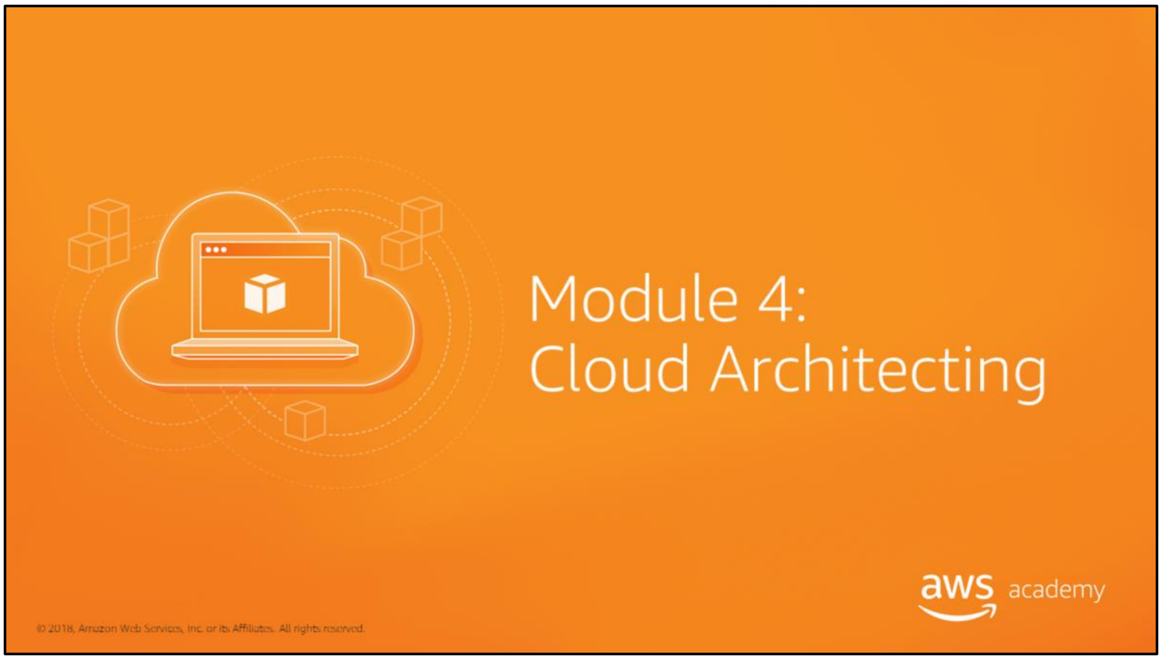
Academy Cloud Foundations (ACF) Module 04 Student Guide Version 1.0.5

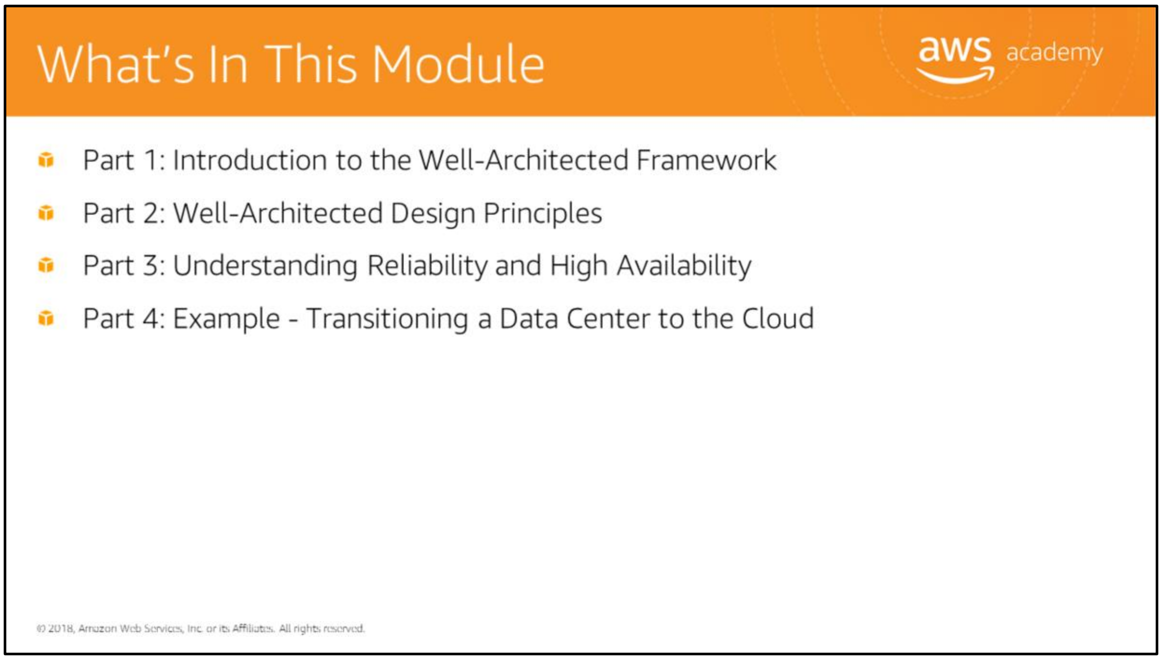
100-ACFNDS-10-EN-SG

Contents



Module 4: Cloud Architecting

Welcome to Module 4: Cloud Architecting Essentials.



What's In This Module

- Part 1: Introduction to the Well-Architected Framework
- Part 2: Well-Architected Design Principles
- Part 3: Understanding Reliability and High Availability
- Part 4: Example - Transitioning a Data Center to the Cloud

In this module, we'll review the well-architected design principles and explore a reference architecture that shows an example of these principles in action.

In part one, we'll review the architectural pillars and design decisions

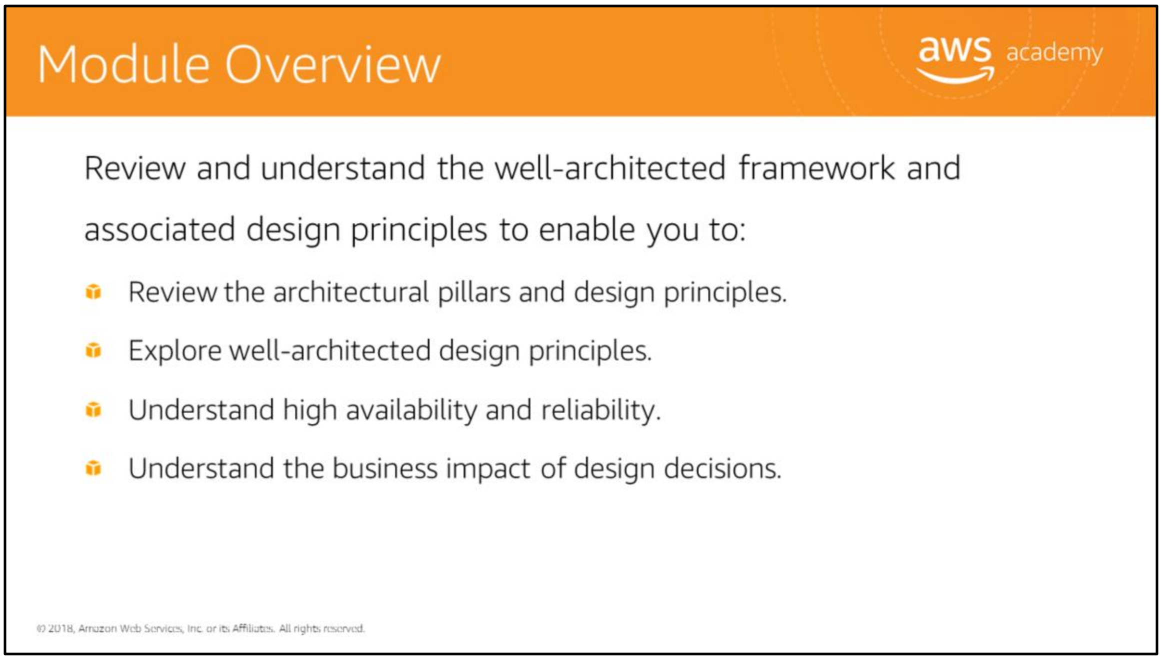
In part two, we'll explore well-architected design principles to help you understand the pros and cons of decisions made while building systems on AWS.

In part three, we'll develop an understanding of how to incorporate high availability and reliability into a cloud architecture

In part four, we'll understand the business impact of design decisions when transitioning a data center for the cloud as we explore questions developed by AWS architecture experts to help customers analyze and critically think about their architecture to determine whether their infrastructure is following best practices.

Architecture is the art and science of designing and building large structures. Large systems, whether buildings, bridges, novels, hardware, or software, require architectures to manage their size and complexity. Architectures are primarily concerned with structures and the interrelationship of the components that are used to build those structures.

Having well-architected systems greatly increases the likelihood of business success. In this module, we will explore the architectural best practices for designing and operating reliable, secure, efficient, and cost-effective systems in the cloud.



Module Overview

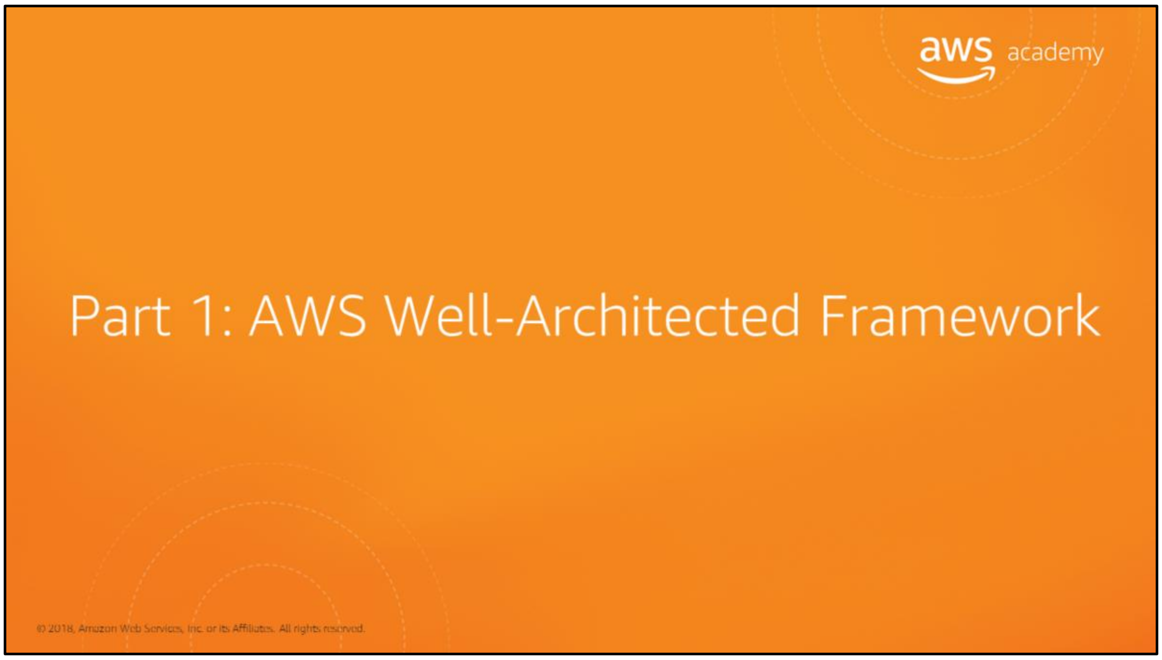
Review and understand the well-architected framework and associated design principles to enable you to:

- Review the architectural pillars and design principles.
- Explore well-architected design principles.
- Understand high availability and reliability.
- Understand the business impact of design decisions.

The goal of this module is to introduce you to some foundational cloud architecting concepts. We'll review and understand the well-architected framework and associated design principles to enable you to:

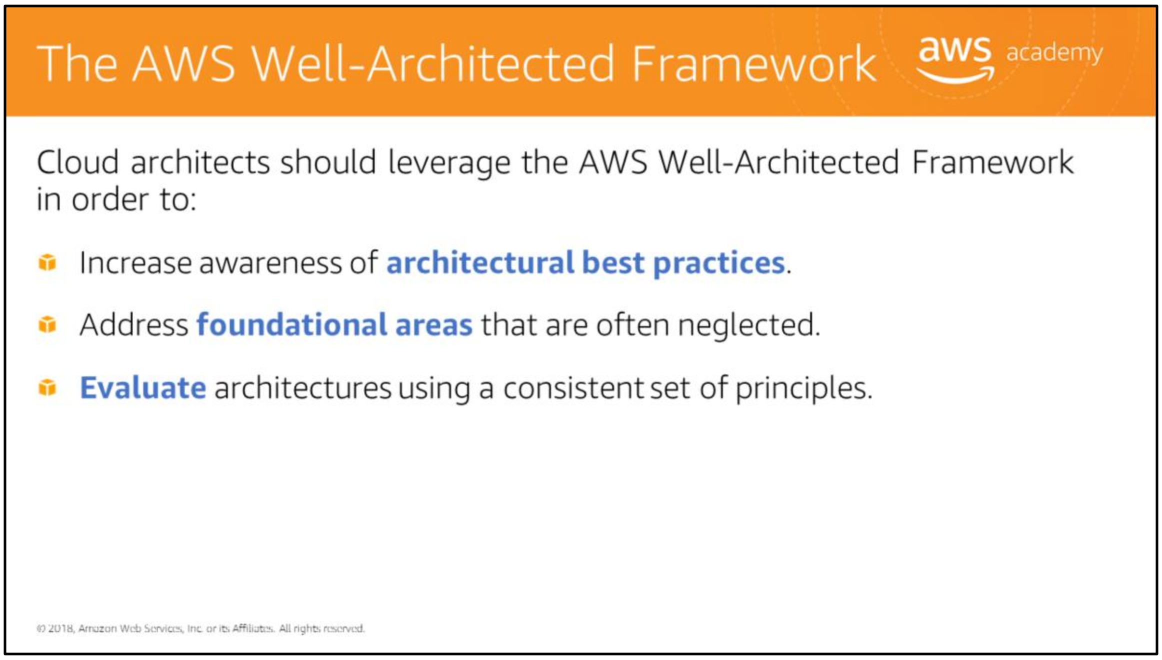
- Review the architectural pillars and design principles.
- Explore well-architected design principles.
- Understand high availability and reliability.
- Understand the business impact of design decisions.

Architecture is the art and science of designing and building large structures. This knowledge will help you begin to understand all of the considerations of creating a well-architected cloud solution.



Part 1: AWS Well-Architected Framework

Introducing Part 1: AWS Well-Architected Framework.



The AWS Well-Architected Framework

Cloud architects should leverage the AWS Well-Architected Framework in order to:

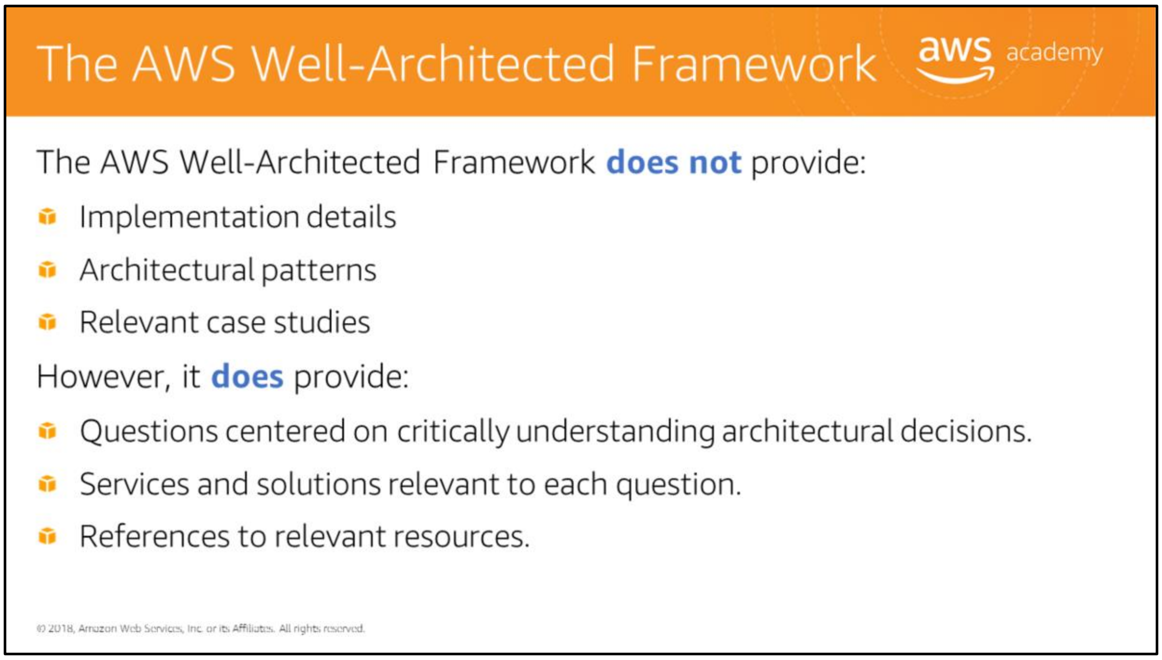
- Increase awareness of **architectural best practices**.
- Address **foundational areas** that are often neglected.
- **Evaluate** architectures using a consistent set of principles.

The AWS Well-Architected Framework documents a set of foundational questions that allow you to understand whether a specific architecture aligns well with cloud best practices. The framework provides a consistent approach to evaluating systems against the qualities you expect from modern cloud-based systems and the remediation that would be required to achieve those qualities.

The AWS Well-Architected Framework helps cloud architects assess and improve their own architectures, all while getting a better understanding of how their design decisions can impact their business.

- It provides a **set of questions developed by AWS experts** to help customers think critically about their architecture.
- It asks, **"Does your infrastructure follow best practices?"**

Note that as cloud technologies continue to evolve and as AWS continues to learn more from working with customers, the definition of well-architected is continually being improved and refined.



The AWS Well-Architected Framework

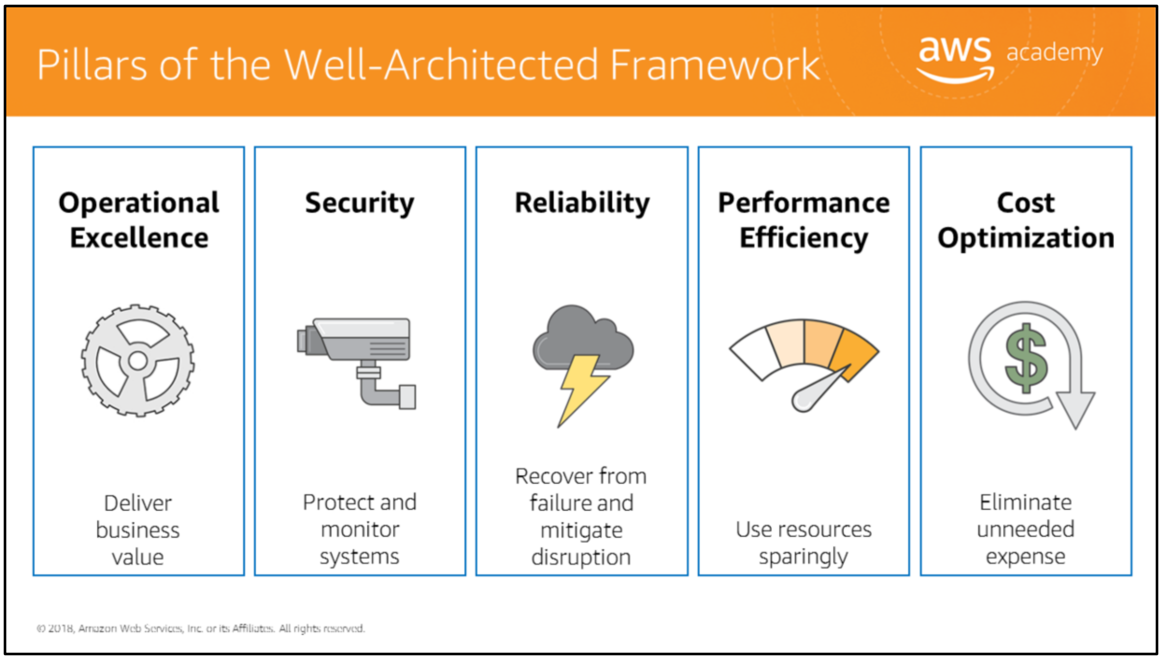
The AWS Well-Architected Framework **does not** provide:

- Implementation details
- Architectural patterns
- Relevant case studies

However, it **does** provide:

- Questions centered on critically understanding architectural decisions.
- Services and solutions relevant to each question.
- References to relevant resources.

