time to the associated user data and services.

Design for Scalability

The pay-as-you go model that Amazon WorkSpaces use can drive cost savings by creating resources as needed, and removing them when they are no longer necessary. It is recommended to design your management process for Amazon WorkSpaces for Linux to allow WorkSpaces be deployed, or removed, rapidly.

Design for Security

When preparing your image, avoid including sensitive information like security keys in the build. Ensure that the image patched with operating system and application updates. Consider configuring agents to validate the security posture of the WorkSpace, and to forward application and operating system logs for analysis:agents can include antivirus agents, malware agents, and event monitoring agents.

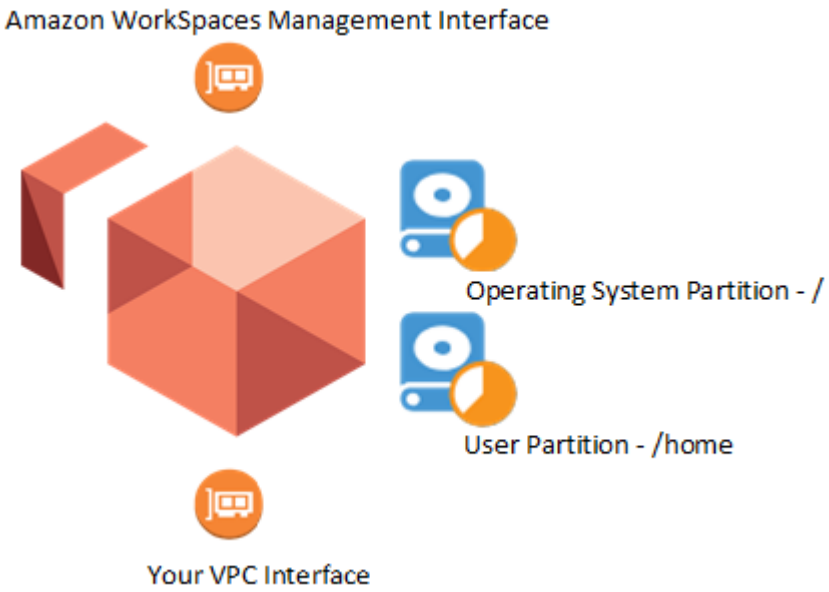
You can force higher levels of security for the streaming traffic and restrict user ability to cut and paste data in/out of the WorkSpace.

You can encrypt the data at rest in an Amazon WorkSpace. Encryption must happen at deployment. You cannot use Encrypted workspaces to create new images. Do not encrypt your template images.

Design for Supportability

Users make mistakes. Design your image management and application deployment to allow for non-invasive support and recovery mechanisms and processes.

Amazon WorkSpaces Configuration



Each Amazon WorkSpace comes with two disks. With an Amazon Linux WorkSpace, the system disk (`/dev/xvda1`) mounts as the root folder. You can select a starting size for this drive of 80 GB, or 175 GB. A second disk stores user data and applications; `/dev/xvdf1` mounts as `/home`. You can select a starting size of 10 GB, 50 GB, or 100 GB.

A best practice is to minimize the size of the user drive, which reduces the cost of running the WorkSpace. You can allow users to increase the volumes as necessary, up to 2000 GB each.

You can encrypt either the root, or the home volume before WorkSpace creation. If you encrypt an Amazon WorkSpace partition, that encrypted cannot create future WorkSpace images.

The Amazon WorkSpaces service scans all WorkSpaces for health states every 12 hours, and takes snapshots of both root and user volumes when WorkSpaces are in a healthy state. Amazon WorkSpaces has a restore function that recreates both of the WorkSpace's root and user volumes based on the most recent snapshot pair taken when the WorkSpace reports healthy from the last scan. After restore, all changes made to the WorkSpace after the snapshot time are lost.

When the user 'rebuilds' a WorkSpace, the WorkSpace restores the most recent image of the bundle used to create the WorkSpace on the system disk. The service restores the user data disk from the last healthy automatic snapshot taken of the data drive. In addition, the primary Elastic Network Interface (ENI) is re-created. The rebuilt WorkSpace receives a new private IP address from the pool of addresses available for the Virtual Private Cloud that the WorkSpaces are connected to.

