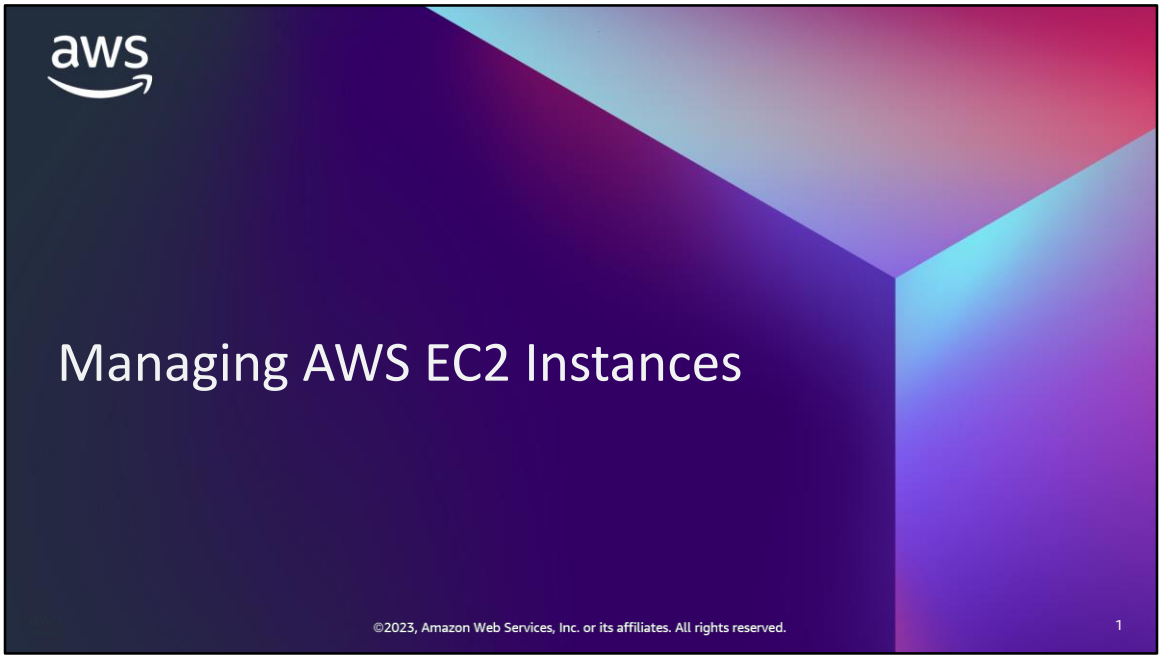
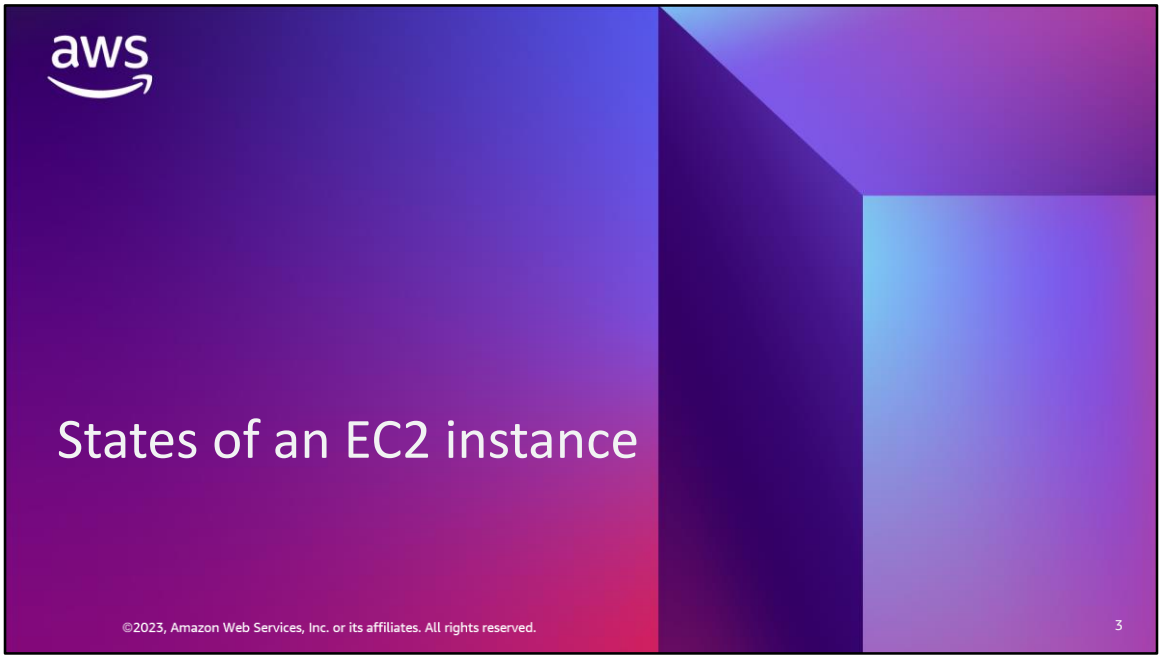