The AWS Well-Architected Framework does not provide implementation details or architectural patterns. However, it does provide questions a set of foundational questions that allow you to understand if a specific architecture aligns well with cloud best practices. It also includes information about services and solutions that are relevant to each question and references to relevant resources.

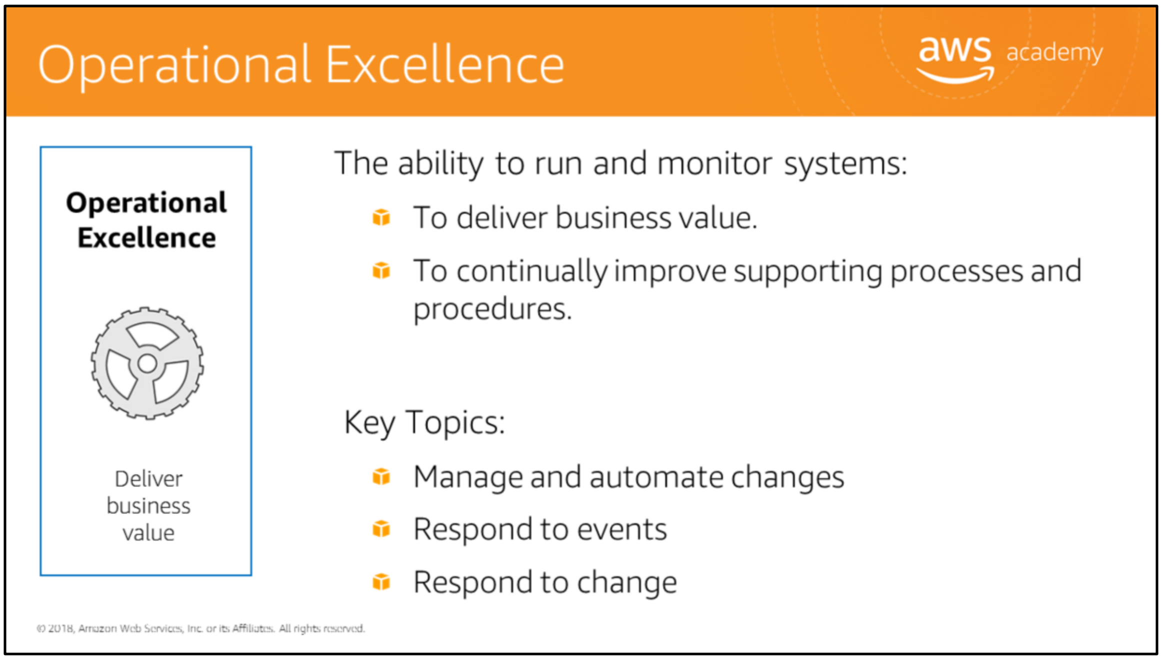


Pillars of the Well-Architected Framework

Operational Excellence	Security	Reliability	Performance Efficiency	Cost Optimization
Deliver business value	Protect and monitor systems	Recover from failure and mitigate disruption	Use resources sparingly	Eliminate unneeded expense

AWS has developed a guide to help you with the design of your architecture from five different perspectives or **pillars**. The pillars are operation excellence, security, reliability, performance efficiency, and cost optimization.

We will go over in more detail each of the pillars and discuss the design principles for each pillar.



Operational Excellence

aws academy

Operational Excellence

Deliver business value

The ability to run and monitor systems:

- To deliver business value.
- To continually improve supporting processes and procedures.

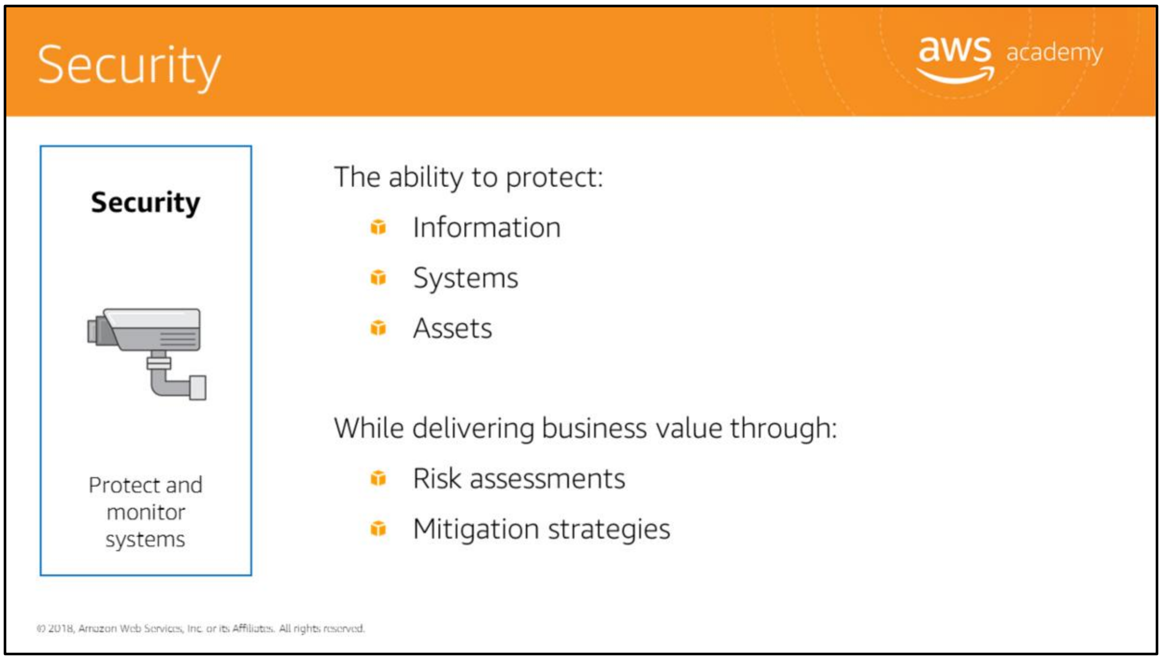
Key Topics:

- Manage and automate changes
- Respond to events
- Respond to change

The first pillar is the operational excellence pillar. This pillar focuses on running and monitoring systems to deliver business value, and continually improving supporting processes and procedures.

Key topics include managing and automating changes, responding to events, and defining standards to successfully manage daily operations.

For more information, select the link to view "The Operational Excellence Pillar" whitepaper. https://d0.awsstatic.com/whitepapers/architecture/AWS-Operational-Excellence-Pillar.pdf.



Security

Security

Protect and monitor systems

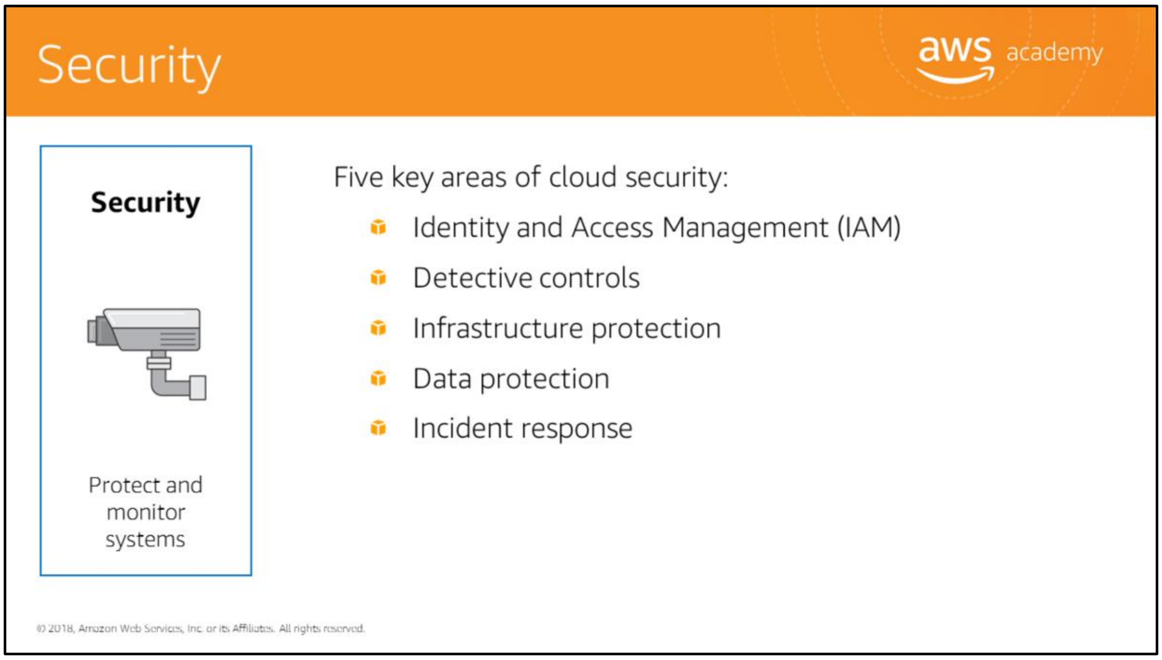
The ability to protect:

- Information
- Systems
- Assets

While delivering business value through:

- Risk assessments
- Mitigation strategies

The security pillar encompasses the ability to protect your information, systems, and assets. It accomplishes this while delivering business value through risk assessments and mitigation strategies.



Security

Security

Protect and monitor systems

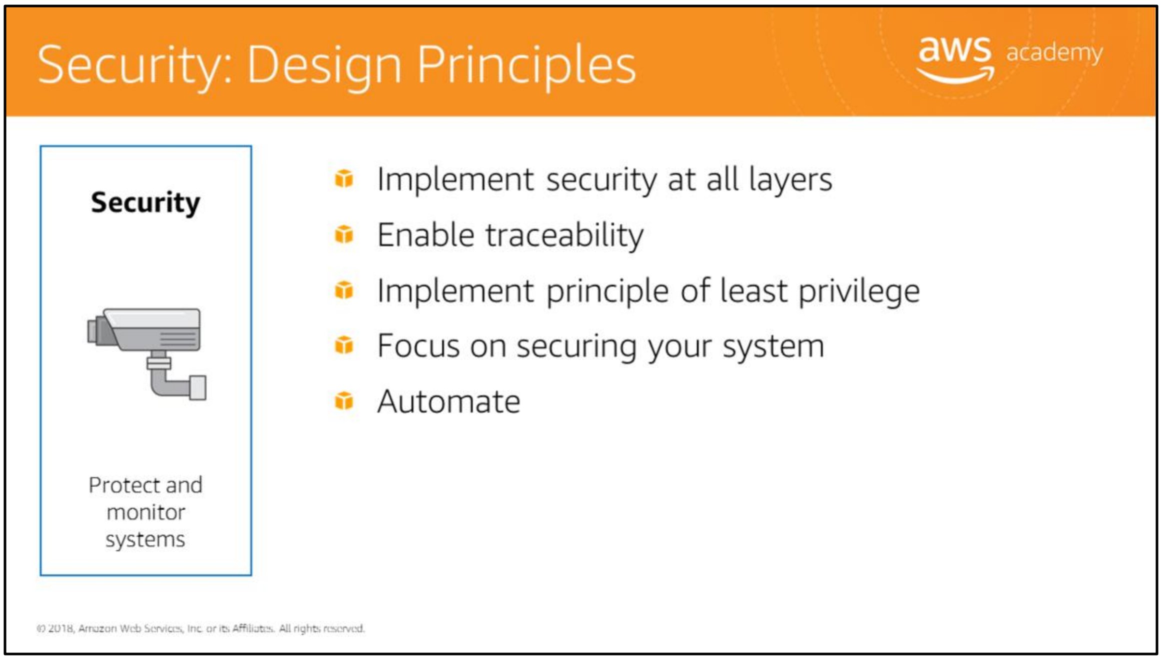
Five key areas of cloud security:

- Identity and Access Management (IAM)
- Detective controls
- Infrastructure protection
- Data protection
- Incident response

To dive a little deeper, cloud security is composed of five areas. Let's briefly look at each area:

1. **Identity and Access Management (IAM):** IAM is critical to ensure that only authorized and authenticated users are able to access your resources and only in the manner you intend.
2. **Detective controls:** Detective controls can be used to identify a potential security incident by considering some approaches such as capturing or analyzing logs and integrating auditing controls.
3. **Infrastructure protection:** Infrastructure protection ensures that systems and services within your architecture are protected against unintended and unauthorized access. For instance, the user can create network boundaries, hardening and patching, users/keys/access levels and application firewalls or gateways.
4. **Data protection:** With data protection, there are numerous approaches and methods to consider. Some of them include data classification, encryption, protecting data at rest and in transit and data backup, replication, and recovery when needed.
5. **Incident response:** Even with all the preventative and detective measures, organizations should still create an incident response process to respond and mitigate any potential

security incidents. Incident response will ensure that your architecture is updated to accommodate a timely investigation and recovery.



Security: Design Principles

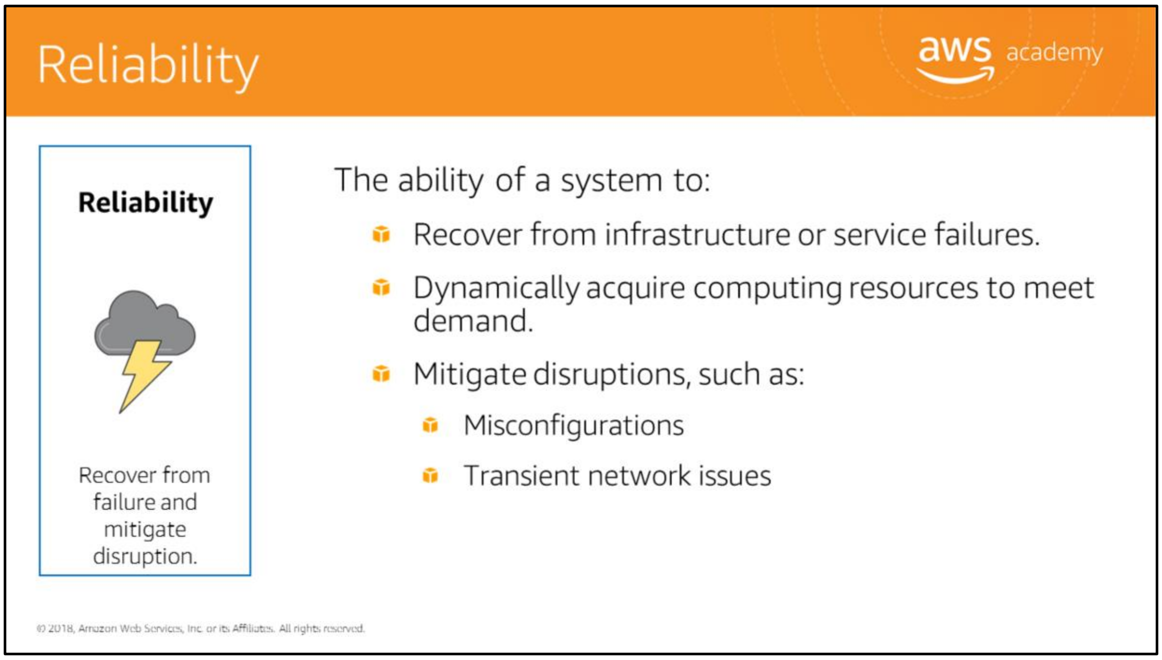
aws academy
Security
Protect and monitor systems

- Implement security at all layers
- Enable traceability
- Implement principle of least privilege
- Focus on securing your system
- Automate

When you are architecting, it's important to consider specific design principles to help you strengthen your security, which include:

1. **Apply security at all layers:** You want to make sure that have multiple layers of defense by securing your infrastructure **everywhere and at every layer.** In a physical data center, security is typically considered only at the perimeter. With AWS, you can implement security at the perimeter and within and between your resources. This ensures that your individual environment and components are secured from each other as well.
2. **Enable traceability:** Enable traceability through logging and auditing all actions or changes to your environment.
3. **Implement principle of least privilege:** Another useful design principle is the **principle of least privilege.** Make sure that authorization within your environment is adequate and that you are implementing strong logical access controls to your AWS resources that grants minimum privileges required for business requirements.
4. **Secure your system: Focus on securing your system.** With the AWS shared responsibility model, you can focus clearly on securing your application, data, and operating systems while AWS provides secure infrastructure and services.
5. **Automate security best practices:** Software-based security mechanisms improve your ability to securely scale more rapidly and cost-effectively. For example, create and save a patched, hardened image of a virtual server so that when you need an image, you can use that image

automatically to create a new instance. Another best practice is to automate the response to both routine and anomalous security events.



Reliability

Reliability

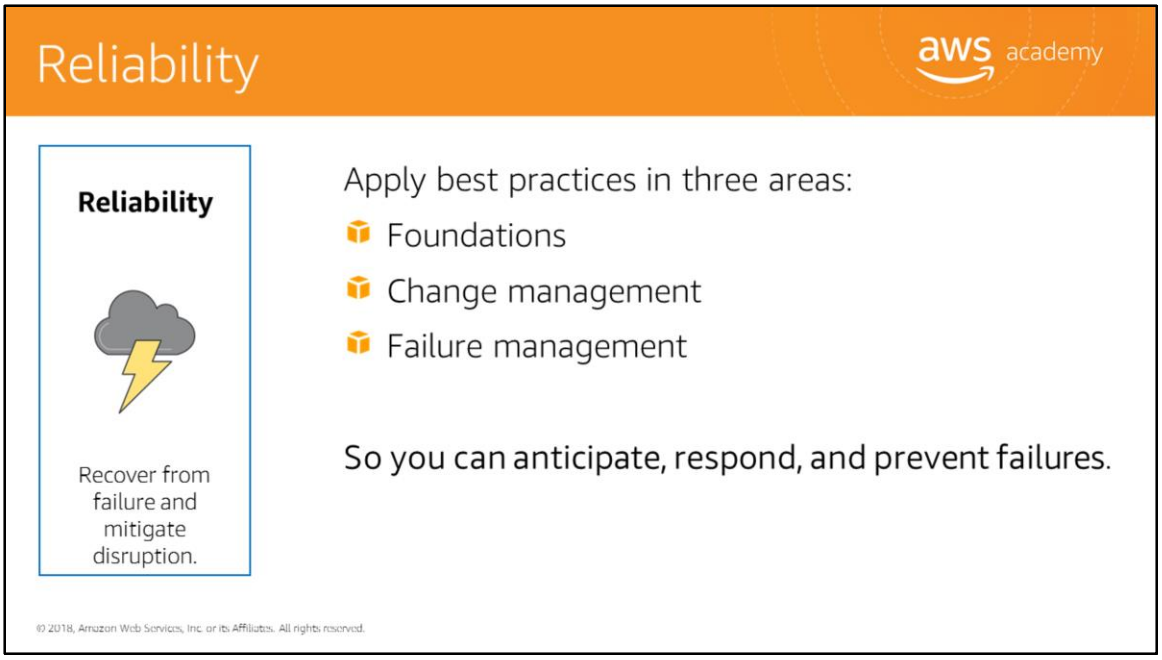
Recover from failure and mitigate disruption.