Best Practices for Amazon WorkSpaces Images and Bundles

When preparing an image, consider how you will manage updates in a production environment. You may employ all or some of the following best practices.

To best manage at scale, create a custom bundle (or bundles), and deploy your Amazon WorkSpaces from the bundle (or bundles) you create. When creating a custom bundle, ensure that the WorkSpaces for your users have everything that they need installed. If you must perform software updates or install additional software on your WorkSpaces, you can update the image assigned to the bundle and rebuild your WorkSpaces.

All organizations should have a documented process for provisioning to ensure consistent creation and recreation of images. When determining what to include in a Linux WorkSpace image, consider the following design best practices:

- Avoid embedding passwords, private keys, or other sensitive information in the image.
- Avoid hardcoding application license details in the image to ensure that you do not deploy WorkSpace instances that contravene your application license requirements.
- Leverage AWS CloudFormation, scripting, or a third-party configuration management tool to document and automate the image creation and update process.
- Create a library of reusable, modular application packages or scripts that you can programmatically assemble to create different types of WorkSpace images.
- Instrument images with a standard bootstrapping capability to allow the instance to reference runtime information at launch.
- Develop a consistent strategy for tagging images to allow for easy organization and identification of the images and their contents.

You can create a custom WorkSpace bundle from the image you create. After the bundle is available, you can launch WorkSpaces from your custom bundle.

For more information on Amazon WorkSpaces Images and Bundles, refer to the [WorkSpace Bundles and Images documentation.]

Amazon Linux WorkSpace Image Design Options

Amazon WorkSpace image design options exist along a spectrum of deployment simplicity in relation to deployment flexibility. The simplest images are fully baked (also known as ‘fat’ images) and purpose-built to deploy a complete running instance, including the installation and configuration of all required software. However, this approach limits flexibility, as a fully baked image will have a fixed application set. The most flexible images, zero images, will include only minimal configurations and software before dynamically installing the required packages either on first boot, or by policy, based on the user’s requirements. This approach is a trade-off of simplicity for flexibility, as each instance must be properly configured before it can be functional for the user.

You may also have thin images, (also known as hybrid images) which have some applications installed into the image and some delivered remotely.

The ideal WorkSpace image design for your environment depends largely on your user’s application needs and configuration setup. Typically, organizations have groups

of users with similar application portfolio requirements and configuration setups. These different user personas require different images that in turn require different image design methods. When considering which option is right for each of your user profiles, keep the following questions in mind:

- How quickly do you need to be able to deploy or rebuild a failed WorkSpace, or add additional WorkSpace capacity?
- Does the user persona baseline stay static for a relatively long period?
- Does the target persona require manual provisioning or configuration?
- Do you need to minimize the complexity of deploying resources to both AWS and on-premises environments?
- Are there existing application provisioning tools or processes that you are trying to align with AWS?

Fully Baked

Image with all software installed and managed as a single unit.

On one end of the spectrum is the most common method for image design that provisions a fully functional Amazon Linux WorkSpace, including all necessary software for a specific user persona. Organizations often create a baseline image that conforms to minimal security and configuration requirements, and then build on this baseline to create fully baked images that are specific to applications or infrastructures for actual instance deployment.

Fully baked images are the simplest to deploy and provide the fastest time to value for users. For this reason, fully baked images are useful to quickly deploy replacement instances or add additional capacity. Fully baked images also require minimal changes on launch. You can capture manual installation and configuration steps that you cannot automate. The process of using fully baked images is similar to how many organizations deploy desktops, which makes this simple approach applicable to customers who are new to Amazon WorkSpaces service, and are accustomed to image-based server deployments.

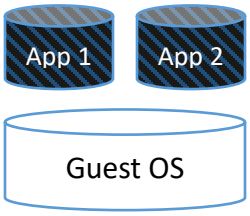
At scale, it can be cumbersome to maintain unique images for large numbers of user personas. This approach is therefore more suitable for a small number of well defined, consistent persona deployments, or when combined with an automated custom bundle build and management system where there is a regular skew from a base image.