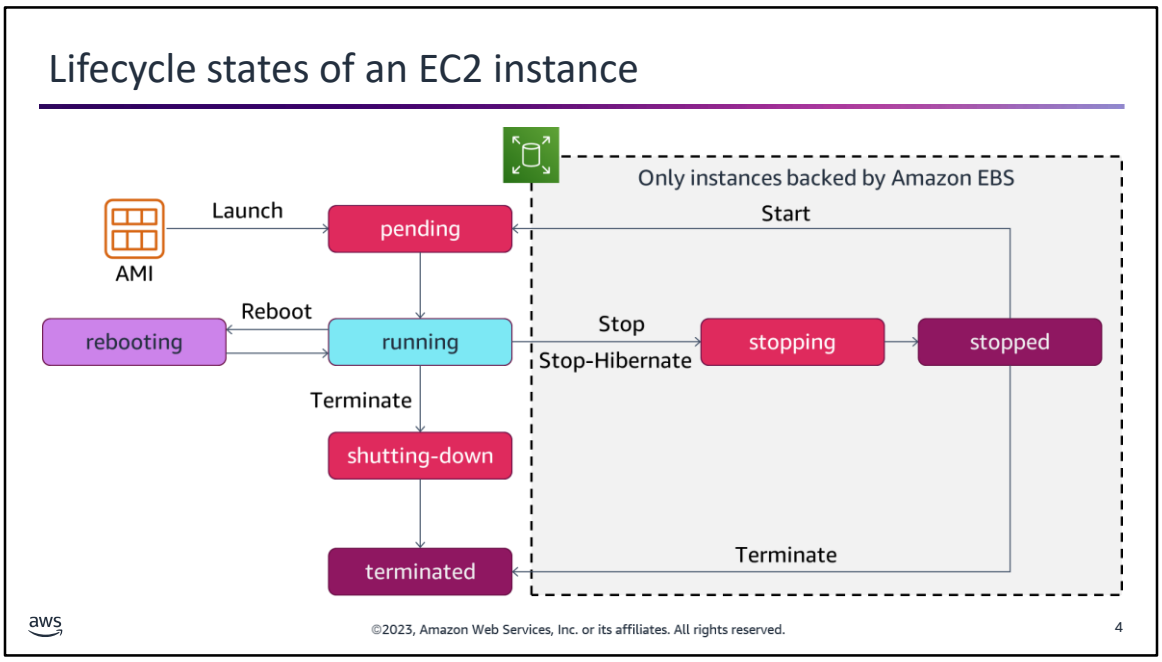
Managing AWS EC2 Instances

At the core of the lesson

You will learn how to do the following:

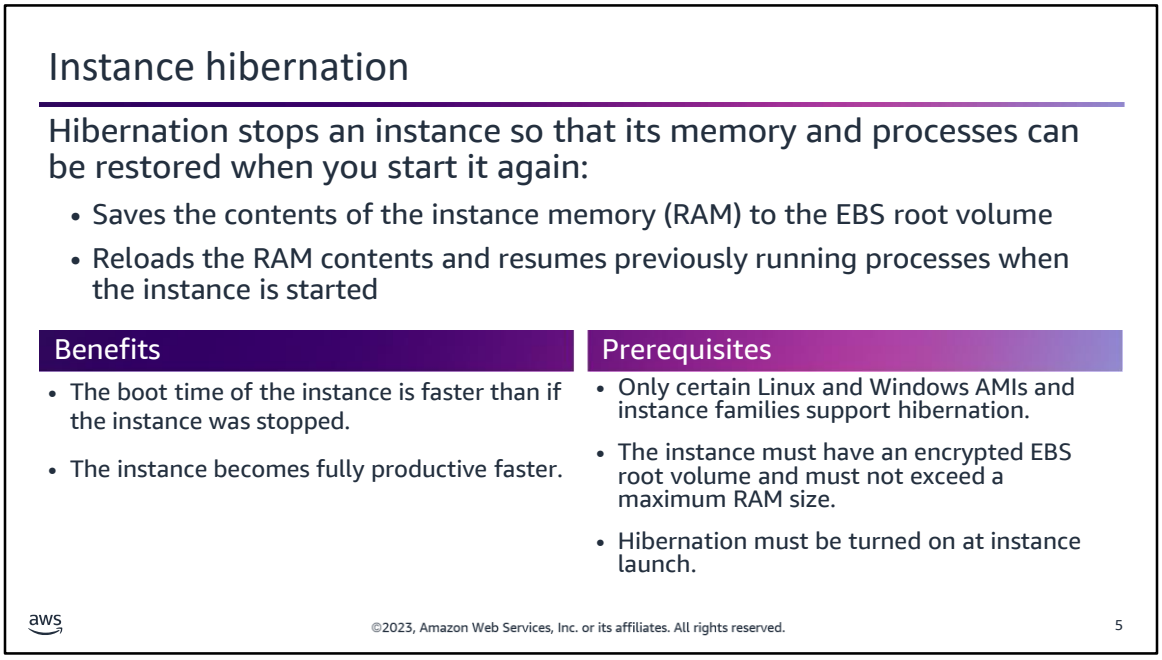
- Define the states of an Amazon Elastic Compute Cloud (Amazon EC2) instance and their effects.
- Modify an EC2 instance.


aws
States of an EC2 instance
©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.
3



This diagram shows the lifecycle of an instance. The arrows show actions that you can take, and the boxes show the state that the instance will enter after that action. An instance can be in one of the following states:

- Pending: When an instance is first launched from an Amazon Machine Image (AMI) or when you start a stopped instance, it first enters the pending state. At this point, the instance is booted and deployed to a host computer. The instance type that you specified at launch determines the hardware of the host computer for your instance.
- Running: When the instance is fully booted and ready, it exits the pending state and enters the running state. At this time, you can connect over the internet to your running instance.
- Rebooting: An instance temporarily goes into a rebooting state as a result of a reboot action. AWS recommends that you reboot an instance by using the Amazon EC2 console, AWS Command Line Interface (AWS CLI), or AWS SDKs instead of invoking a reboot from within the guest operating system (OS). A rebooted instance stays on the same physical host and maintains the same public Domain Name System (DNS) name and public IP address. If the instance has instance store volumes, they retain their data.
- Shutting-down: This state is an intermediary state between running and terminated. A terminate action on the instance initiates this state.
- Terminated: An instance reaches the terminated state as the result of a terminate action. A terminated instance remains visible in the Amazon EC2 console until the virtual machine is deleted. However, you can't connect to or recover a terminated instance.
- Stopping: Instances that are backed by Amazon Elastic Block Store (Amazon EBS) can be stopped or hibernated. They enter the stopping state before they attain the fully stopped state.
- Stopped: A stopped instance is not billed for usage. While the instance is in the stopped state, you can modify certain attributes of the instance (for example, the instance type). Starting a stopped instance puts it back into the pending state, which typically moves the instance to a new host machine. You can also terminate a stopped instance.