The ability of a system to:

- Recover from infrastructure or service failures.
- Dynamically acquire computing resources to meet demand.
- Mitigate disruptions, such as:
  - Misconfigurations
  - Transient network issues

The reliability pillar concerns the ability of a system to recover from infrastructure or service failures and dynamically acquire computing resources to meet demand and mitigate disruptions.

The bottom line is that reliability is there to assist in the ability to recover from failures and meet demand.



Reliability

Reliability

Recover from failure and mitigate disruption.

Apply best practices in three areas:

- Foundations
- Change management
- Failure management

So you can anticipate, respond, and prevent failures.

Reliability in the cloud is comprised of three areas: Foundations, Change Management, Failure Management

Foundations
In order to achieve reliability, your architecture and system must have a well-planned foundation in place that can handle changes in demand, or with requirements, and also detect failure and automatically heal itself.

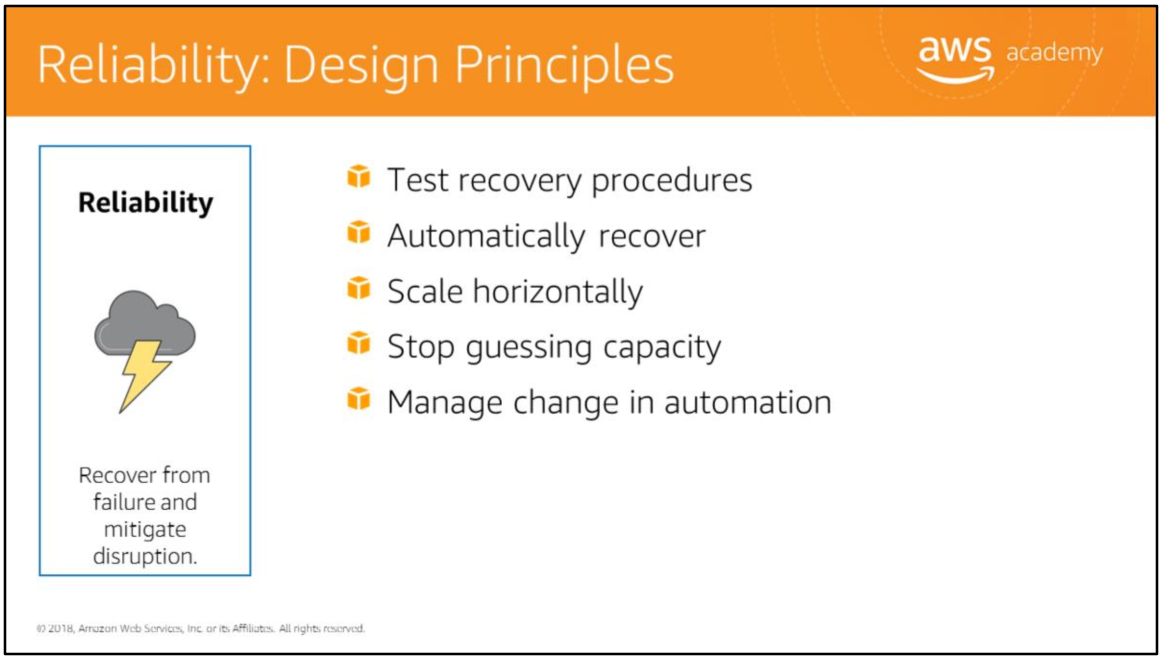
Before architecting any sort of structure, it's critical to look at the foundation. Before architecting ANY system, foundational requirements that influence reliability should be in place.

Change Management
With change management, it's important to fully understand how change can affect your system. If you plan proactively and monitor your systems, you can accommodate change and adjust to it quickly and reliably.

Failure Management
To really make sure your architecture is reliable, it's key to anticipate, become aware of, respond to, and prevent failures from happening. In a cloud environment, you can take

advantage of automation with monitoring, replace systems in your environment, and later troubleshoot failed systems, all at low cost, and all while still being reliable.

Careful evaluation of each of these elements will enable you to anticipate, respond to, and prevent failures.



Reliability: Design Principles

Reliability

Recover from failure and mitigate disruption.

- Test recovery procedures
- Automatically recover
- Scale horizontally
- Stop guessing capacity
- Manage change in automation

Several design principles can increase reliability:

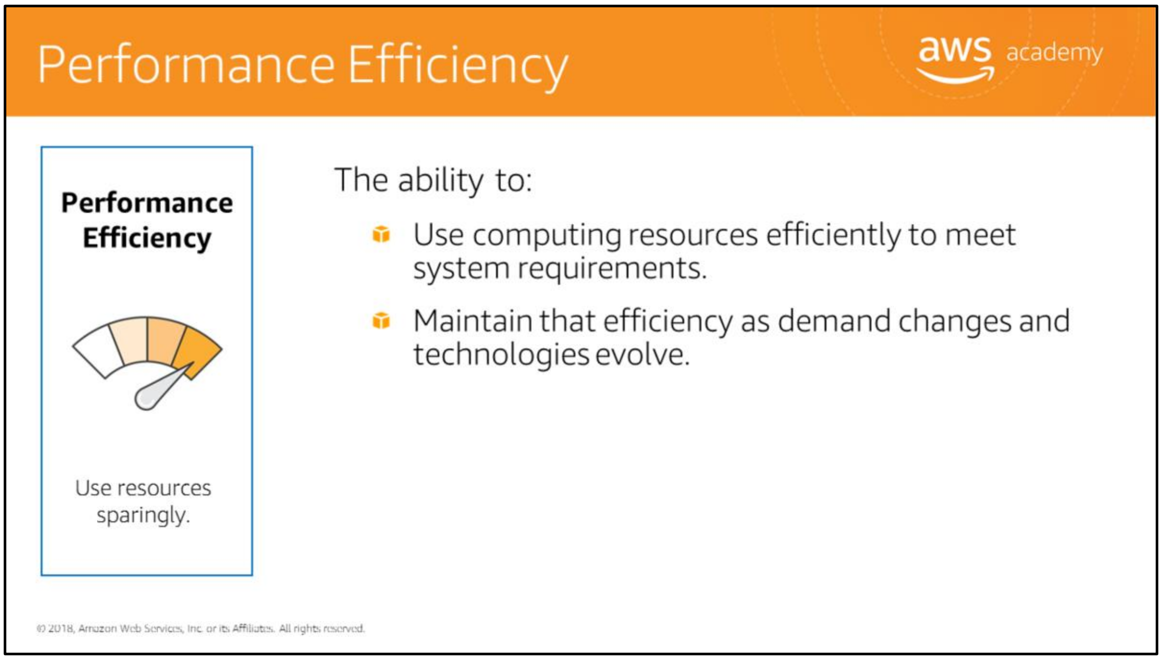
Testing recovery procedures
In the cloud, users have the ability to test how systems fail and can validate their recovery procedures. Users can simulate and expose difference failures and then rectify **before** a real failure occurs.

Automatically recover from failure
In AWS, users can trigger automated responses when thresholds are breached. This makes it possible to anticipate and remediate failures before they occur.

Scale horizontally to increase aggregate system availability.
When you have one large resource, it's beneficial to replace that large resource with multiple small resources to reduce the impact of a single point of failure on the overall system. The goal is to scale horizontally and distribute requires amongst the multiple small resources.

Stop guessing capacity
In the cloud environment, you have the ability to monitor demand and system utilization, and automate the addition or removal of resources. This ensures that you have the optimal level to satisfy your demand without over or under provisioning.

Manage change in automation
Changes to your architectures and infrastructure should be made using automation. With this, you only need to manage change to your automation not every single system or resource.



Performance Efficiency

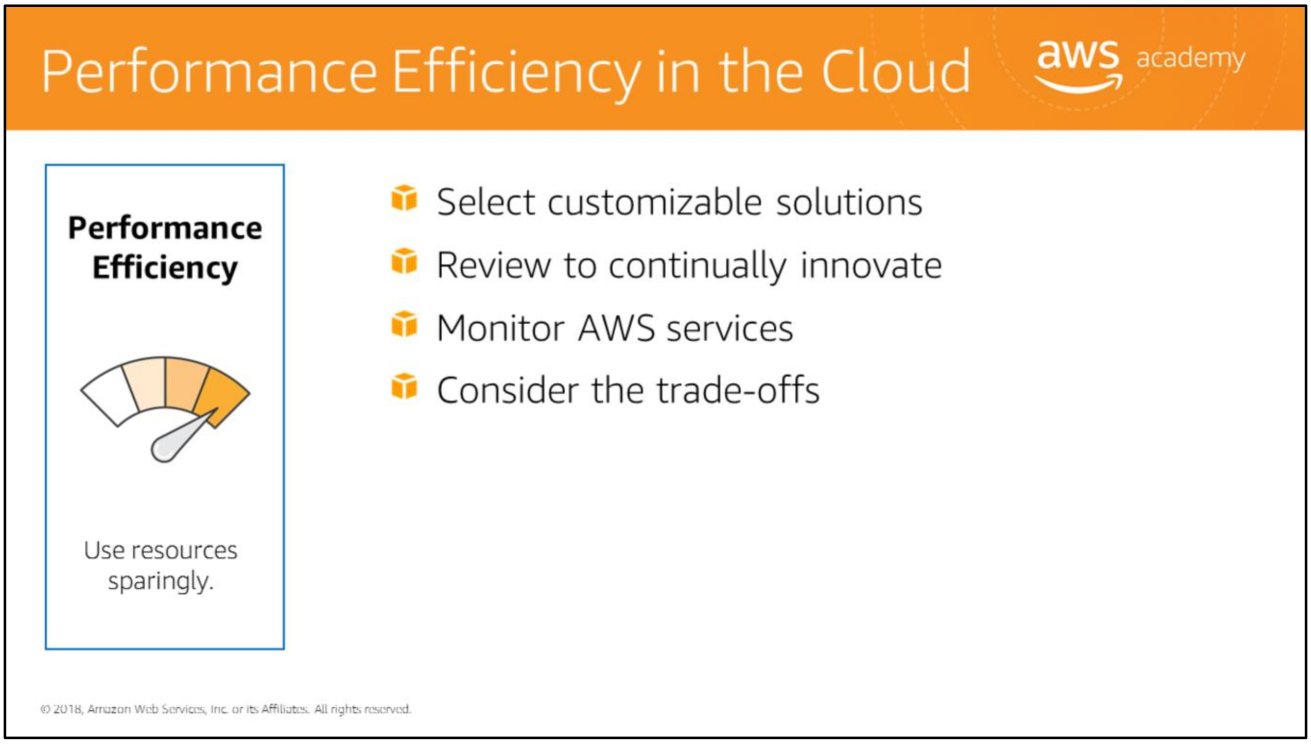
Performance Efficiency

Use resources sparingly.

The ability to:

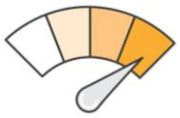
- Use computing resources efficiently to meet system requirements.
- Maintain that efficiency as demand changes and technologies evolve.

Now let’s look at the performance efficiency pillar. Performance efficiency refers to using computing resources efficiently while meeting system requirements. At the same time, it is important to maintain that efficiency as demand fluctuates and technologies evolve.



Performance Efficiency in the Cloud

Performance Efficiency



Use resources sparingly.

- Select customizable solutions
- Review to continually innovate
- Monitor AWS services
- Consider the trade-offs

The four pieces that make up performance efficiency in the cloud include:

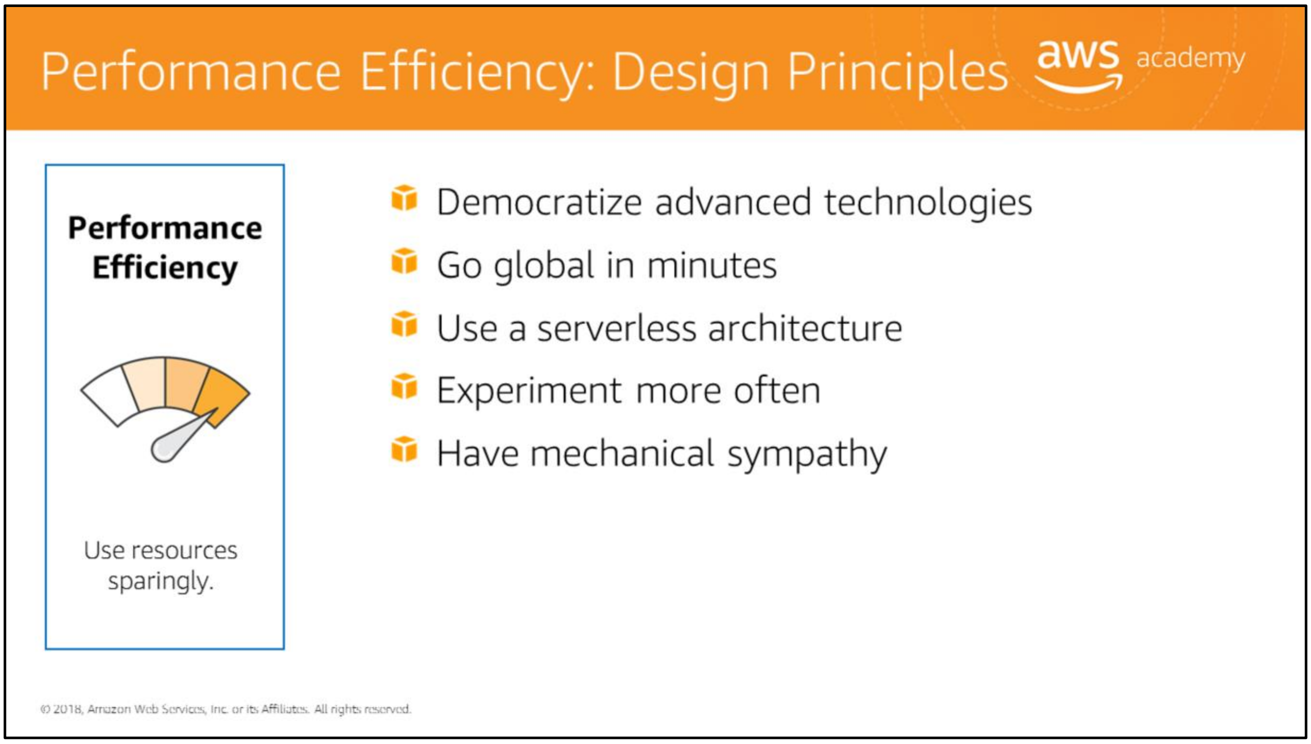
- Selection
- Review
- Monitoring
- Tradeoffs

Let's dive deeper into each area. With **selection**, it’s important to choose the best solution that will optimize your architecture. However, these solutions vary based on the kind of workload you have. With AWS, resources are virtualized and allow you to customize your solutions in many different types and configurations.

With **review**, you can continually innovate your solutions and take advantage of the newer technologies and approaches that become available. Any of these newer releases could improve the performance efficiency of your architecture.

In regards to **monitoring**, after you have implemented your architecture, you will need to monitor performance to ensure that you can remediate any issues before customers are affected and aware of them. With AWS, you can use automation and monitor your architecture with tools such as Amazon CloudWatch, Amazon Kinesis, Amazon Simple Queue Service, and AWS Lambda.

Finally, we have **tradeoffs**. An example of a trade-off that ensures optimal approach is trading consistency, durability and space versus time or latency, to deliver higher performance.



Performance Efficiency: Design Principles

Performance Efficiency

Use resources sparingly.

- Democratize advanced technologies
- Go global in minutes
- Use a serverless architecture
- Experiment more often
- Have mechanical sympathy

There are several design principles that can help you achieve performance efficiency:

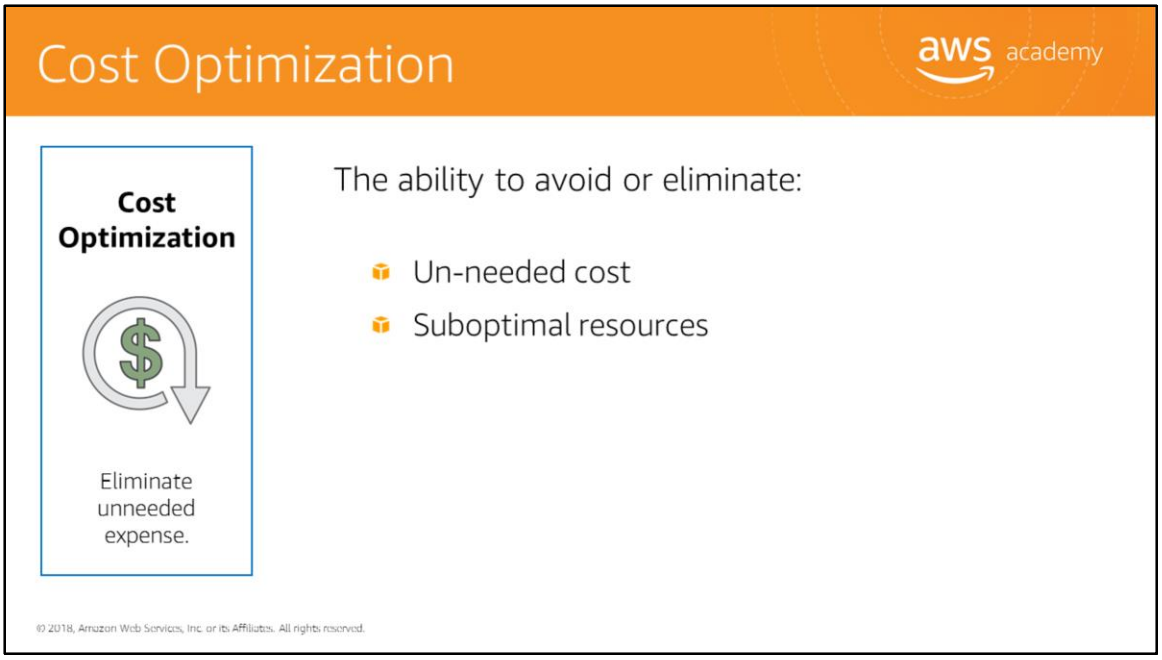
First, **democratize advanced technologies**.
Technologies that are difficult to implement can become simpler to consume by pushing that knowledge and complexity into the cloud vendor's domain. Instead of having your IT team learn how to host and run a new technology, they can consume it as a service.

Second, **go global in minutes**.
With AWS, you can easily deploy your system in multiple regions around the world while providing a lower latency and better experience for your customers at minimal cost.

Third, **use a serverless architecture**.
Serverless computing is a cloud computing execution model in which the cloud provider dynamically manages the allocation of machine resources. Pricing is based on the actual amount of resources consumed by an application, rather than on pre-purchased units of capacity. In the cloud, this enables you to remove the need to run and maintain traditional servers for compute activities. This also removes the operational burden and can lower transactional costs.

Fourth, **experiment more often!** With virtualization you can quickly carry out testing to enhance efficiency.

Finally, we have **mechanical sympathy**. This principle suggests that you use the technology approach that best aligns to what you are trying to achieve.