Considerations

- The user persona for deployment typically needs a discrete, pre-defined set of applications that rarely changes.
- The application installation process requires no additional tooling.
- License control and reclamation can be complicated when all applications are installed and available to users in a standard configuration and the application itself has no license management capability.
- There is no requirement for additional application management tools. The application update process will be managed by applying updated images to custom bundles applied to WorkSpaces. Regular patch management can be accommodated through the WorkSpace Maintenance Window
- Updating WorkSpaces using rebuild has the least impact to users because all application installs are inherent in the image.
- Create a startup script to remove the user from sudo permissions

.

Zero Images



Core OS only with software distribution agents

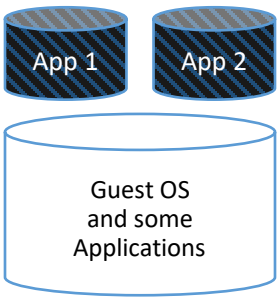
At the other end of the spectrum is having only the operating system provided by the image with application delivery fully abstracted and not part of the image. This approach creates an image that combines a minimal operating system with a configuration management agent (or agents) that builds a fully functional system at instance launch. On WorkSpace startup, the configuration agent downloads, installs, configures, and integrates all the required software.

Zero images offer the most flexibility during deployment and the highest levels of portability because they leverage minimal server images. Customers who prefer this approach typically have experience with AWS and configuration automation tools. They want deployment flexibility to be able to pull the latest software builds and updates at launch, or they might require minimal complexity in order to deploy resources to both AWS and on-premises environments.

Considerations

- The user's persona for deployment requires a set of pre-defined applications available through an application deployment tool that can automatically register newly built WorkSpaces. It is recommended that sudo access for the user be removed as part of the image creation process to prevent the user altering the configuration of the environment.
- Delivering applications can be more efficient where different personas have variable application needs. Application deployment is delivered through a configuration tool (such as Puppet) rather than by creating and maintaining multiple images with different application sets.
- Delivering applications can be more agile because, by using a configuration management tool, users can be targeted with updates while working rather than wait for the rebuild of a new custom bundle.
- Using application delivery methods separates applications from the OS.
- Applications can be quickly deployed, updated, and reclaimed.
- Zero images enable you to deliver applications based on user groups (or tags) through a configuration management tool, and zero images remove the need to give users Administrator access on the image. Zero images abstract application delivery from image deployment. By deploying applications only when the user needs them, rather than have them installed by default as part of the image, you can deliver more focused licensing control and user access visibility.
- Adapting to business needs is faster because the time to deliver applications on the demand of the user is not dependent on an image update.
- There is a requirement to have an image deployment/image management tool in place.
- Create a startup script to remove the user from sudo permissions.

Thin Images



Zero + light footprint of manually installed software

Thin WorkSpace images provide a subset of the software needed to produce a fully functional instance, falling in between the fully baked and zero options on the image design spectrum. This approach creates a partially baked, WorkSpace image that is configured on first boot based on specific application requirements. You base the decision about what to bake into the image and what to install and configure at launch on a number of factors. First, how reusable the underlying software package is across all instances. Also, how long it takes to download, install, and configure the software component. For example, to reduce instance launch times, include server management agents (Such as agents for Puppet or Chef) and baseline security configurations into a WorkSpace.

Thin images combine the flexibility of post-launch changes with the speed of preinstalled and preconfigured infrastructure components. This can be especially useful for merging frequently changing software (such as security updates) with infrequently changing software (such as management agents or productivity applications). Customers who use this approach typically have a moderate amount of experience with AWS and configuration automation tools. These customers want to combine the deployment simplicity of a baked image with the deployment flexibility that pulls the latest software builds and updates at launch.

Considerations

- The user's persona for deployment requires a set of pre-defined applications as part of the base configuration. You need a set of applications available through an application deployment tool that can automatically register newly built WorkSpaces.
- Adapting to business needs can be faster than the fully baked method, as the application can be deployed without rebuilding the instance.
- Application deployment happens after the WorkSpace is deployed and the WorkSpace has been identified by the application management tool. There will be a delay in time to value for new instances while the application installation completes. Delivery of application updates will be faster than a complete rebuild, which is required by a fully baked image and can occur while the user is actively using their WorkSpace.

- Although this approach requires fewer images to maintain than the fully baked image approach, it still results in more images to maintain than the zero image approach.
- You are required to use an image deployment/image management tool.