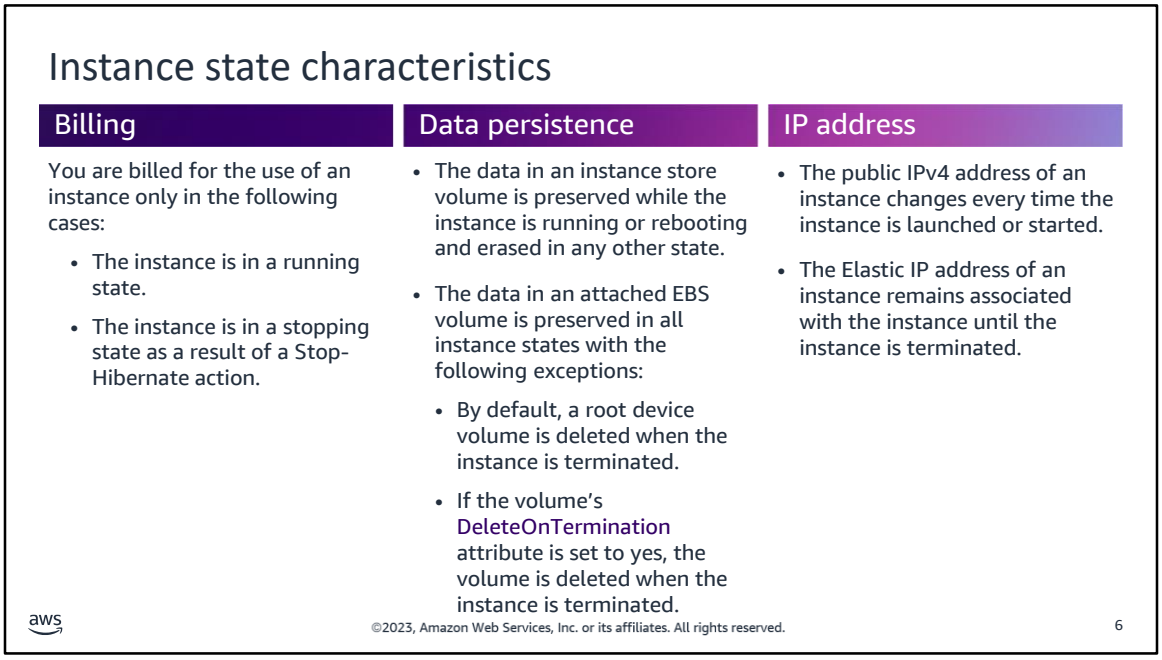


Some instances that are backed by Amazon EBS support hibernation. When you hibernate an instance, the guest OS saves the contents from the instance memory (RAM) to the EBS root volume.

When you restart the instance, the following actions occur:
- The EBS root volume is restored to its previous state.
- The RAM contents are reloaded.
- The processes that were previously running on the instance are resumed.
- Previously attached data volumes are reattached, and the instance retains its instance ID.

Hibernation is useful when you have an instance that you must quickly restart but that takes a long time to warm up if you stop and start it.

The hibernation feature is available for On-Demand Instances, and only certain Linux and Windows AMIs and certain instance types support hibernation. Hibernation also requires that you encrypt the root EBS volume and has a restriction on the maximum RAM size of the instance. The maximum RAM size is 150 GB for Linux and 16 GB for Windows. In addition, you must turn on hibernation when the instance is first launched. You cannot turn on hibernation on an existing instance after it is launched.

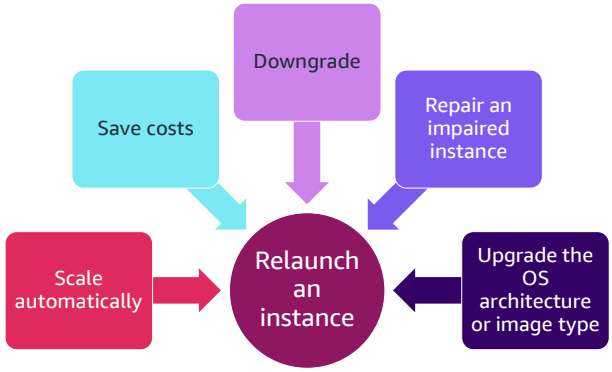


This slide highlights important characteristics of instance states and the effects of instance state changes on the components that are associated with an instance.

For example, it emphasizes that when you hibernate an instance, you continue to incur a usage charge while the instance is preparing to hibernate. In addition, data on instance store volumes is temporary, but data on EBS volumes is persistent. This difference is important to remember to avoid accidental data loss due to the incorrect placement of data. Similarly, an instance will get a new public IPv4 address every time it is launched or started, but any Elastic IP address will remain associated with the instance until the instance is terminated.