Cost Optimization

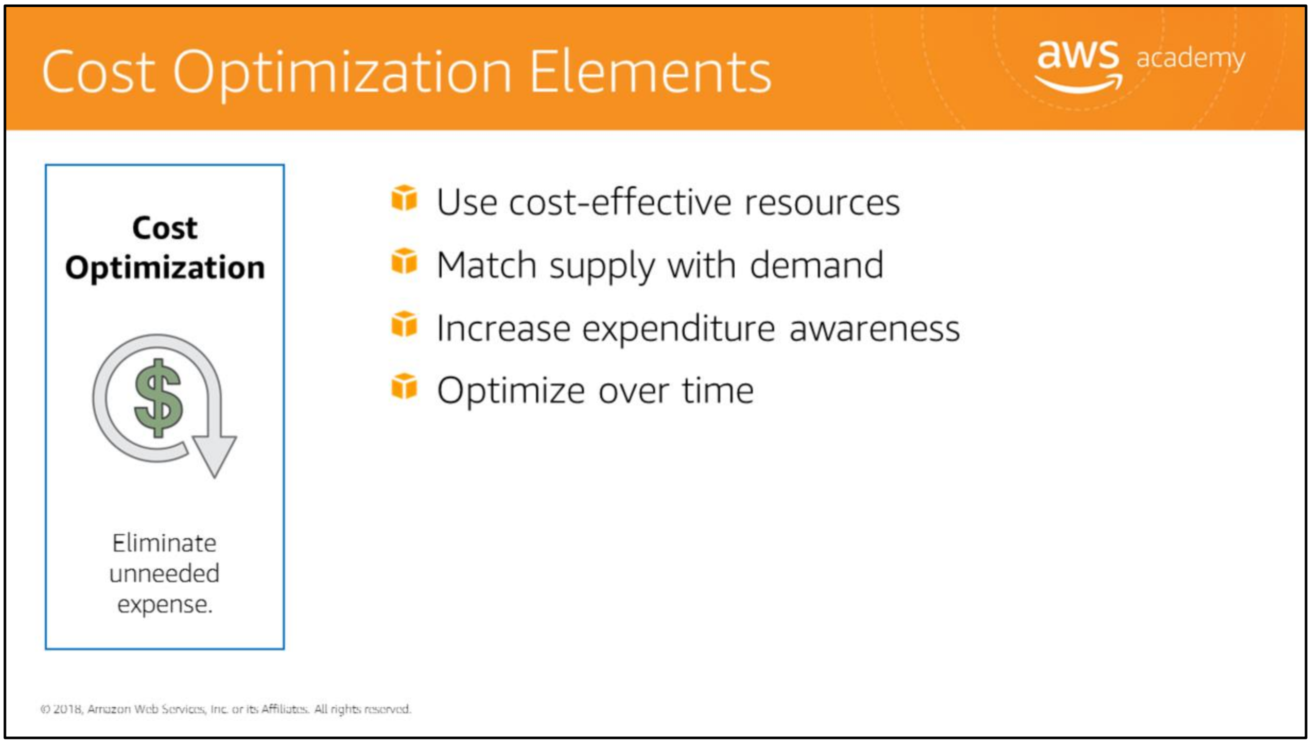
Cost Optimization

Eliminate unneeded expense.

The ability to avoid or eliminate:

- Un-needed cost
- Suboptimal resources

With cost optimization, you have the ability to avoid or eliminate un-needed costs and suboptimal resources.



Cost Optimization Elements

Cost Optimization

Eliminate unneeded expense.

- Use cost-effective resources
- Match supply with demand
- Increase expenditure awareness
- Optimize over time

The four areas that make up the cost optimization pillar include cost-effective resources, matching supply with demand, expenditure awareness and optimizing over time.

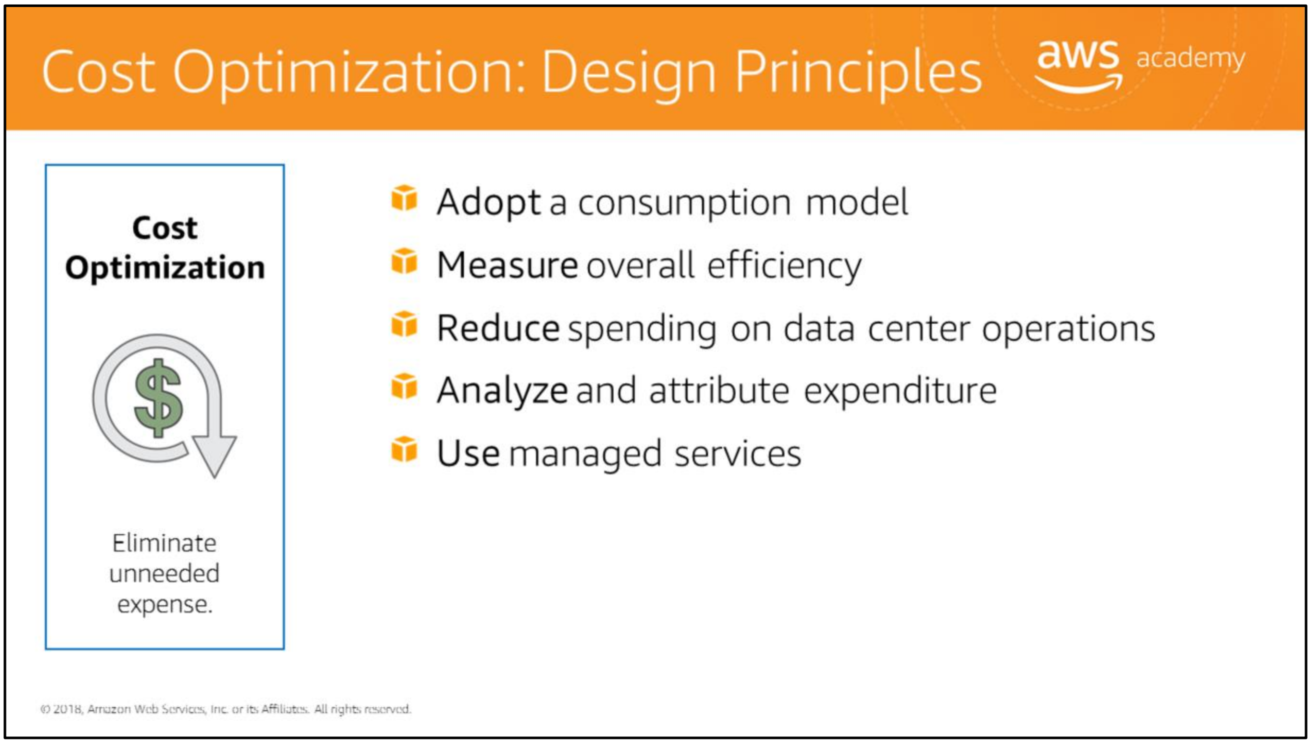
A fully cost-optimized system will use all resources to achieve the best outcome at the lowest possible price point, while still meeting your functional requirements.

Making sure that your systems are **using the appropriate services**, resources and configurations is one of the key parts to cost savings. As a user, you want to focus on the details such as provisioning, sizing, purchasing options and other specifics to ensure you have the best architecture for your needs.

Another component to cost optimization is **matching your supply with your demand**. With AWS, you can leverage the elasticity of the cloud architecture to meet demands as they change. You can scale and be notified by other services to adjust your supply due to demand changes.

Next, we have **expenditure awareness**. Being fully aware and cognizant of what spending and cost drivers are happening with your business is critical. So have the ability to see, understand and break down the current costs, predict future costs, and plan accordingly only enhances the cost optimization of your architecture in the cloud.

Finally, in AWS you can **optimize over time**. With all of the tools and difference approaches, you can measure, monitor and improve your architecture from the data you collected in the AWS platform.



Cost Optimization: Design Principles

Cost Optimization

Eliminate unneeded expense.

- Adopt a consumption model
- Measure overall efficiency
- Reduce spending on data center operations
- Analyze and attribute expenditure
- Use managed services

Now let's look at the design principles that can help you achieve cost optimization:

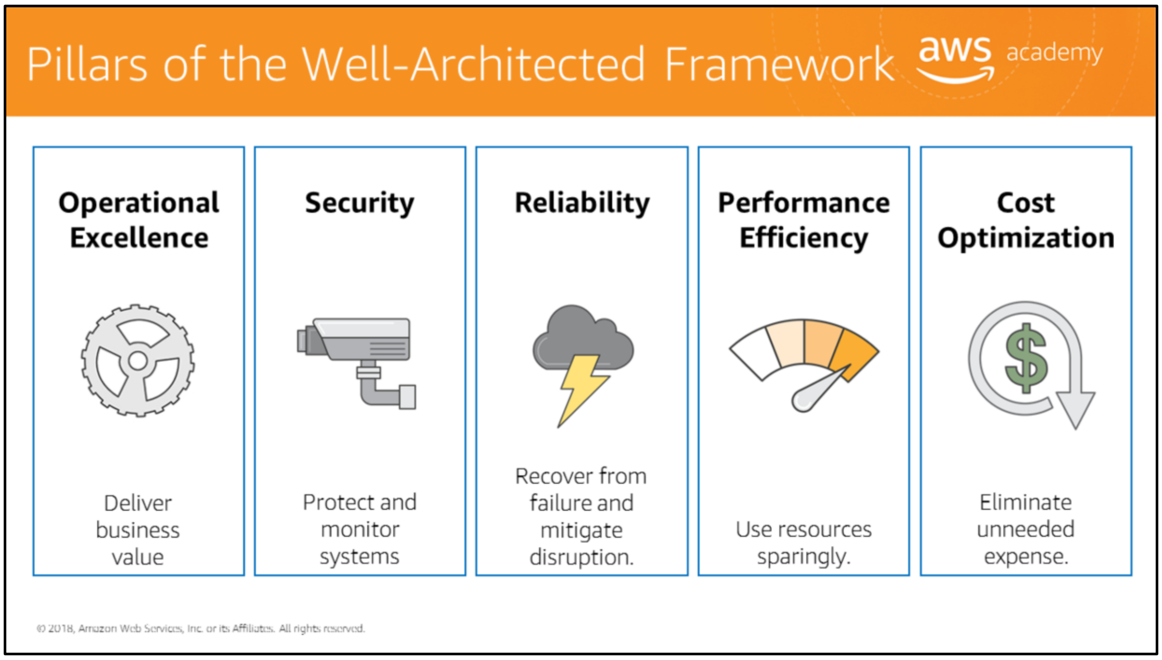
1. **Adopt a consumption model:** With the consumption model, you pay only for what computing resources you use and then increase or decrease depending on business requirements.

2. **Measure overall efficiency:** It's important to measure the business output of the systems and costs associated with delivering it. Then take this measurement to understand how gains are made from increasing output and reducing costs.

3. **Reduce spending money on data center operations:** With AWS, you no longer have to do the heavy lifting of racking, stacking, and powering servers. Instead, you can completely focus on your customers and business projects instead of the IT infrastructure.

4. **Analyze and attribute expenditure:** With the cloud, it's so much simpler and easier to accurately identify the usage and cost of systems. Customers are able to measure their return on investment, which provides them the opportunity to optimize resources and reduce costs.

5. **Use managed services:** Use managed services to reduce cost of ownership. The cloud has provided many managed services to remove the operational burden of maintaining servers for tasks such as sending email or managing databases, and since this is all done on a cloud scale, cloud service providers can offer a lower cost per transaction or service.



Pillars of the Well-Architected Framework

Operational Excellence	Security	Reliability	Performance Efficiency	Cost Optimization
Deliver business value	Protect and monitor systems	Recover from failure and mitigate disruption.	Use resources sparingly.	Eliminate unneeded expense.

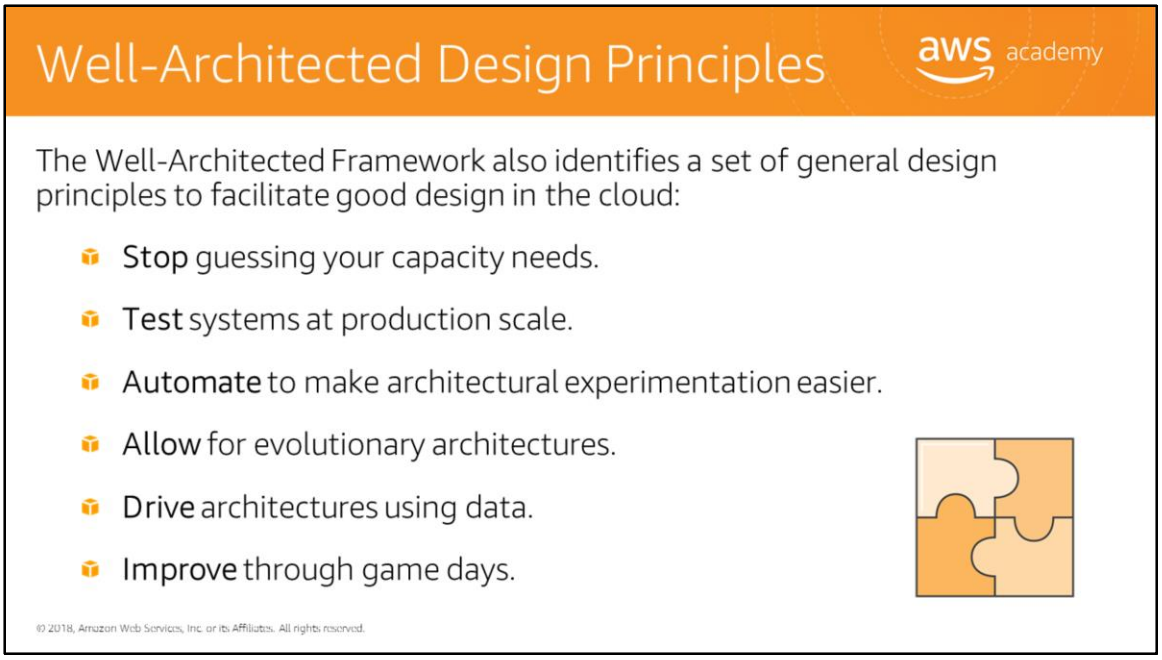
In summary, the AWS Well-Architected Framework has been developed to assist customers with assessing and improving their architectures while getting a better understanding of how their design decisions can impact their business. We reviewed each of the pillars that make up the AWS Well-Architected Framework including Operational Excellence, Security, Reliability, Performance Efficiency, and Cost Optimization.

For more detailed information and strategies on this framework, select the link.
https://aws.amazon.com/architecture/well-architected/

You can download the Well Architected whitepaper there as well as several other whitepapers to assist you with your architecture decisions.

Part 2: Well-Architected Design Principles

In Part 2, Well-Architected Design Principles, we take a deeper look into the design principles by comparing practices in a traditional environment to practices for cloud environments.



Well-Architected Design Principles

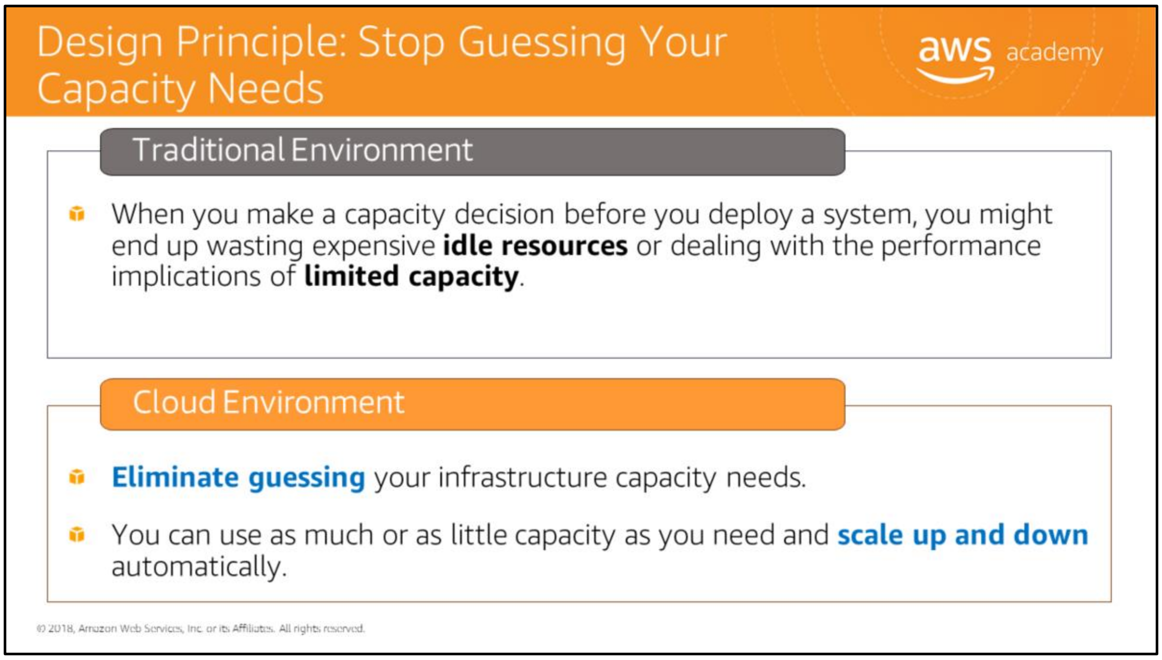
aws academy

The Well-Architected Framework also identifies a set of general design principles to facilitate good design in the cloud:

- **Stop** guessing your capacity needs.
- **Test** systems at production scale.
- **Automate** to make architectural experimentation easier.
- **Allow** for evolutionary architectures.
- **Drive** architectures using data.
- **Improve** through game days.

The Well-Architected Framework identifies a set of **general design principles** to facilitate good design in the cloud.

- **Stop** guessing your capacity needs.
- **Test** systems at production scale.
- **Automate** to make architectural experimentation easier.
- **Allow** for evolutionary architectures.
- **Drive** architectures using data.
- **Improve** through game days.



Design Principle: Stop Guessing Your Capacity Needs

Traditional Environment

- When you make a capacity decision before you deploy a system, you might end up wasting expensive **idle resources** or dealing with the performance implications of **limited capacity**.

Cloud Environment

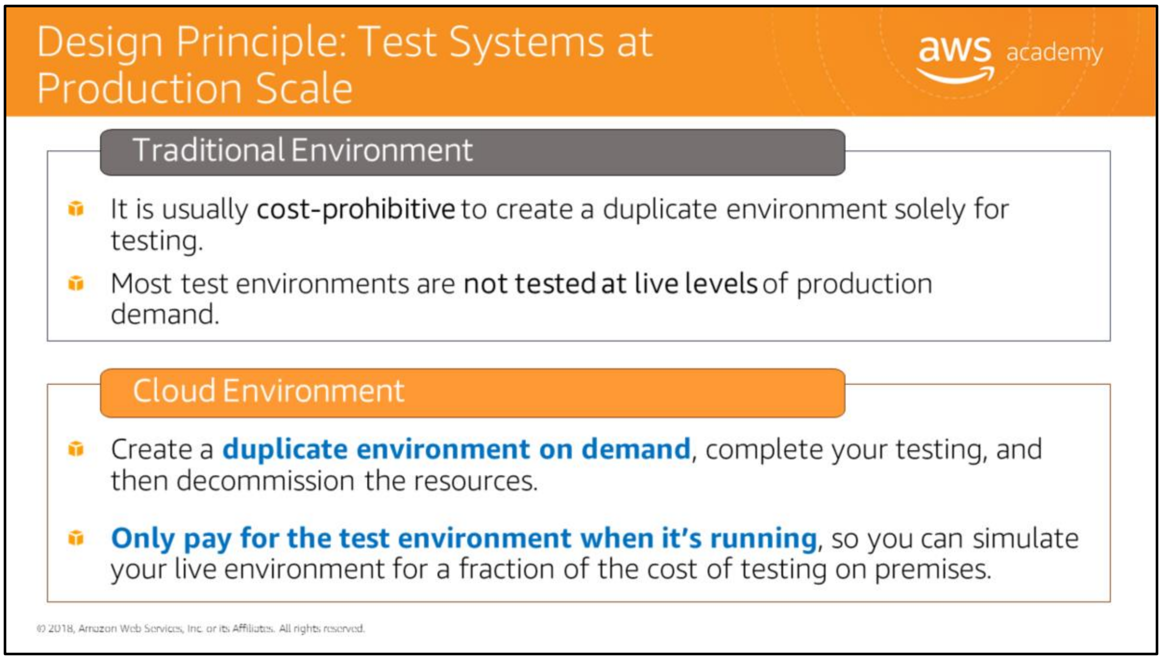
- **Eliminate guessing** your infrastructure capacity needs.
- You can use as much or as little capacity as you need and **scale up and down** automatically.

