SWINBURNE UNIVERSITY OF TECHNOLOGY

Cloud Computing Architecture

Lecture 11
Cloud Architecture Evaluation

Last Week

- Routing across Regions – Route53
- High availability and reliability
  - More high availability patterns
  - Disaster recovery patterns
- Sample Architectures

SWINBURNE UNIVERSITY OF TECHNOLOGY

This week

- AWS Well-architected Framework
  - ☐ Operational Excellence
  - ☐ Security
  - ☐ Reliability
  - ☐ Performance
  - ☐ Cost
- AWS Trusted Advisor

Quizzes:
ACF Mod 4 Introduction to Cloud Architecting
ACA Mod 3.08 – Mod 3.12 Assessments

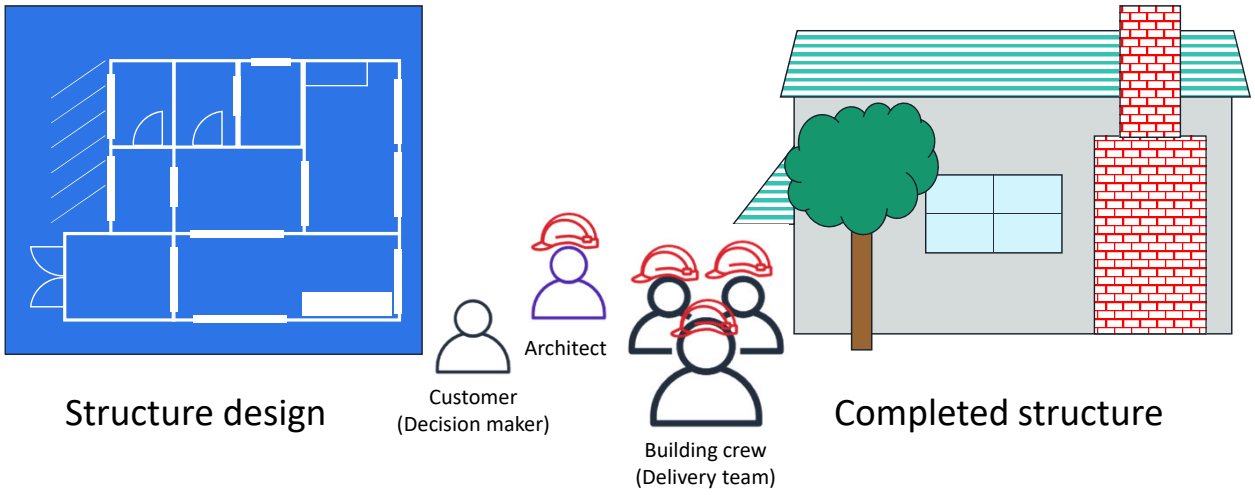
SWINBURNE UNIVERSITY OF TECHNOLOGY

This week

- **AWS Well-architected Framework**
  - ☐ Operational Excellence
  - ☐ Security
  - ☐ Reliability
  - ☐ Performance
  - ☐ Cost
- AWS Trusted Advisor

SWINBURNE UNIVERSITY OF TECHNOLOGY

Architecture: designing and building



Architecture is the art and science of designing and building large structures. Large systems require architects to manage their size and complexity.

Cloud architects:

- Engage with decision makers to identify the business goal and the capabilities that need improvement.
- Ensure alignment between technology deliverables of a solution and the business goals.
- Work with delivery teams that are implementing the solution to ensure that the technology features are appropriate.

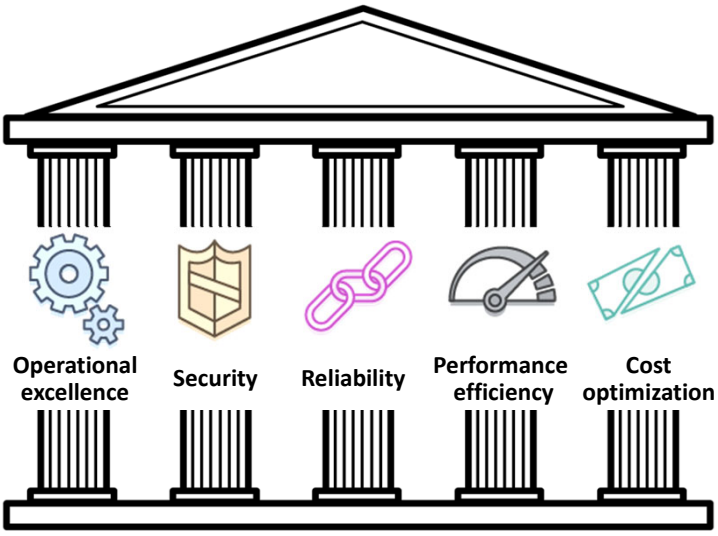
Having well-architected systems greatly increases the likelihood of business success.