Instance design best practice

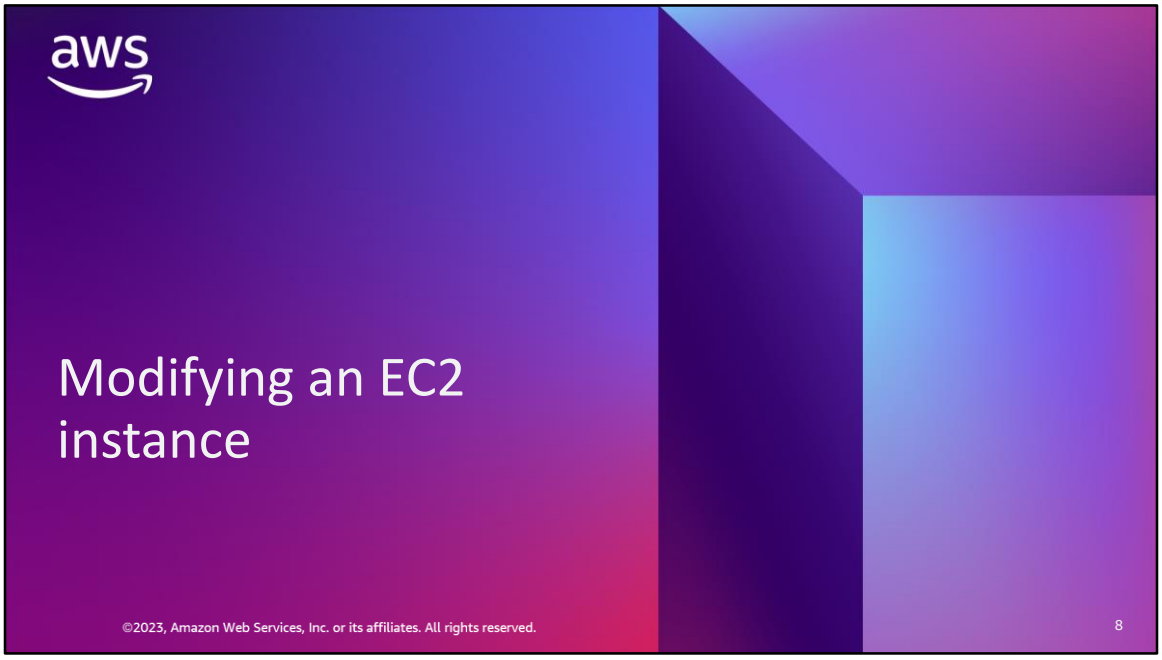
Design your instances for quick buildup and teardown because they may need to be relaunched often.



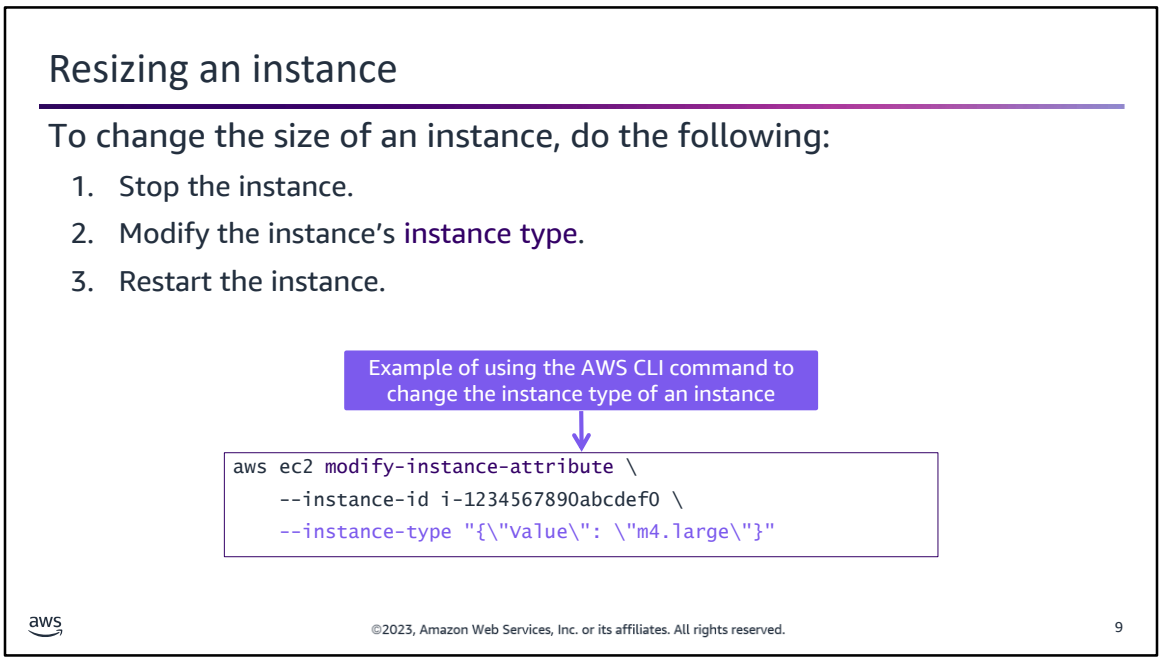
It is a best practice to regard EC2 instances as ephemeral resources that can be built up, torn down, and rebuilt at a moment's notice. If you are new to the cloud computing paradigm, this concept might be new to you.