In a traditional environment, when you make a capacity decision before you deploy a system, you might end up wasting expensive idle resources or dealing with the performance implications of limited capacity.

In a traditional environment, when you make a capacity decision before you deploy a system, you might end up wasting expensive idle resources or dealing with the performance implications of limited capacity.

In a cloud environment, **eliminate guessing** your infrastructure capacity needs. You can use as much or as little capacity as you need and **scale up and down** automatically.

In a cloud environment, **eliminate guessing** your infrastructure capacity needs. You can use as much or as little capacity as you need and **scale up and down** automatically.



Design Principle: Test Systems at Production Scale

Traditional Environment

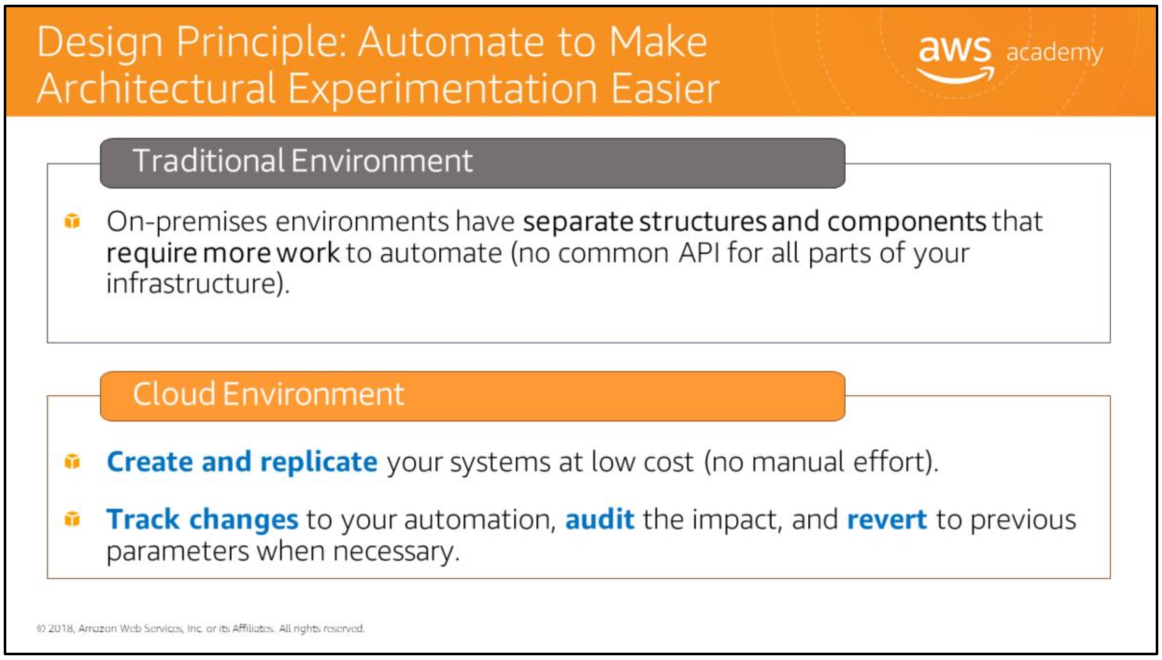
- It is usually **cost-prohibitive** to create a duplicate environment solely for testing.
- Most test environments are **not tested at live levels** of production demand.

Cloud Environment

- Create a **duplicate environment on demand**, complete your testing, and then decommission the resources.
- **Only pay for the test environment when it's running**, so you can simulate your live environment for a fraction of the cost of testing on premises.

In a traditional, non-cloud environment, it is usually cost-prohibitive to create a duplicate environment solely for testing. Consequently, most test environments are not tested at live levels of production demand.

In the cloud, you can create a duplicate a environment on demand, complete your testing, and then decommission the resources. Because you only pay for the test environment when it is running, you can simulate your live environment for a fraction of the cost of testing on premises.



Design Principle: Automate to Make Architectural Experimentation Easier

Traditional Environment

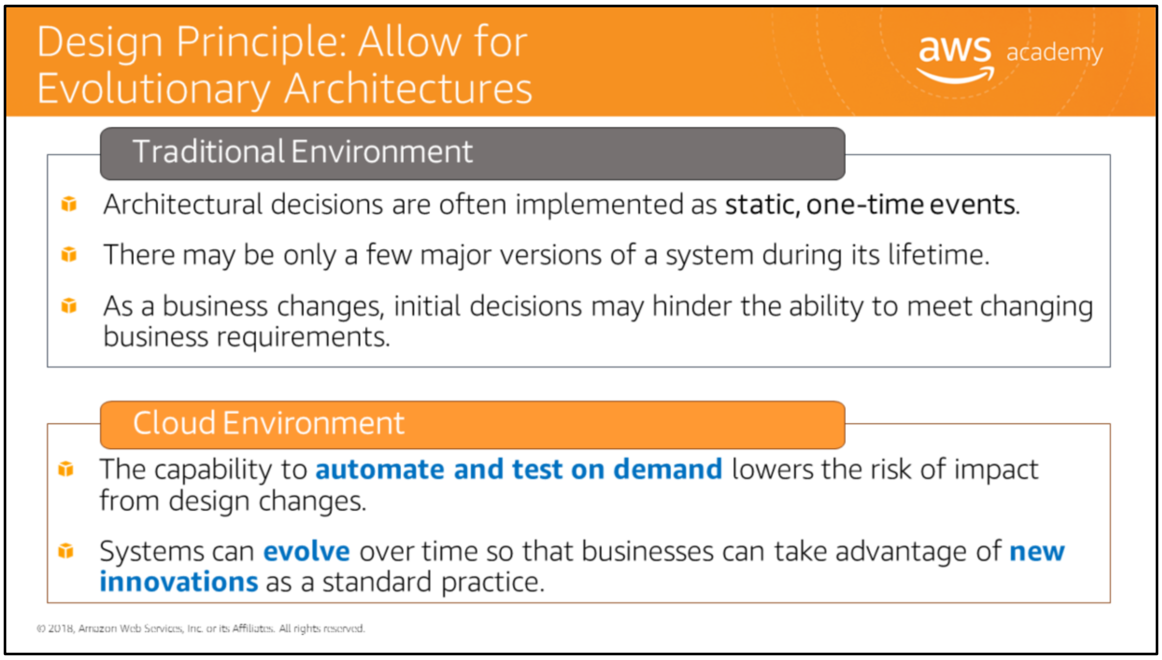
- On-premises environments have **separate structures and components** that **require more work** to automate (no common API for all parts of your infrastructure).

Cloud Environment

- **Create and replicate** your systems at low cost (no manual effort).
- **Track changes** to your automation, **audit** the impact, and **revert** to previous parameters when necessary.

On-premises environments have separate structures and components that require more work to automate with no common API for all parts of your infrastructure.

Automation allows you to create and replicate your systems at low cost with no manual effort. You can track changes to your automation, audit the impact, and revert to previous parameters when necessary.



Design Principle: Allow for Evolutionary Architectures

Traditional Environment

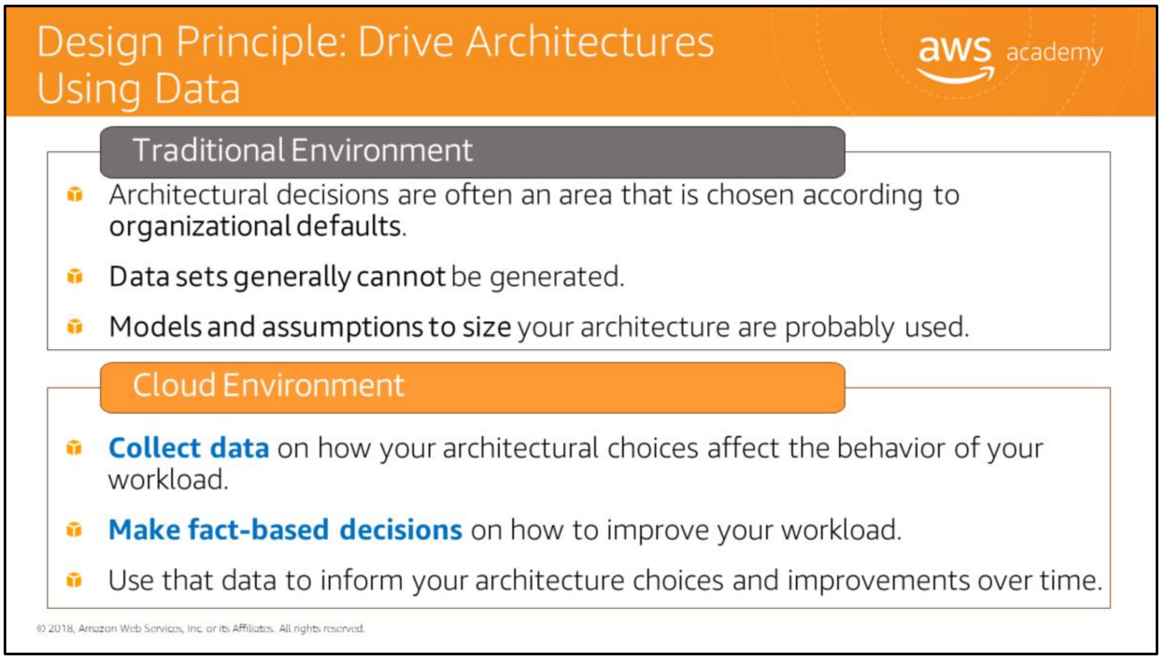
- Architectural decisions are often implemented as static, one-time events.
- There may be only a few major versions of a system during its lifetime.
- As a business changes, initial decisions may hinder the ability to meet changing business requirements.

Cloud Environment

- The capability to **automate and test on demand** lowers the risk of impact from design changes.
- Systems can **evolve** over time so that businesses can take advantage of **new innovations** as a standard practice.

In a traditional environment, architectural decisions are often implemented as a static, one-time events, with a few major versions of a system during its lifetime. As a business and its context continue to change, these initial decisions may hinder the system's ability to meet changing business requirements.

In the cloud, the capability to automate and test on demand lowers the risk of impact from design changes. This allows systems to evolve over time so that businesses can take advantage of new innovations as a standard practice.



Design Principle: Drive Architectures Using Data

Traditional Environment

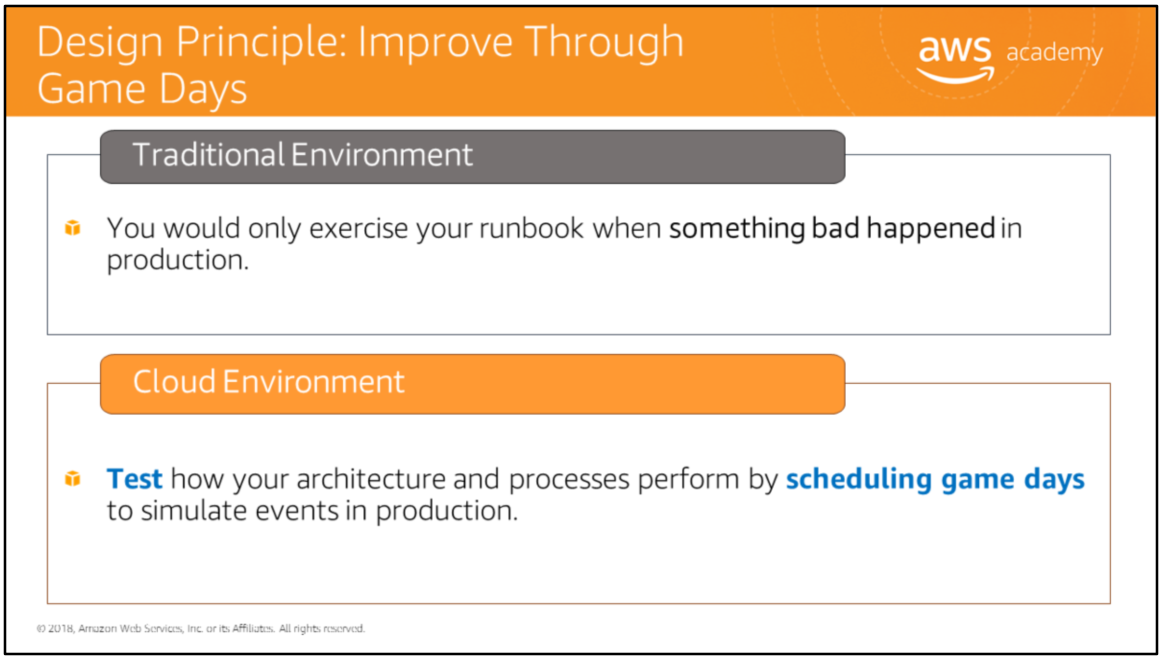
- Architectural decisions are often an area that is chosen according to **organizational defaults**.
- **Data sets generally cannot** be generated.
- **Models and assumptions to size** your architecture are probably used.

Cloud Environment

- **Collect data** on how your architectural choices affect the behavior of your workload.
- **Make fact-based decisions** on how to improve your workload.
- Use that data to inform your architecture choices and improvements over time.

In a traditional, non-cloud environment, architectural choices are often made according to organizational defaults rather than through a data-driven approach. You generally **could not generate data sets** that would allow you to make informed decisions, so you probably **used models and assumptions to size your architecture.**

In the cloud, you can collect data on how your architectural choices affect the behavior of your workload. This lets you make fact-based decisions on how to improve your workload. Your cloud infrastructure is code, so you can use that data to inform your architecture choices and improvements over time.



Design Principle: Improve Through Game Days

aws academy

Traditional Environment

- You would only exercise your runbook when **something bad happened** in production.

Cloud Environment

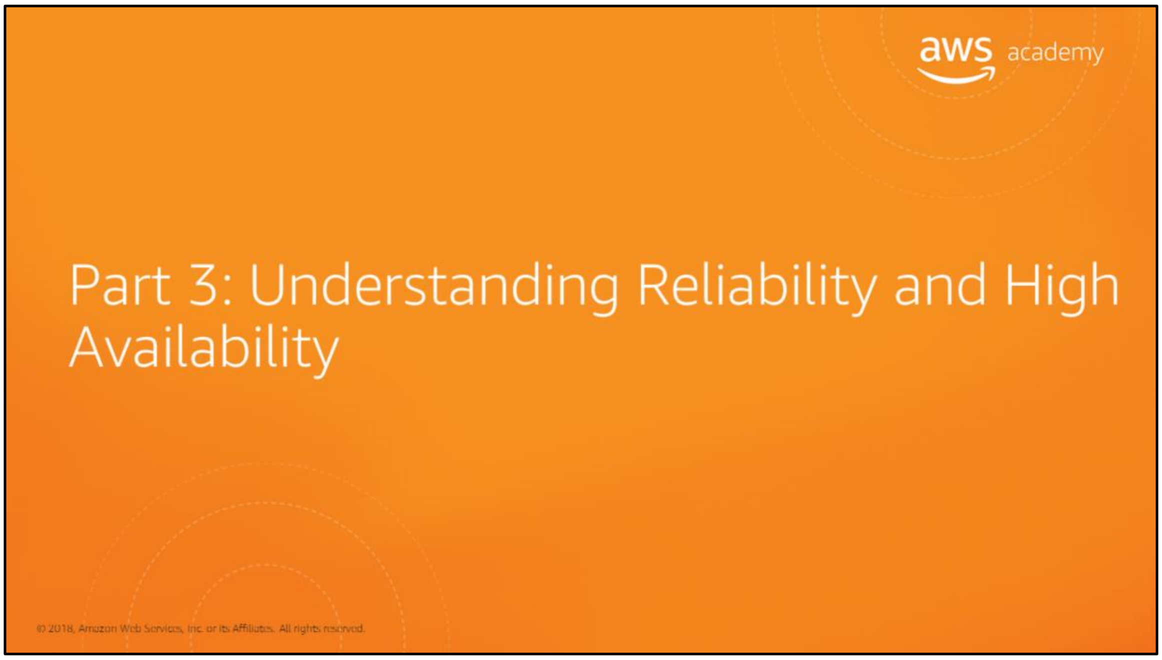
- **Test** how your architecture and processes perform by **scheduling game days** to simulate events in production.

In a traditional environment, you would **only exercise** your **runbook** when something **bad happened** in production.

In the cloud, test how your architecture and processes perform by regularly scheduling game days to simulate events in production. This will help you understand where improvements can be made and can help develop organizational experience in dealing with events.

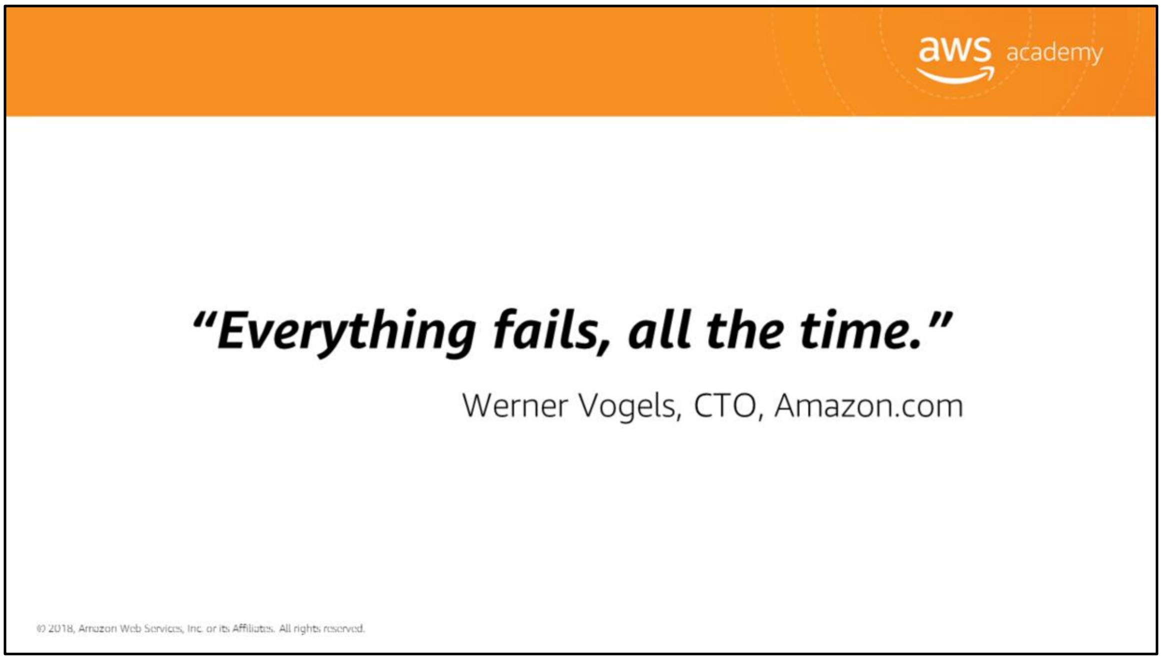
An example of this is Chaos Monkey, a software tool that was developed in 2011 by Netflix engineers to test the resiliency and recoverability of their AWS environment. The software simulates failures of instances of services running within AWS Auto Scaling Groups (ASG) by shutting down one or more of the virtual machines. Chaos Monkey works on the principle that the best way to avoid major failures is to fail constantly. Chaos Monkey is now part of a larger suite of tools called the Simian Army designed to simulate and test responses to various system failures and edge cases.