What is the AWS Well-Architected Framework?

- A guide for designing infrastructures that are:
  - ✓Secure
  - ✓High-performing
  - ✓Resilient
  - ✓Efficient
- A consistent approach to evaluating and implementing cloud architectures
- A way to provide best practices that were developed through lessons learned by reviewing customer architectures

The AWS Well-Architected Framework is a guide that is designed to help you build the most secure, high-performing, resilient, and efficient infrastructure possible for your cloud applications and workloads. It provides a set of foundational questions and best practices that can help you evaluate and implement your cloud architectures. AWS developed the Well-Architected Framework after reviewing thousands of customer architectures on AWS.

Pillars of the AWS Well-Architected Framework



The AWS Well-Architected Framework is organized into five pillars: operational excellence, security, reliability, performance efficiency, and cost optimization.

Pillar organization

Best practice area	**Identity and Access Management**
Question text	**SEC 1: How do you manage credentials and authentication?**
Question context	Credential and authentication mechanisms include passwords, tokens, and keys that grant access directly or indirectly in your workload. Protect credentials with appropriate mechanisms to help reduce the risk of accidental or malicious use.
Best practices	Best practices: • Define requirements for identity and access management • Secure AWS account root user • Enforce use of multi-factor authentication • Automate enforcement of access controls • Integrate with centralized federation provider • Enforce password requirements • Rotate credentials regularly • Audit credentials periodically

Each pillar includes a set of design principles and best practice areas. A set of foundational questions is under each best practice area. Some context and a list of best practices are provided for each question.

This week

- AWS Well-architected Framework
  - **Operational Excellence**
  - Security
  - Reliability
  - Performance
  - Cost
- AWS Trusted Advisor

SWINBURNE UNIVERSITY OF TECHNOLOGY

Operational Excellence pillar

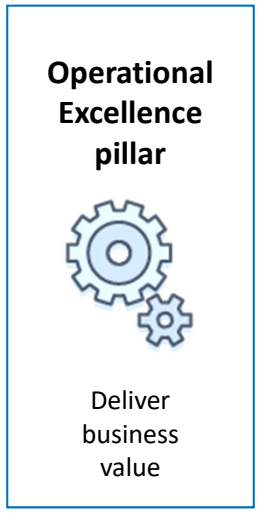
Operational Excellence pillar

Deliver business value

- Focus
  - Run and monitor systems to deliver business value, and to continually improve supporting processes and procedures.
- Key topics
  - Managing and automating changes
  - Responding to events
  - Defining standards to successfully manage daily operations

The *Operational Excellence pillar* focuses on the ability to run and monitor systems to deliver business value, and to continually improve supporting processes and procedures. Key topics include: managing and automating changes, responding to events, and defining standards to successfully manage daily operations.

Operational excellence design principles



- Perform operations as code
- Annotate documentation
- Make frequent, small, reversible changes
- Refine operations procedures frequently
- Anticipate failure
- Learn from all operational events and failures

There are six design principles for operational excellence in the cloud:

- *Perform operations as code* – Define your entire workload (that is, applications and infrastructure) as code and update it with code. Implement operations procedures as code and configure them to automatically trigger in response to events. By performing operations as code, you limit human error and enable consistent responses to events.
- *Annotate documentation* – Automate the creation of annotated documentation after every build. Annotated documentation can be used by people and systems. Annotations can be used as input to your operations code.
- *Make frequent, small, reversible changes* – Design workloads to enable components to be updated regularly. Make changes in small increments that can be reversed if they fail (without affecting customers when possible).
- *Refine operations procedures frequently* – Look for opportunities to improve operations procedures. Evolve your procedures appropriately as your workloads evolve. Set up regular game days to review all procedures, validate their effectiveness, and ensure that teams are familiar with them.
- *Anticipate failure* – Identify potential sources of failure so that they can be removed or mitigated. Test failure scenarios and validate your understanding of their impact. Test your response procedures to ensure that they are effective and that teams are familiar with their execution. Set up regular game days to test workloads and team responses to simulated events.
- *Learn from all operational failures* – Drive improvement through lessons learned from all operational events and failures. Share what is learned across teams and through the entire organization.