Many situations in the cloud require building a new server, including the following:

- Automatic scaling: You might have solutions that must be able to deploy new instances without human intervention.
- Cost savings: You might decide that you do not need to keep an instance right now. However, perhaps you do need the ability to recreate it on a short notice. Batch processing use cases typically are in this category.
- Downgrading: You might want to downgrade an instance to save on costs. For example, you can downgrade an instance that runs on hardware that is dedicated to you, a single customer, to hardware that has shared tenancy. Alternatively, you might want downgrade the size of an instance (for example, from t2.xlarge to t2.large).
- Repairing impaired instances: The underlying hardware supporting an EC2 instance can fail. Booting an instance will place the new EC2 instance on healthy infrastructure.
- Upgrading: Upgrading the OS architecture or image type might require you to launch a new instance.



Modifying an EC2 instance



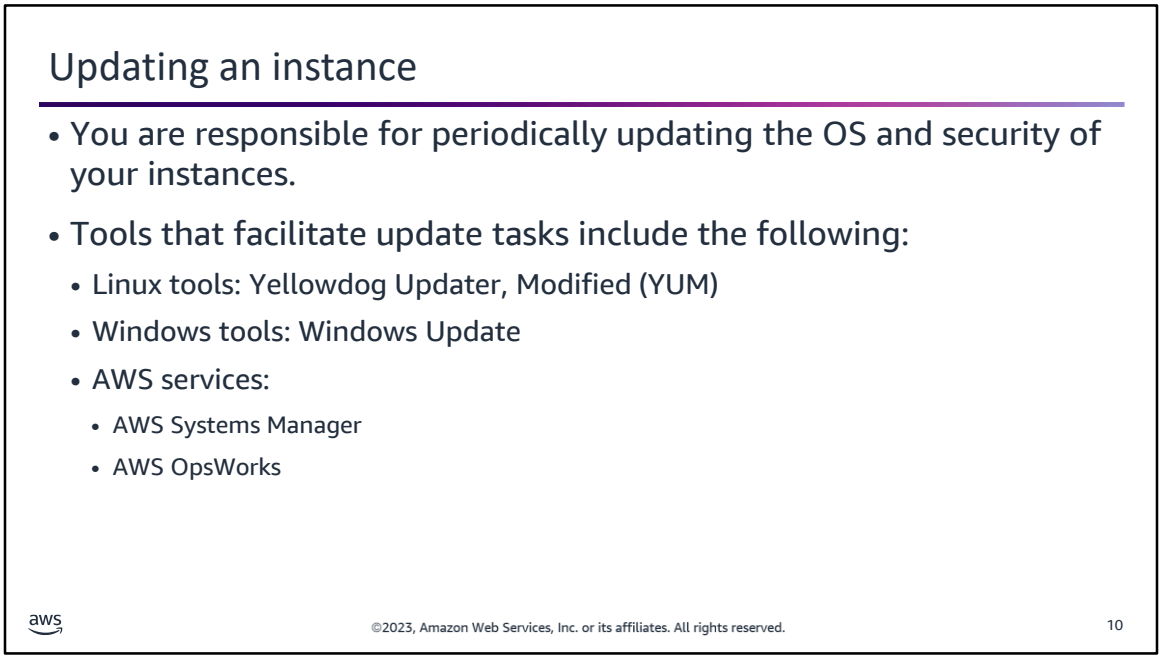
As your application needs change, you might find the following about the state of your instance:

- Your instance is overused because the instance type is too small. For example, workloads that run on the instance run out of memory or cause the CPU utilization to stay at 100 percent.
- Your instance is underused because the instance type is too large. For example, workloads that run on the instance do not use all the allocated memory or CPU processing power.

If you experience either situation, you can modify the size of the instance by changing its instance type. For example, if a t2.micro instance is too small for your workload, you can change it to a t2.large instance. When you update the instance type, the new instance type must support the same architecture. For example, an instance with a 64-bit architecture can only be updated to a different instance type that also has a 64-bit architecture.

Note that you can change the instance type of an existing instance only if the root device is an EBS volume and the new instance type that you choose is compatible with the instance's current configuration. In this case, you must stop the instance to change its instance type. In any other condition, you will need to change the instance type by launching a new instance.

The example on the slide shows how to use the AWS CLI modify-instance-attribute command to change the instance type of an existing instance to m4.large.