To learn more about Chaos Monkey select the link.
https://whatis.techtarget.com/definition/Chaos-Monkey.



Part 3: Understanding Reliability and High Availability

In Part 3, Understanding Reliability and High Availability, we take a look at what it means to design for these important solution attributes.

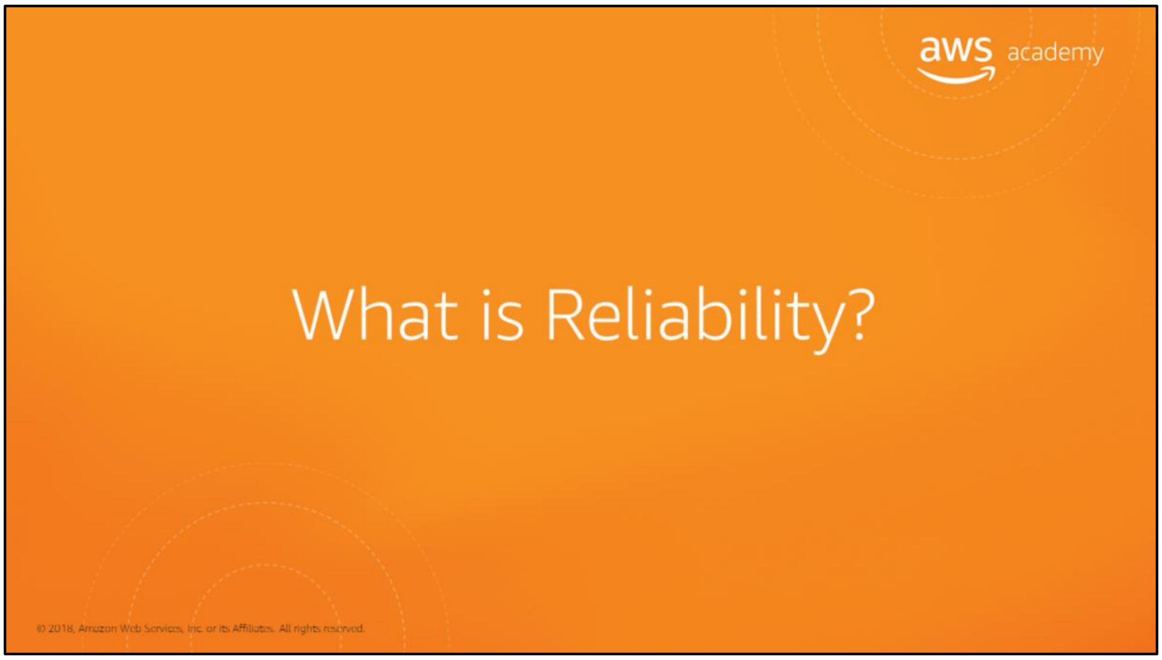


“Everything fails, all the time.”

Werner Vogels, CTO, Amazon.com

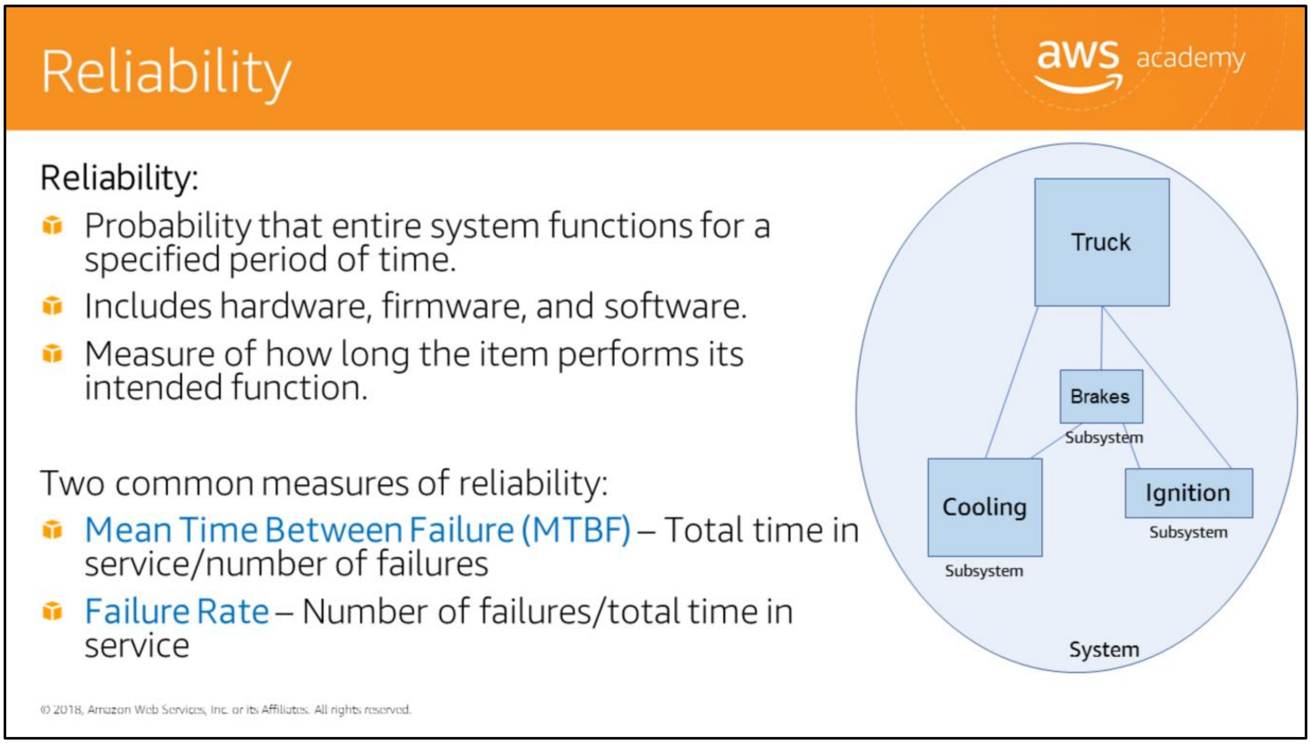
It has been said that, “Everything fails, all the time.”

Failures are costly to businesses.



What is Reliability?

So, what exactly is reliability?



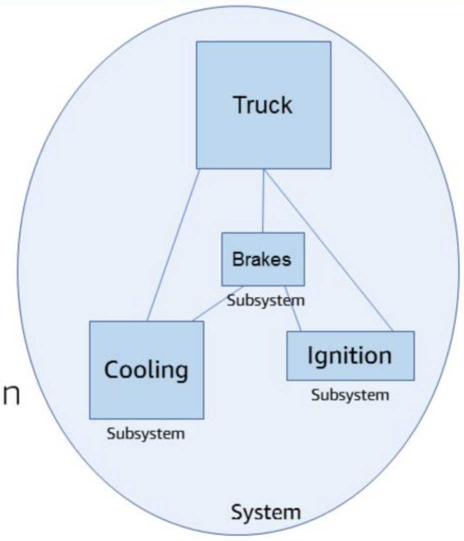
Reliability

Reliability:

- Probability that entire system functions for a specified period of time.
- Includes hardware, firmware, and software.
- Measure of how long the item performs its intended function.

Two common measures of reliability:

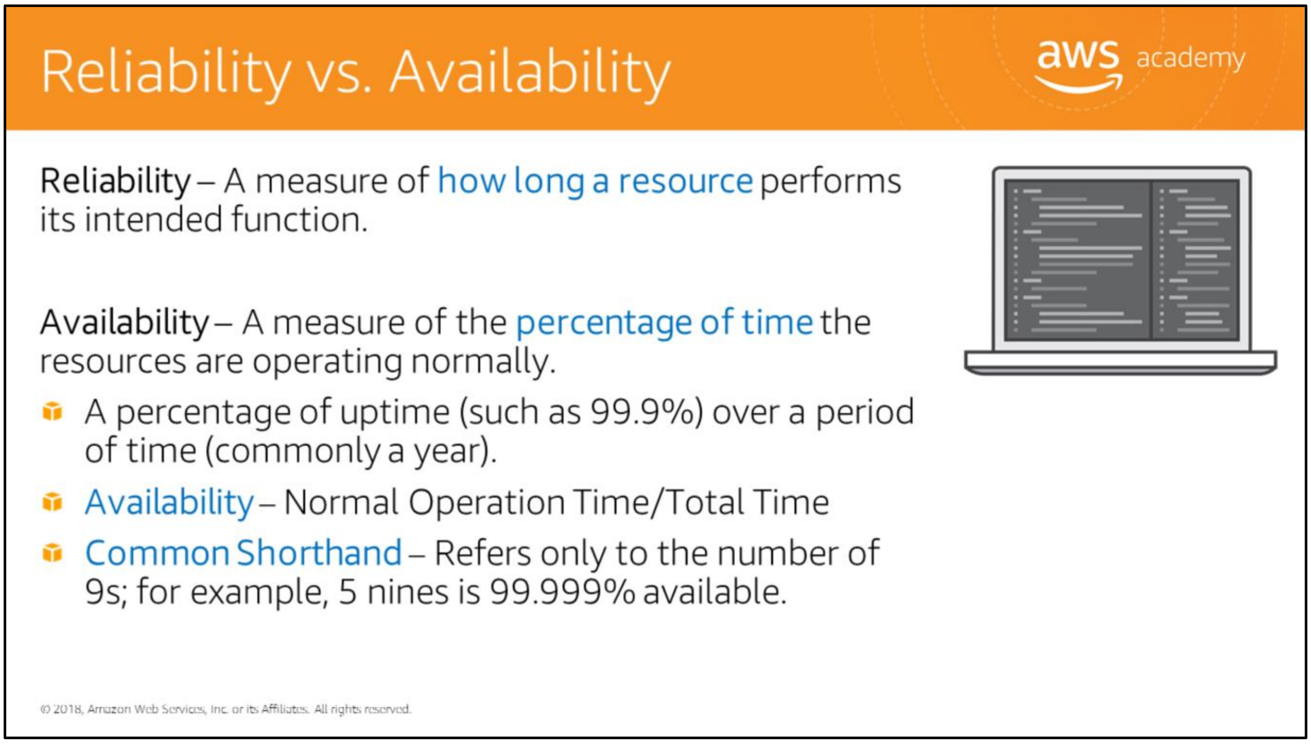
- Mean Time Between Failure (MTBF) – Total time in service/number of failures
- Failure Rate – Number of failures/total time in service



Reliability is the probability that an entire system, including all hardware, firmware, and software, will satisfactorily function for a specified period of time. Simply put, reliability is a measure of how long the item performs its intended function.

Two common measures of reliability include Mean Time Between Failure (or MTBF), which is the total time in service or number of failures, and failure rate, which is the number of failures or total time in service.

It is helpful to look at an example using a common object to understand reliability. When you purchase a vehicle, you purchase a system or collection of subsystems that must work together for the vehicle to be considered reliable. The reliability of each of the subsystems, including the cooling, the ignition, and the brakes are a part of determining the reliability of the vehicle. So, if you go out to the parking lot and try to start the vehicle and the ignition fails, reliability is negatively impacted. Remember, while we can also measure subsystem reliability, reliability is based on the entire system functioning as designed.



Reliability vs. Availability

Reliability – A measure of how long a resource performs its intended function.

Availability – A measure of the percentage of time the resources are operating normally.

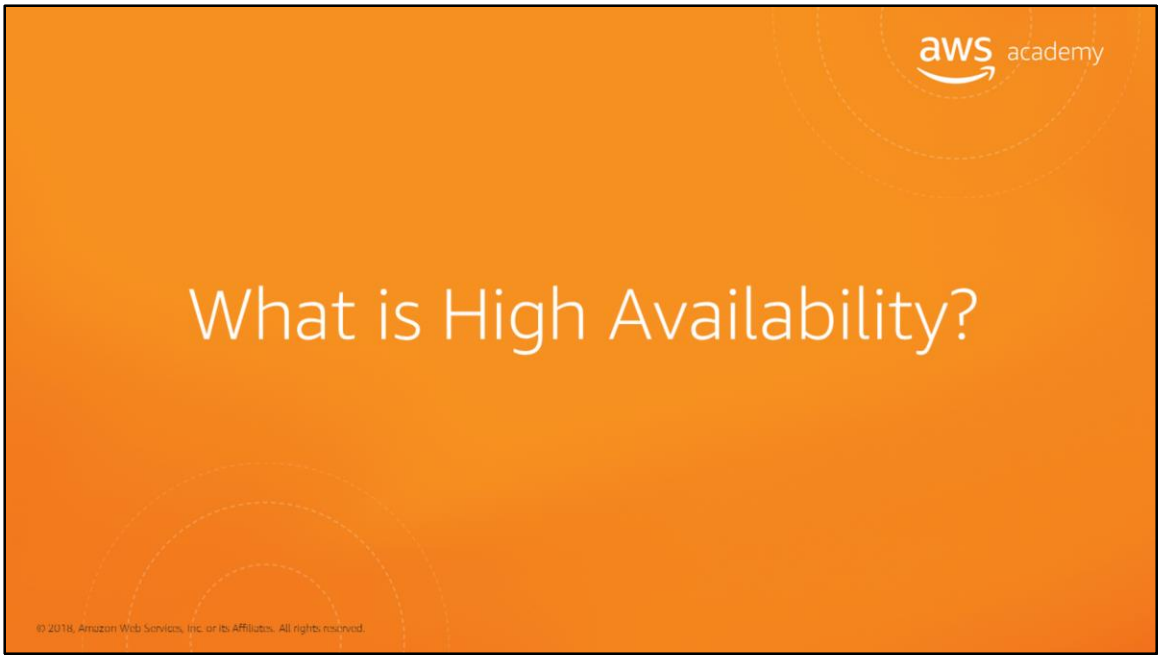
- A percentage of uptime (such as 99.9%) over a period of time (commonly a year).
- Availability – Normal Operation Time/Total Time
- Common Shorthand – Refers only to the number of 9s; for example, 5 nines is 99.999% available.

Reliability is a measure of how long a resource performs its intended function. What's the difference between that and availability?

As a matter of a fact, **reliability** is closely related to **availability**. **Reliability** is a measure of how long a resource performs its intended function while **availability** is a measure of the percentage of time the resources are in an operable state.

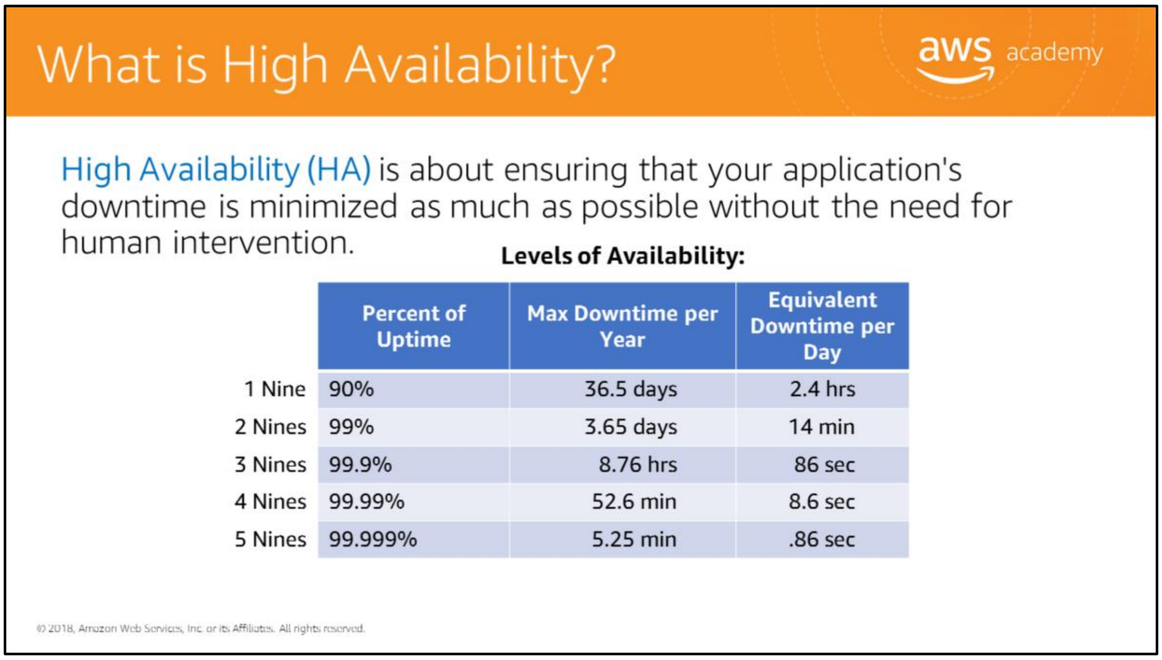
As we looked at the services, we often saw numbers like 99.99%. Those numbers refer to the availability, or the percentage of time that a system or application is correctly performing the operations expected. A common shorthand refers only to the number of9s. For example, 5 nines is 99.999% available.

For more information about reliability, select the link. https://d0.awsstatic.com/whitepapers/architecture/AWS-Reliability-Pillar.pdf.



What is High Availability?

What is high availability?



What is High Availability?

High Availability (HA) is about ensuring that your application's downtime is minimized as much as possible without the need for human intervention.