Example WorkSpace Image and Bundle Tagging Structures

You can assign tags to your metadata to help you manage your images. The most common tags are listed below.

Technical Tags

- Name – A simple identifier for individual instances.
- Version – An identifier to help distinguish between Images with different versions.
- Software – A list of software included in the image.
- Role – The role of an instance launched from this bundle (for example, web server, message broker).
- System – The IT or application environment into which this image supports. This is especially useful for fully baked images that have to embed system-specific configuration information.

Business Tags

- Project – One or more specific projects for which this bundle was created for.
- Cost Center/Business Unit – The cost center or business unit associated with the bundle.
- Stage – The development stage of the bundle to help identify if it is appropriate for production use.

Security Tags

- Confidentiality – An identifier for the specific data-confidentiality level of the bundle/image. This is useful for organizations that want to embed additional security controls in WorkSpaces that process particular levels of classified data.

- Compliance – An identifier for images/bundles designed to adhere to specific compliance requirements.

Amazon WorkSpaces Image Maintenance

You can create an image either from a default Amazon Linux WorkSpace image, or from an image that you have customized. You can create, or update, custom bundles from this image.

All Amazon Linux bundles come with the following packages: Firefox, LibreOffice, Evolution, Python, Pidgin IM, GNU Image Manager, MATE Desktop Environment.

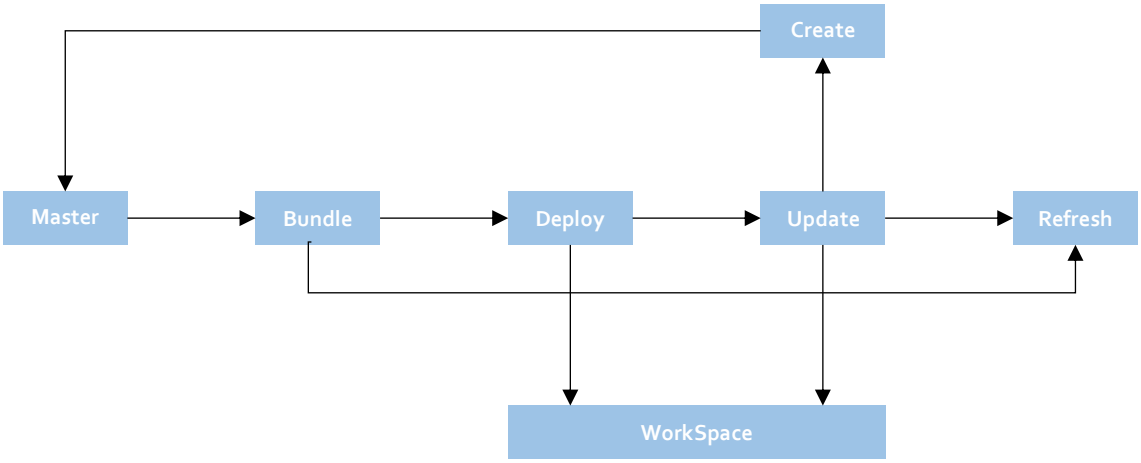
Process for Amazon Linux Image Creation

In summary, the process to prepare a default WorkSpace instance for a custom image is as follows:

1. Select the base image. This may be a default bundle supplied by Amazon Web Services, or an existing Linux WorkSpace image in your account.

   Note: To optimize costs, select a minimum disk size for the system drive (minimum is 0GB) and for the user data drive (minimum is 10 GB).
2. Validate that the most recent operating system patches are applied.
3. Download the required software to a shared drive/data location.
4. Install the required features and updates into the image from the shared drive/data location.
5. Install drivers where applicable.
6. Reboot.
7. Test the build.
8. Clean down the image (remove temporary install files, logs, user accounts).
9. Create/update a version control file on the image.
10. Reboot.
11. Create a new Image and create/update tags.

Image Deployment Workflow



1. **Create:** A master image is created and configured.
2. **Bundle:** Assign the master image to a custom WorkSpace bundle.
3. **Deploy:** The custom bundle is used to create new Amazon Linux WorkSpaces.
4. **Update:** Using a new WorkSpace, update the operating system and applications with the appropriate patches as per the Image Management process. Create a new Image. Apply the new image to the custom Bundle. All new WorkSpaces will have the most recent image applied.
5. To update deployed WorkSpaces:
    a. Rebuild the WorkSpaces. The rebuild process uses the updated image associated with the custom bundle from which the WorkSpace was created.