Remember that you own your instances and that keeping them up to date is your responsibility.

Although many instances may be short-lived, some instances can run for weeks or even months. For example, you might have development and test environments, or database servers that must remain stable over time. In these situations, it is recommended that you frequently update the instances in these environments so that they have the most recent operating system and security updates.

You can use various tools to perform these update tasks, including the following:

- YUM: YUM is an open-source, Redhat Package Manager (RPM)-based package installer for Linux. AWS maintains its own YUM software repository for Amazon Linux instances. You can run YUM to install the most recent available updates for all applications that are currently on your server. You can also use it to install updates one at a time if you reference only specific package names.
- Microsoft Windows Update: Window Update can install updates automatically. You can also choose to turn off automatic updates and install updates manually. In this way, you can test recent updates to ensure that they don’t break any existing applications that might be running on the server.
- AWS Systems Manager and AWS OpsWorks: These AWS services offer features that simplify updating EC2 instances.

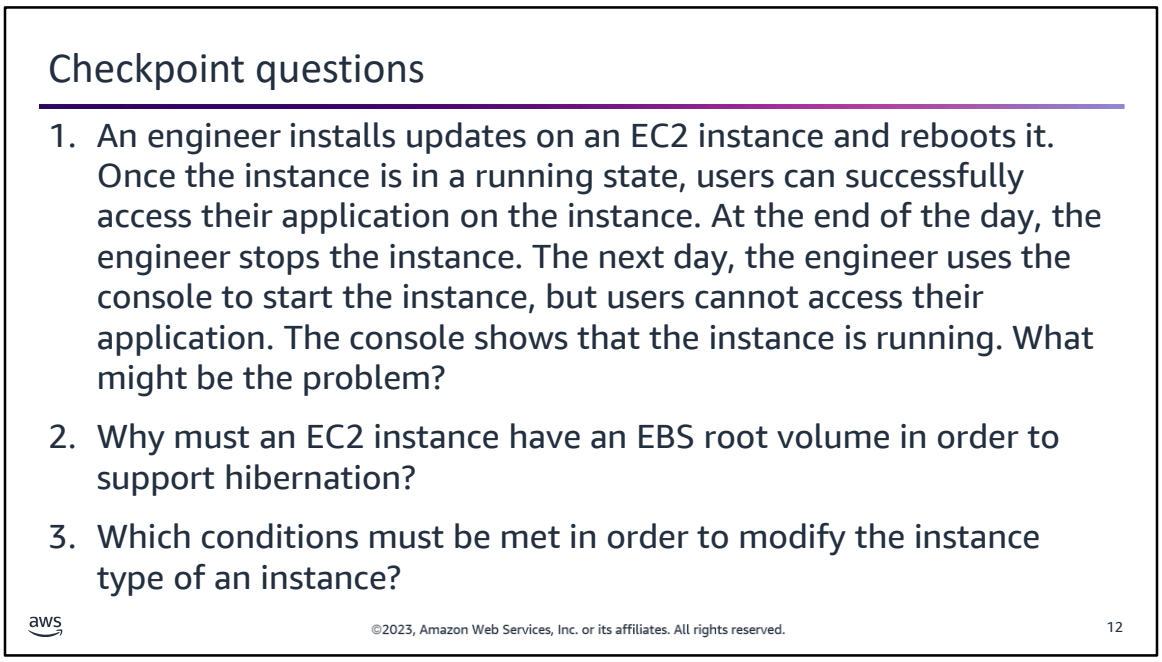
AMI deprecation

- Consider the deprecation date of an AMI to keep your instances up to date.
- The deprecation date of public AMIs is 2 years from the AMI creation date.
- Deprecating an AMI results in the following:
  - Instances that were launched using the AMI before the deprecation date are not affected.
  - No new instances can be launched using the AMI in the Amazon EC2 console.
  - You can continue to launch instances using the AMI by using the AWS CLI, Amazon EC2 API, or the AWS SDKs.

aws

AWS regularly updates public AMIs to provide a stable and secure environment for running applications on Amazon EC2. For example, AWS updates Linux AMIs on a regular basis to include the latest components of pre-installed packages. Likewise, AWS provides updated and fully patched Windows AMIs within 5 business days of when Microsoft releases a patch. However, when an AMI is out of date and should not be used, AWS deprecates it. By default, the deprecation date of a public AMI is 2 years after its creation date.