Levels of Availability:

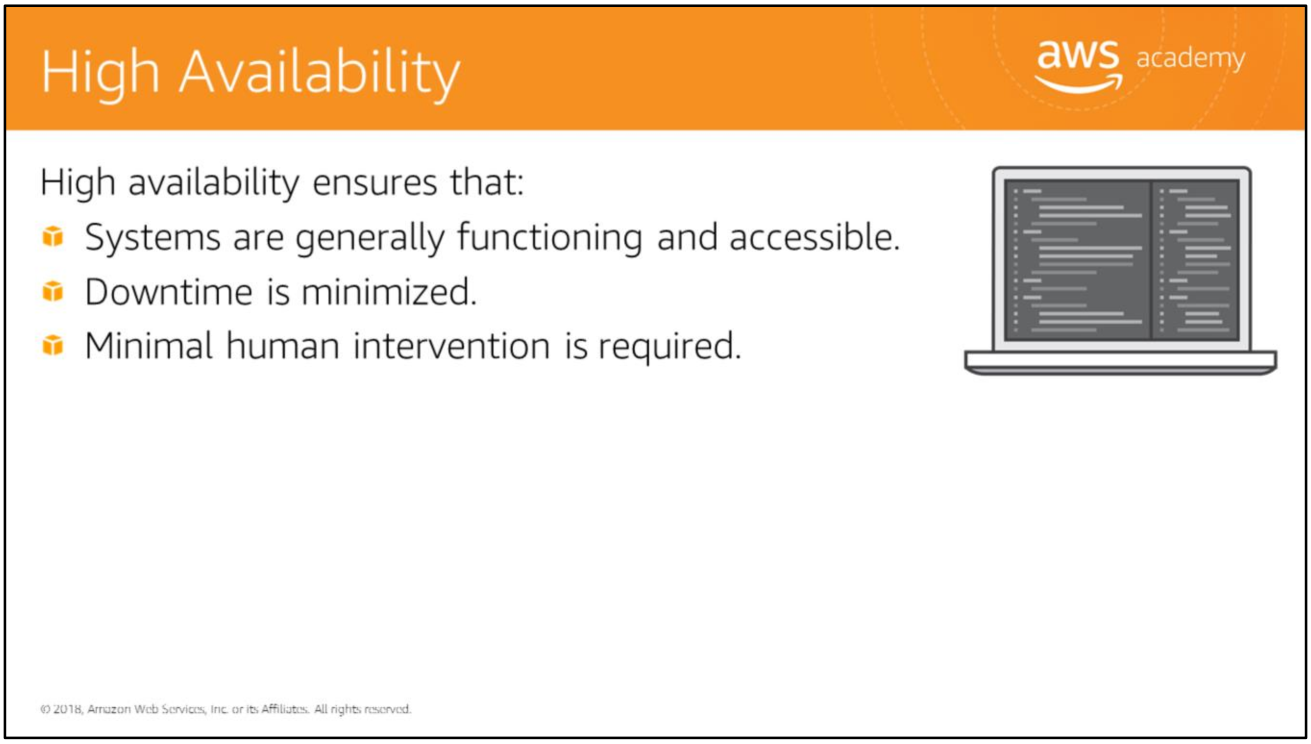
	Percent of Uptime	Max Downtime per Year	Equivalent Downtime per Day
1 Nine	90%	36.5 days	2.4 hrs
2 Nines	99%	3.65 days	14 min
3 Nines	99.9%	8.76 hrs	86 sec
4 Nines	99.99%	52.6 min	8.6 sec
5 Nines	99.999%	5.25 min	.86 sec

Availability specifically refers to the amount of time your system is in a functioning condition. In general terms, your availability is referred to as 100% minus your system's downtime.

High Availability (or HA) is about ensuring that your application's downtime is minimized as much as possible without the need for human intervention. It views availability not as a series of replicated physical components, but rather as a set of system-wide, shared resources that cooperate to guarantee essential services. High availability combines software with industry-standard hardware to minimize downtime by quickly restoring essential services when a system, component, or application fails. While not instantaneous, services are restored rapidly, often in less than a minute.

Since events which may disrupt your system's availability are never entirely predictable, there are always ways to make an application more available, but keep in mind that improving availability usually leads to increased cost. When considering how to make your environment more available, it's important to balance the cost of the improvement with the benefit to your users.

Does HA mean that you ensure your application is always alive or reachable, or does it mean that the app is servicing requests within an acceptable level of performance?



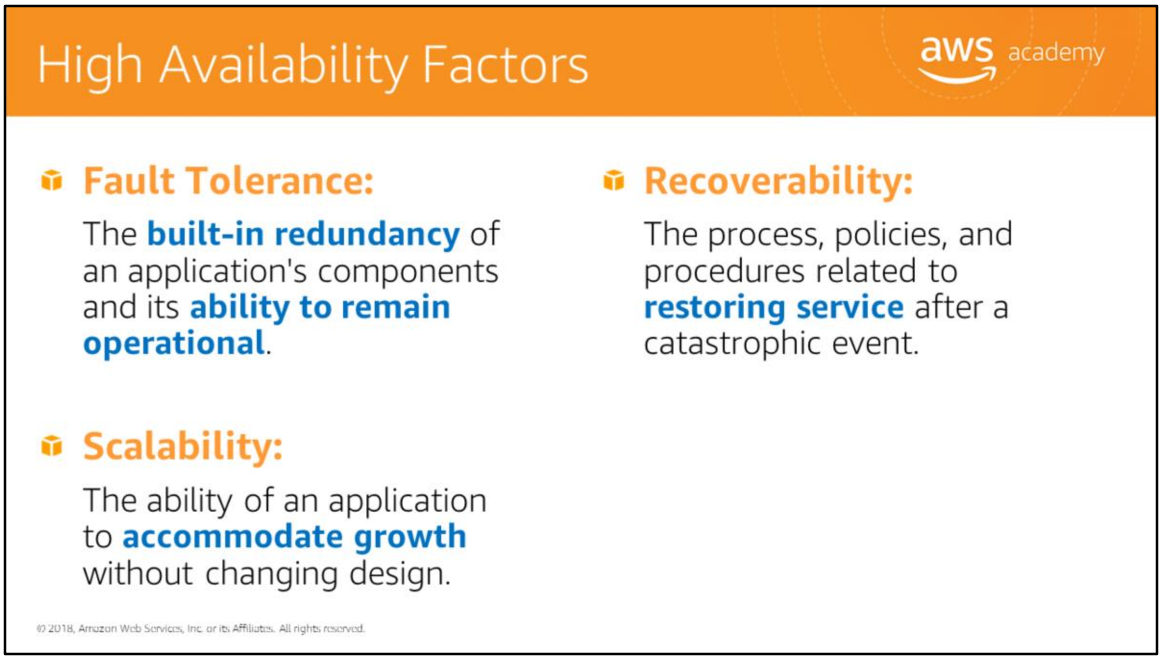
High Availability

High availability ensures that:

- Systems are generally functioning and accessible.
- Downtime is minimized.
- Minimal human intervention is required.

High availability is a concept regarding the entire system. Its goal is to ensure that your systems are always functioning and accessible and that downtime is minimized as much as possible, without the need for human intervention.

The AWS platform is available for users to build fault-tolerant, highly available systems and architectures. Users can build these systems with minimal human interaction and up-front financial investment and it is all customizable to your needs.



High Availability Factors

- **Fault Tolerance:** The **built-in redundancy** of an application's components and its **ability to remain operational**.
- **Recoverability:** The process, policies, and procedures related to **restoring service** after a catastrophic event.
- **Scalability:** The ability of an application to **accommodate growth** without changing design.

Fault tolerance, recoverability, and scalability are the prime factors that determine the overall availability of your application.

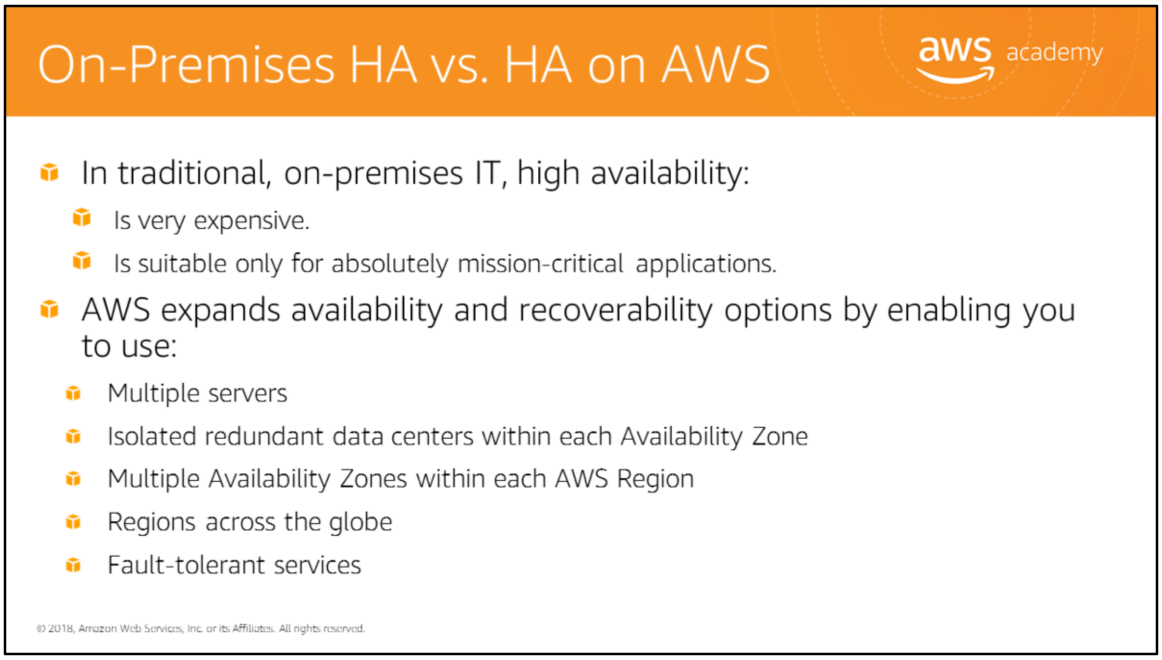
Fault Tolerance is often confused with high availability, but fault tolerance refers to the built-in redundancy of an application's components and its **ability to remain operational** even if some of the components of that system fail. Fault tolerance relies on specialized hardware to detect a hardware fault and instantaneously switch to a redundant hardware component, whether the failed component is a processor, memory board, power supply, I/O subsystem, or storage subsystem. The fault tolerant model does not address software failures, which is by far the most common reason for downtime.

Scalability is a question of how quickly your application's infrastructure can respond to increased capacity needs to ensure that your application is available and performs within your required standards. It does not guarantee availability, but is one part of your application's availability.

Recoverability is often overlooked as a component of availability. In the event a natural disaster makes one or more of your components unavailable, or destroys your primary data source, can you restore service quickly and without lost data? We will not discuss specific disaster recovery strategies in this module.

It is these non-functional requirements that typically define the design of your infrastructure.

While a highly available and fault-tolerant environment may span multiple Availability Zones and AWS Regions, there are costs associated with this design that must be balanced with the availability requirements.



On-Premises HA vs. HA on AWS

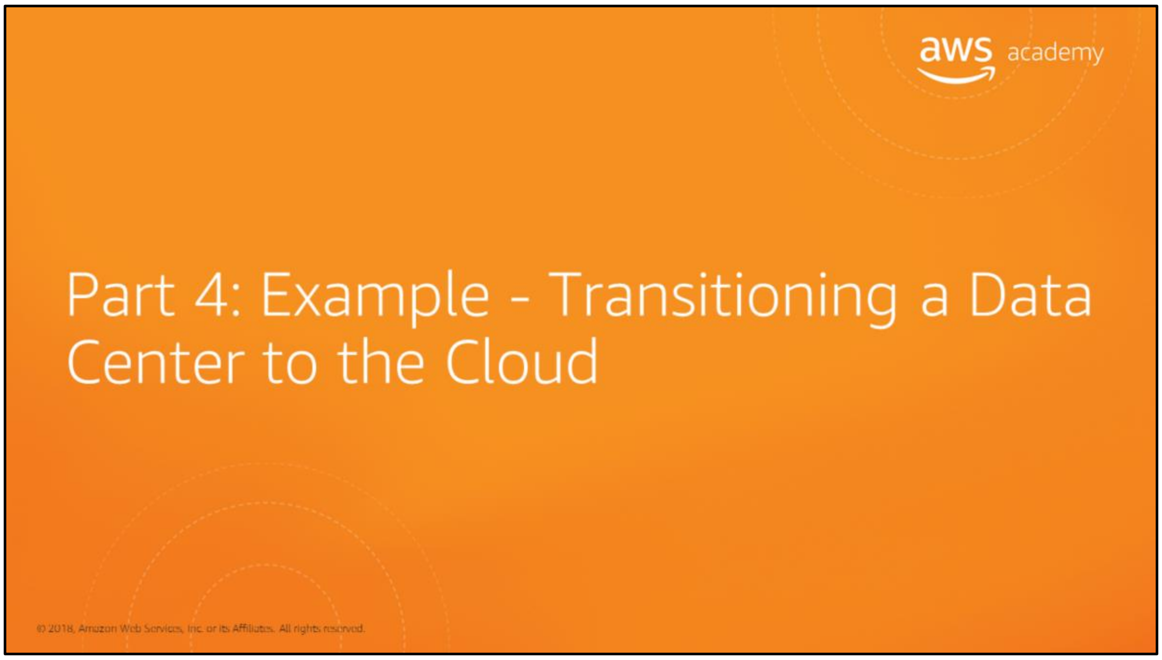
- In traditional, on-premises IT, high availability:
  - Is very expensive.
  - Is suitable only for absolutely mission-critical applications.
- AWS expands availability and recoverability options by enabling you to use:
  - Multiple servers
  - Isolated redundant data centers within each Availability Zone
  - Multiple Availability Zones within each AWS Region
  - Regions across the globe
  - Fault-tolerant services

To understand the differences between availability solutions on-premises and in the cloud, let's compare them.

Traditionally, ensuring high availability at your local datacenters can be expensive. Usually it's only used on absolutely mission-critical applications.

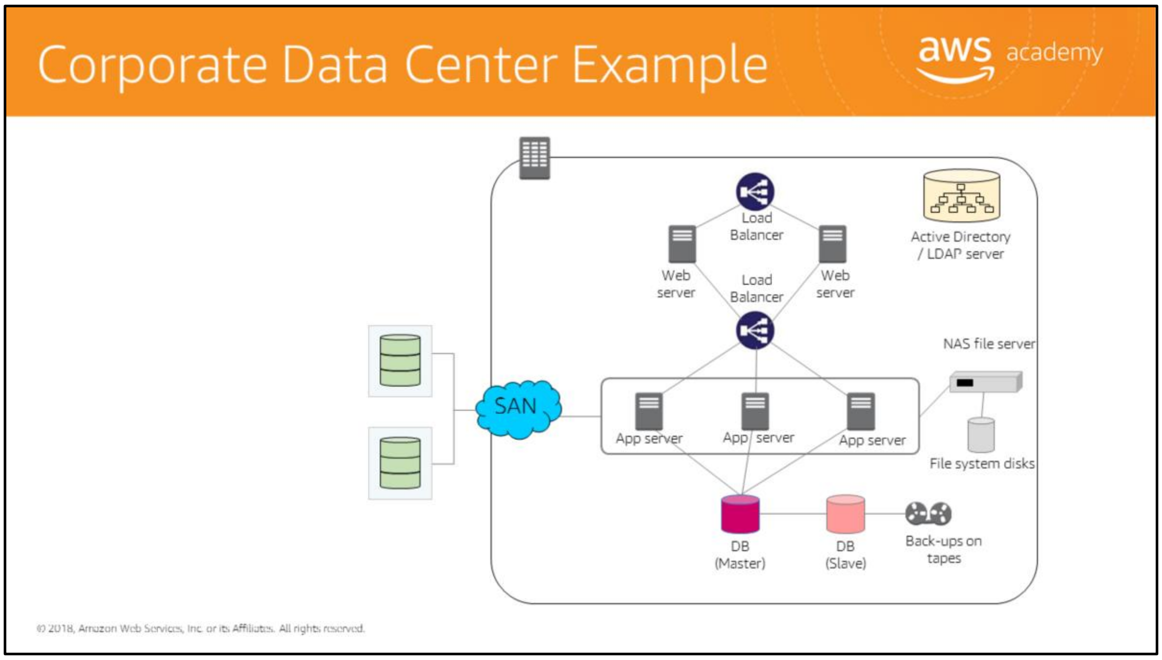
However, on AWS, you have the options to expand availability and recoverability among whatever servers you choose.

You can ensure high availability on:
- Multiple servers
- Isolated, redundant data centers within each Availability Zone
- Multiple Availability Zones within each Region
- Multiple Regions across the globe
- Fault tolerant services to use as you please



Part 4: Example - Transitioning a Data Center to the Cloud

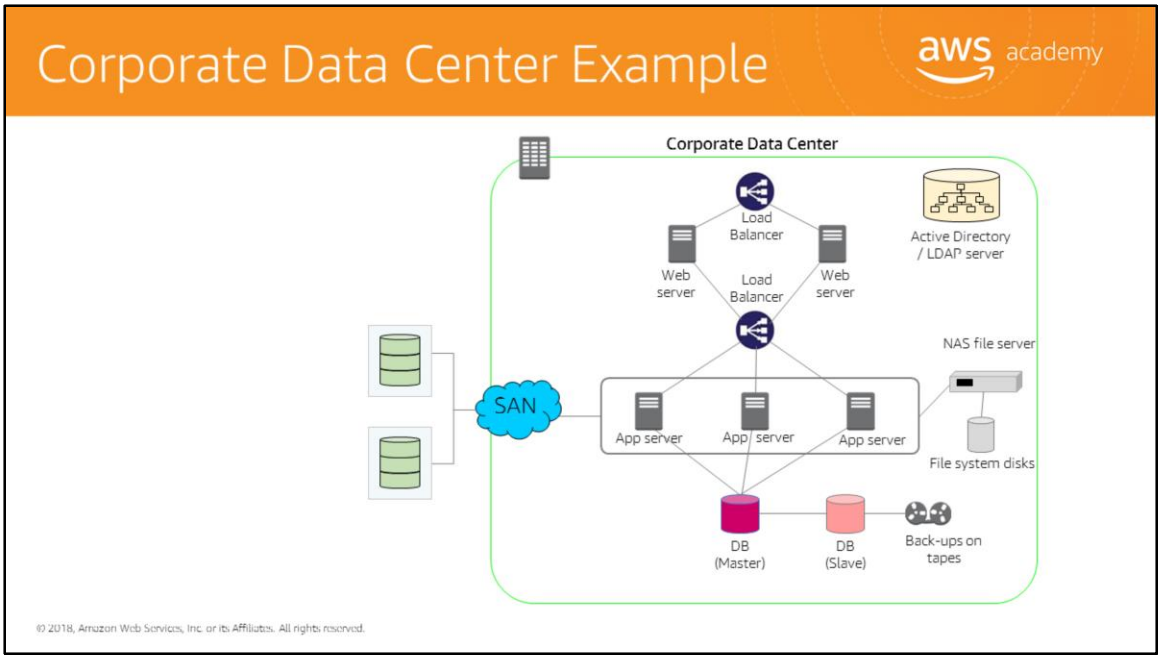
Now, let's review an example of transitioning a data center to the cloud.



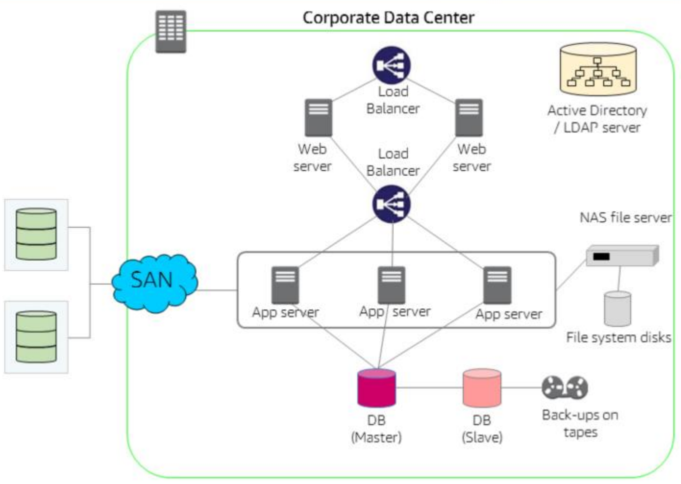
Corporate Data Center Example

A traditional on-premises or corporate data center–based infrastructure might include a setup like this.

Let's walk through one example of how an arrangement like this could be set up and run on AWS instead.