Note: The operating system drive is updated with the current custom bundle. The user data drive is reset to the last user data snapshot.

b. Update existing WorkSpaces with patch updates and new applications. The updated bundle can be used to rebuild existing workspaces, or to deploy new workspaces with the most recent revision.

Bundle Maintenance

Use an image to create or update a custom bundle. Custom bundles ensure that user's WorkSpaces link to a specific configuration. If you need to perform software updates or install additional software on your WorkSpaces, you can update your image, test it, validate it, and apply it to the custom bundle. You can then schedule a re-build for your users WorkSpaces to make use of the updated custom bundle.

A rebuild of a WorkSpace resets the system drive of the WorkSpaces to that of the current base bundle associated with the WorkSpace. A rebuild of a WorkSpace also resets the user data drive to the last snapshot, which can be up to 12 hours old.

Recommended Image Customizations

This section defines best practices for settings and configurations. We recommend that you test and validate in your own environment and use these settings as a base, rather than a default.

Operating System Patch Updates

It is a best practice to build your images using the most up-to-date operating systems, packages, and software. The Amazon Linux 2 WorkSpace image is updated on a regular basis. It is a best practice to check that the installed software is up to date before committing your image.

- To check the image:
  - `sudo yum check-update`
- To update the image:
  - `sudo yum update`

PCoIP Configuration

You can use the *Personal Computer over Internet Protocol* (PCoIP) protocol to connect from your Amazon WorkSpaces client to your Amazon Linux WorkSpace.

You control the behavior of the PCoIP Agent by configuration settings in the `pcoip-agent.conf` file, located in the `/etc/pcoip-agent/` directory.

- Description of all settings:
  - `man pcoip-agent.conf`
- To edit the file:
  - `sudo pluma /etc/pcoip-agent/pcoip-agent.conf`

Consider at a minimum, forcing the encryption to be AES256 only.

Setting the PCoIP configuration in the image provides a consistent client connection configuration.

To deploy and enforce changes to the policy, use a configuration management solution that supports Amazon Linux, such as Puppet, or AWS Systems Manager. Any changes take effect when the agent starts: restarting the agent ends any open connections and restarts the Window manager.

File Locations for Application Configuration

When selecting disk locations for application deployment, consider that an Amazon Linux WorkSpace has two disks, a system disk (under `root /`) – which is saved as the image, and a user disk (files stored under `/home`) – which is created for each new user.

When users rebuild an image, the system disk is recreated based on the latest bundle version. The user disk is recreated from the last snapshot, which could be up to 12 hours old.

As a best practice user-wide, managed configuration items, application installs, and patch installs, should be installed on the system disk (i.e under the root location).

If users are to install their own packages, they should install those packages and configuration items on the user disk (that is, under the /home).

MATE Desktop Environment

Amazon Linux WorkSpaces use the MATE Desktop Environment. This environment provides an intuitive and attractive desktop environment using traditional metaphors for Linux operating systems. Configure the MATE framework to help you create a consistent user experience as part of your Amazon Linux WorkSpace bundle. You can also allow users granular control over configuration changes and locking desktop settings.

Desktop user experience options include:

- Background
- Desktop Icons
- Definition of application settings
- Color Schemes
- Menu Options

MATE settings are stored using `dconf`, a simple key-based configuration system. `Dconf` stores information in a binary database file. You can change the `dconf` file in an Amazon Linux WorkSpace using the `gsettings` command line tool.

Applications using dconf install a schema (a list of the available settings). Users can change the settings, and those settings are stored in a binary file located at `/home/[username]/.config/dconf/user`.

For more information on how to configure the MATE Desktop Environment, visit https://mate-desktop.org/ . For a guided walkthrough for Amazon Linux desktop configuration, view Customizing the Amazon Linux WorkSpace

Ensure that Security keys and Passwords are not stored in an image

When preparing your image, avoid including sensitive information like security keys in the build. Ensure that such configuration items are deleted before saving the image.