After an AMI is deprecated, you can no longer find that AMI in the console. However, by using the AWS CLI, Amazon EC2 API, or an AWS SDK, you can still retrieve the AMI ID and use it to launch an instance. In addition, instances that were launched using the AMI before it was deprecated can still be used, stopped, started, and rebooted.



The answers to the questions are as follows:

1. An engineer installs updates on an EC2 instance and reboots it. Once the instance is in a running state, users can successfully access their application on the instance. At the end of the day, the engineer stops the instance. The next day, the engineer uses the console to start the instance, but users cannot access their application. The console shows that the instance is running. What might be the problem?

   When the engineer rebooted the instance, it retained the public IP address that it was assigned. When the engineer stopped the instance, the public IP address that was initially assigned was released. If the users are using the IP address from the previous day, they cannot access their application.

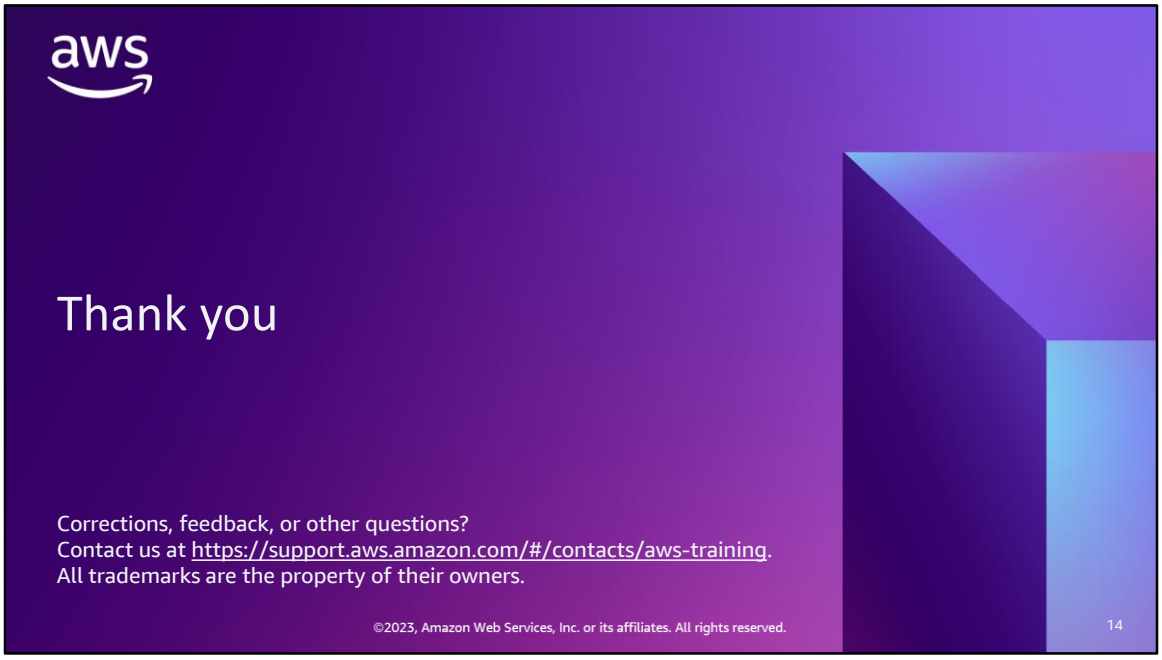
2. Why must an EC2 instance have an EBS root volume in order to support hibernation?

   EBS volumes persist between shutdowns. When an EC2 instance is hibernated, its memory is stored to the persistent EBS root volume before the instance is stopped. Then when the instance is started, it recovers working memory from the EBS volume and continues as if the shutdown did not occur.

3. Which conditions must be met in order to modify the instance type of an instance?
   - The instance’s root device is an EBS volume.
   - The new instance type that you choose is compatible with the instance's current configuration. For example, if the current instance type has a 64-bit architecture, the new instance type also has to have a 64-bit architecture.
   - The instance must first be stopped.

Key ideas

- The following are the different states of an EC2 instance:
  - Pending
  - Running
  - Rebooting
  - Shutting-down
  - Stopping
  - Stopped
  - Terminated
- Only instances that are backed by Amazon EBS can be stopped or hibernated.
- Design instances for quick buildup and teardown so that they can be relaunched faster.


aws
Thank you
Corrections, feedback, or other questions?
Contact us at https://support.aws.amazon.com/#/contacts/aws-training.
All trademarks are the property of their owners.
©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.
14