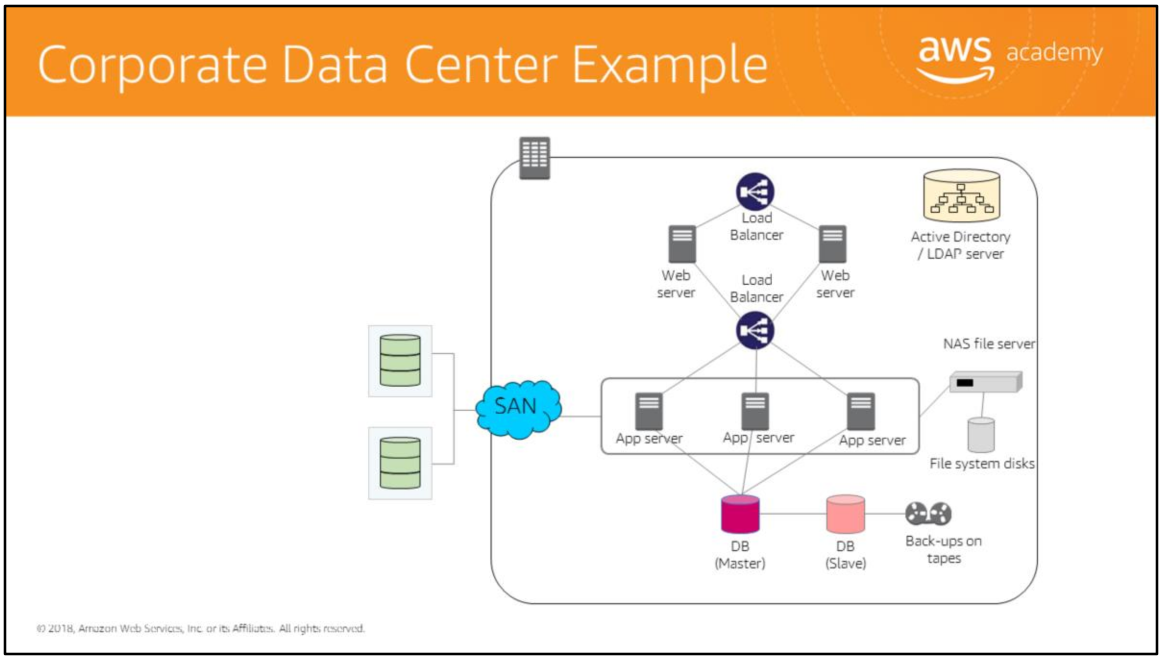


Corporate Data Center Example



The bright green box indicates what is contained within the data center.

The blue cloud labeled **SAN** with the attached external disks refers to storage that is outside the corporate data center. A **Storage Area Network (SAN)** is a specialized, high-speed network that provides block-level network access to storage. SANs are often used to improve application availability (e.g., multiple data paths) and enhance application performance (e.g., off-load storage functions, segregate networks, etc.).



Corporate Data Center Example

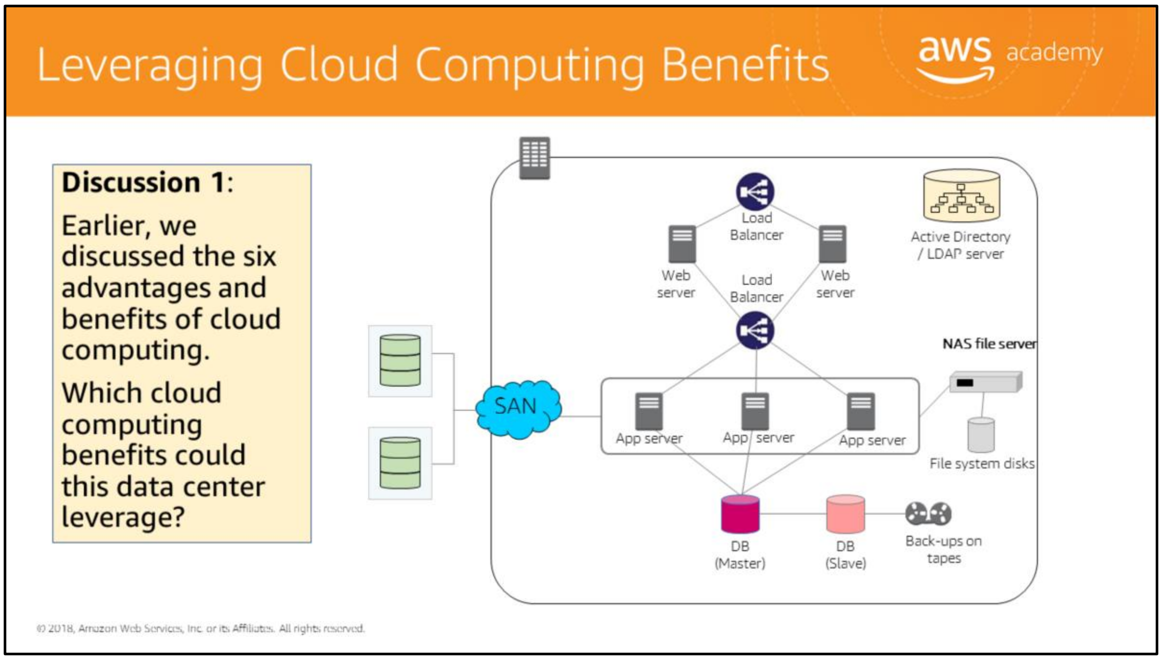
This diagram represents a three-tier client-server architecture in the corporate data center.

At the bottom of this diagram are the database servers with attached tape back devices. This tier is responsible for the database logic.

The middle of the diagram contains the application servers. An application server is a component-based product that resides in the middle-tier of a server centric architecture. It provides middleware services for security and state maintenance, along with data access and persistence. The application servers also contain the business logic. The middle section also contains **Network Attached Storage (NAS)**. NAS devices are file servers that provide a centralized location for users on a network to store, access, edit, and share files.

The web servers are located at the top of the diagram. The web servers are responsible for the presentation logic. They are accompanied by **load balancers**. Load balancers are responsible for efficiently distributing incoming network traffic across a group of backend servers.

The **Active Directory (AD) / LDAP (Lightweight Directory Access Protocol)** server is like a phone book that enables anyone to locate organizations, individuals, and other resources, such as files and devices in a network, whether on the public Internet or on a corporate intranet.



Leveraging Cloud Computing Benefits

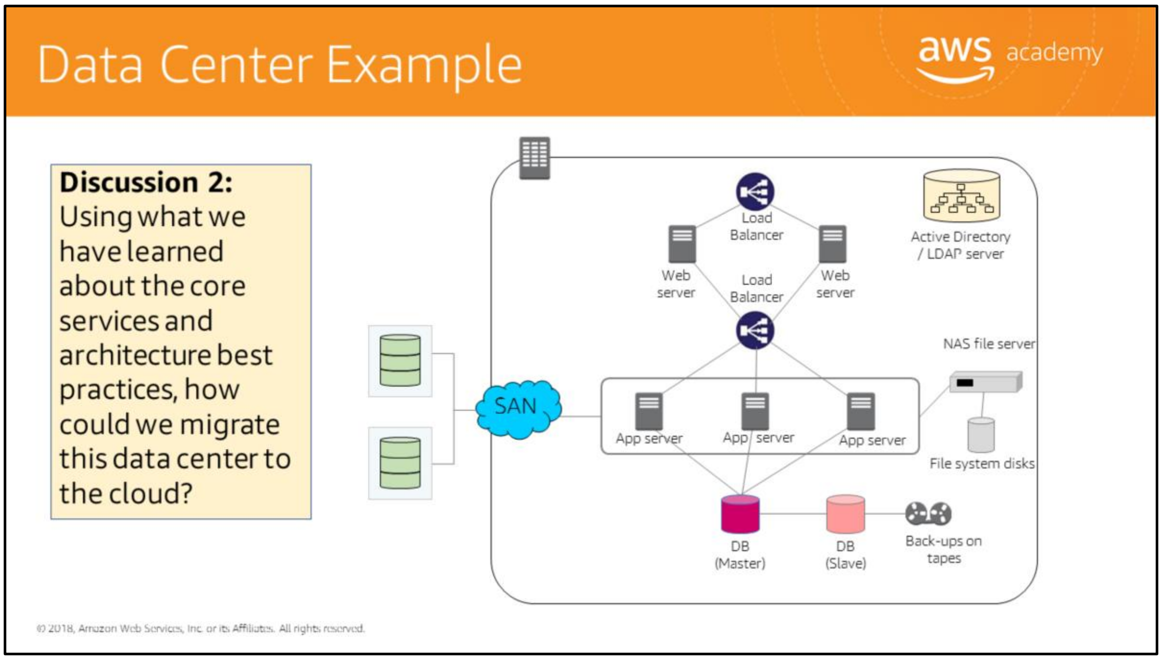
Discussion 1:

Earlier, we discussed the six advantages and benefits of cloud computing.

Which cloud computing benefits could this data center leverage?

Earlier, we discussed the six advantages and benefits of cloud computing. Which cloud computing benefits could this data center leverage?

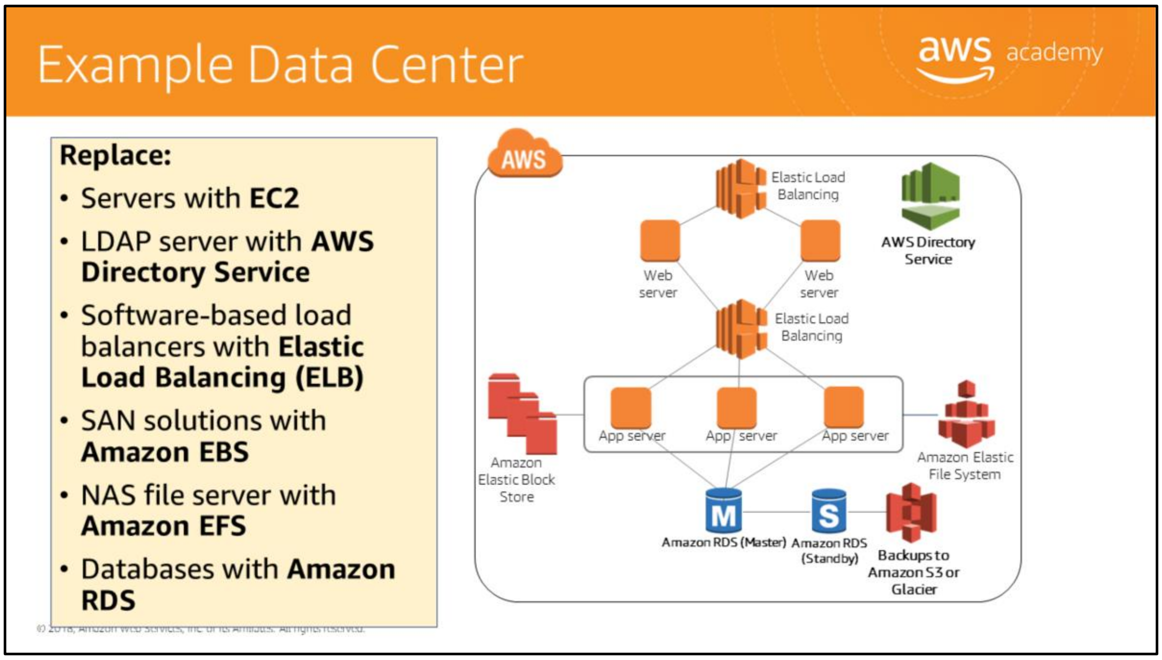
1. Trade **capital expense** for **variable expense** → Stop buying hardware.
2. Benefit from **massive economies of scale** → Benefit from Amazon's purchasing power.
3. **Eliminate guessing** on your capacity needs --> Construct a flexible, highly available solution using scaling.
4. Increase **speed** and **agility** → Deploy and decommission with just a few clicks.
5. **Stop spending money** to run and maintain data centers → Purchase only the services needed.
6. **Go global** in minutes.



Data Center Example

Discussion 2: Using what we have learned about the core services and architecture best practices, how could we migrate this data center to the cloud?

Using what we have learned about the core services and architecture best practices, how could we migrate this data center to the cloud?



Example Data Center

Replace:
- Servers with **EC2**
- LDAP server with **AWS Directory Service**
- Software-based load balancers with **Elastic Load Balancing (ELB)**
- SAN solutions with **Amazon EBS**
- NAS file server with **Amazon EFS**
- Databases with **Amazon RDS**

Servers, such as these web servers and app servers, are replaced with **Amazon EC2** instances that run all of the same software. Because Amazon EC2 instances can run a variety of Windows Server, Red Hat, SUSE, Ubuntu, or our own Amazon Linux operating systems, virtually all server applications can be run on Amazon EC2 instances.

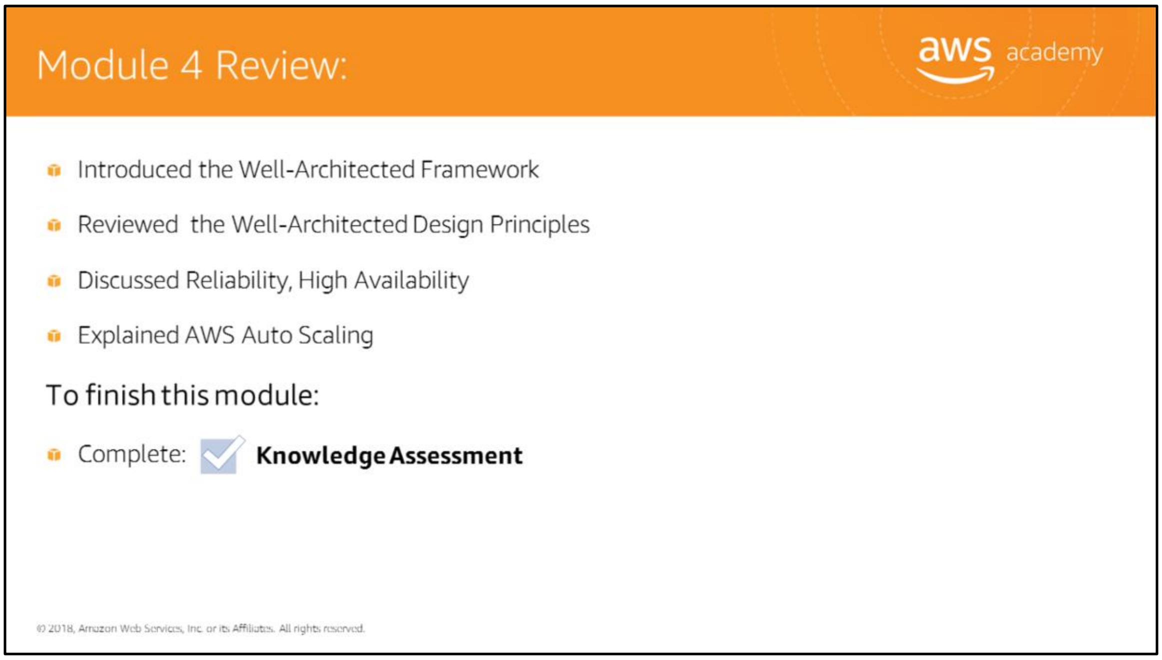
The LDAP server is replaced with **AWS Directory Service**, which supports LDAP authentication and allows you to easily set up and run Microsoft Active Directory in the cloud or connect your AWS resources with existing on-premises Microsoft Active Directory.

Software-based load balancers are replaced with **Elastic Load Balancing (ELB)** load balancers. ELB is a fully managed load balancing solution that scales automatically, as needed, and can perform health checks on attached resources, thus redistributing load away from unhealthy resources as necessary.

SAN solutions can be replaced with **Amazon Elastic Block Store (Amazon EBS)** volumes. These volumes can be attached to the application servers to store data long-term and share the data between instances.

Amazon Elastic File System (Amazon EFS) could be used to replace your NAS file server. Amazon EFS is a file storage service for Amazon EC2 instances with a simple interface that allows you to create and configure file systems. It also grows and shrinks your storage automatically as you add and remove files, so you are always using exactly the amount of storage you need. Another solution could be to run an NAS solution on an Amazon EC2 instance. Many NAS solutions are available via the AWS Marketplace. Select the link to learn more https://aws.amazon.com/marketplace/.

Databases can be replaced with **Amazon Relational Database Service (RDS)**, which lets you run Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle, and Microsoft SQL Server on a managed AWS-based platform. Finally, Amazon RDS instances can be automatically backed up to **Amazon S3**; thus, replacing the need for on-premises database backup hardware.



Module 4 Review:

- Introduced the Well-Architected Framework
- Reviewed the Well-Architected Design Principles
- Discussed Reliability, High Availability
- Explained AWS Auto Scaling

To finish this module:

- Complete: ✓ **Knowledge Assessment**

Building a highly available and fault tolerant architecture does not have to be difficult. With AWS, you can use the services and tools offered to ensure your applications and systems have high availability and increased fault tolerance.

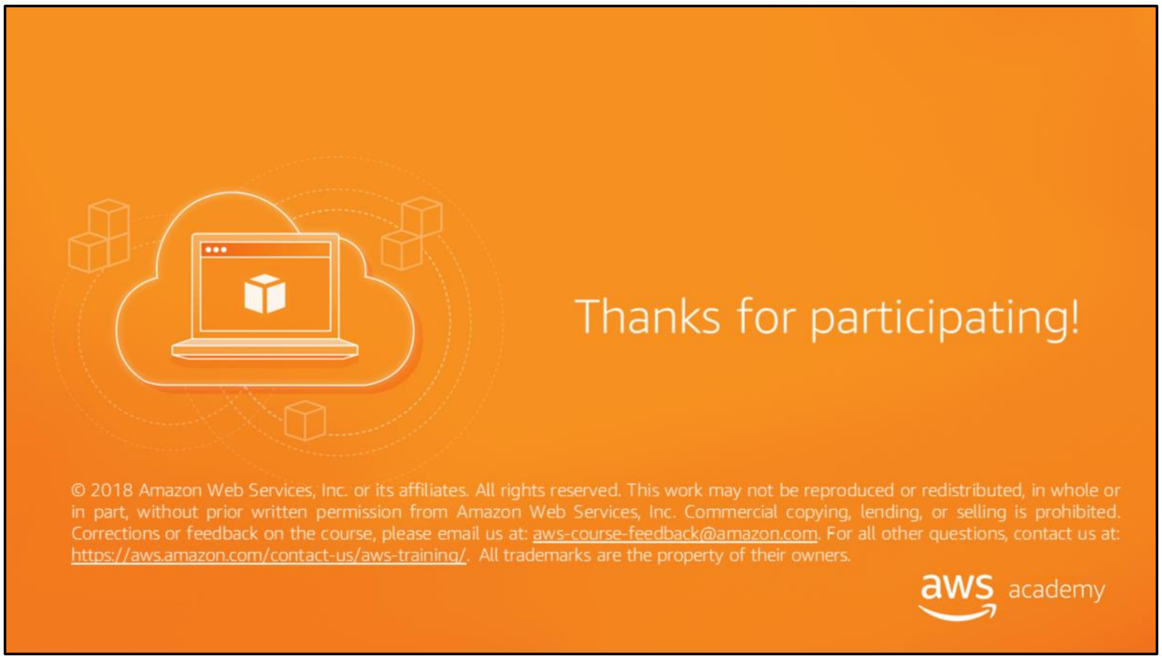
In review, we:
- Introduced and reviewed the Well-Architected Framework.
- Reviewed the Well-Architected Design Principles.
- Discussed the meaning of Reliability, High Availability and Scaling and their importance to a cloud solution.

To finish this module, please complete the corresponding knowledge assessment.

Up Next: Module 5 – Cloud Support

Overview of AWS Technical Support Plans

Like the design considerations for architecting a cloud solution, technical support for your solution is also important. In Module 5, we look at technical support plans.



Thanks for participating!

Thanks for participating!