Configuration Management

You can use images to deploy a standardized WorkSpace for your users, and you can regularly update your images to rebuild and include or update the custom bundle for

your WorkSpaces. This methodology ensures the deployment and maintenance of a fully baked image.

Customers may want to set configuration files at scale for security and compliance; even when the image is fully baked. Including the agent component for a Configuration Management agent in your image helps maintain security and compliance by scanning your instances against your patch, configuration, and custom policies. Using a Configuration Management tool (such as Puppet or Chef), allows you to define patch baselines, maintain up-to-date antivirus definitions, and enforce firewall policies. You can also remotely manage your WorkSpaces at scale without rebuilding each WorkSpace.

Conclusion

There are a number of benefits to preparing a customized Amazon WorkSpace ready for deployment directly to users. When creating the design process for your Amazon WorkSpaces image build, apply the Well-Architected pillars to design for automation, cost optimization, efficiency, flexibility, productivity, scalability, security and supportability.

This whitepaper describes the best practices for doing this. The result of following the guidelines in this whitepaper will be well-architected image creation process that can assist in scaling your Amazon WorkSpaces deployments on the AWS global infrastructure.

Appendix A

Image Design Checklist

This image design checklist can be used as a basis for evaluating how your image creation and management process align with recommended practices. Assess your current activities as image design tasks and provide a score from 1-5, where 1 is not aligned and 5 is highly aligned.

Work with your Amazon Web Services Account team to review this process as part of a wider Well-Architected Review for Amazon WorkSpaces.

Well-Architected Principle	Image Design Task	Score
Automation	Image configuration is scripted.	
Cost Optimization	Only necessary applications are installed into the base image.	
Cost Optimization	Custom bundles are created with the lowest disk storage capacity.	
Cost Optimization	Application license details are stored externally from the image.	
Efficient	The number of images aligns with your identified user personas.	
Efficient	Each image delivers to a discrete application set.	
Efficient	Configuration management tools are used to provide application abstraction.	
Efficient	Images are instrumented with a standard bootstrapping capability to allow the instance to reference runtime information at launch.	

Flexibility	Images have a consistent deployment mechanism that can handle multiple scenarios aligned with use case and profile identifiers.	
Productivity	Images are created to ensure that a preconfigured and validated deployment is available.	
Productivity	You have created a library of reusable, modular application packages or scripts that you can programmatically assemble to create different types of WorkSpace images.	
Scalability	Image configuration does not rely on manual changes.	
Scalability	The image deployment process creates and updates with minimal process steps.	
Scalability	Configuration management tools are used appropriately for application deployment and configuration.	
Security	Security keys or configuration items are not included in the build.	
Security	Agents are configured to validate and ensure the security posture of the WorkSpace.	
Security	Application and operating system logs are centrally collated for analysis.	
Security	PCoIP Configuration is optimized to align with security requirements for secure data encryption and clipboard controls.	

Supportability	Images contain agents and logging to allow application performance and reliability analysis.	
Supportability	Images contain tools to facilitate remote assistance.	
Supportability	You have a documented workflow process for creating and updating images and bundles.	
Supportability	You use a consistent strategy for tagging images to allow for easy organization and identification of the images and their contents.	

Further Reading

For additional information, see:

- AWS Best Practices for Deploying Amazon WorkSpaces
- Troubleshooting AWS Directory Service Administration Issues
- Troubleshooting Amazon WorkSpaces Administration Issues
- Troubleshooting Amazon WorkSpaces Client Issues
- Amazon WorkSpaces Administration Guide
- Amazon WorkSpaces Developer Guide
- Amazon WorkSpaces Clients
- Managing Amazon Linux 2 Amazon WorkSpaces with AWS OpsWorks for Puppet Enterprise
- Customizing the Amazon Linux WorkSpace
- How Amazon WorkSpaces Use AWS KMS
- AWS CLI Command Reference – WorkSpaces
- Monitoring Amazon WorkSpaces Metrics
- MATE Desktop Environment

Contributors

Contributors to this document include:

- Andrew Wood, Senior Specialized SA for End-User Compute, AWS
- Harshitha Putta, Cloud Infrastructure Architect, AWS

Document Revisions

Date	Description
February 2020	First publication