Operational excellence questions

aws academy

Prepare

- How do you determine what your priorities are?
- How do you design your workload so that you can understand its state?
- How do you reduce defects, ease remediation, and improve flow into production?
- How do you mitigate deployment risks?
- How do you know that you are ready to support a workload?

Operate

- How do you understand the health of your workload?
- How do you understand the health of your operations?
- How do you manage workload and operations events?

Evolve

- How do you evolve operations?

The foundational questions for operational excellence fall under three best practice areas: prepare, operate, and evolve.

Operations teams must understand business and customer needs so they can effectively and efficiently support business outcomes. Operations teams create and use procedures to respond to operational events and validate the effectiveness of procedures to support business needs. Operations teams collect metrics that are used to measure the achievement of desired business outcomes. As business context, business priorities, and customer needs, change over time, it's important to design operations that evolve in response to change and to incorporate lessons learned through their performance.

For prescriptive guidance on implementation, see the Operational Excellence Pillar whitepaper.

This week

- AWS Well-architected Framework
  - Operational Excellence
  - **Security**
  - Reliability
  - Performance
  - Cost
- AWS Trusted Advisor

SWINBURNE UNIVERSITY OF TECHNOLOGY

Security pillar

Security pillar

Protect and monitor systems

- Focus
  - Protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies.
- Key topics
  - Identifying and managing who can do what
  - Establishing controls to detect security events
  - Protecting systems and services
  - Protecting confidentiality and integrity of data

The *Security pillar* focuses on the ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies. Key topics include: protecting confidentiality and integrity of data, identifying and managing who can do what (or privilege management), protecting systems, and establishing controls to detect security events.

Security design principles

Security pillar

Protect and monitor systems

- Implement a strong identity foundation
- Enable traceability
- Apply security at all layers
- Automate security best practices
- Protect data in transit and at rest
- Keep people away from data
- Prepare for security events

There are seven design principles that can improve security:

- *Implement a strong identity foundation* – Implement the principle of least privilege and enforce separation of duties with appropriate authorization for each interaction with your AWS resources. Centralize privilege management and reduce or even eliminate reliance on long-term credentials.
- *Enable traceability* – Monitor, alert, and audit actions and changes to your environment in real time. Integrate logs and metrics with systems to automatically respond and take action.
- *Apply security at all layers* – Apply defense in depth and apply security controls to all layers of your architecture (for example, edge network, virtual private cloud, subnet, and load balancer; and every instance, operating system, and application).
- *Automate security best practices* – Automate security mechanisms to improve your ability to securely scale more rapidly and cost effectively. Create secure architectures and implement controls that are defined and managed as code in version-controlled templates.
- *Protect data in transit and at rest* – Classify your data into sensitivity levels and use mechanisms such as encryption, tokenization, and access control where appropriate.
- *Keep people away from data* – To reduce the risk of loss or modification of sensitive data due to human error, create mechanisms and tools to reduce or eliminate the need for direct access or manual processing of data.
- *Prepare for security events* – Have an incident management process that aligns with organizational requirements. Run incident response simulations and use tools with automation to increase your speed of detection, investigation, and recovery.

Security questions

Identity and access management

- How do you manage credentials and authentication?
- How do you control human access?
- How do you control programmatic access?

Detective controls

- How do you detect and investigate security events?
- How do you defend against emerging security threats?

Infrastructure protection

- How do you protect your networks?
- How do you protect your compute resources?

Data protection

- How do you classify your data?
- How do you protect your data at rest?
- How do you protect your data in transit?

Incident response

- How do you respond to an incident?

The foundational questions for security fall under five best practice areas: identity and access management, detective controls, infrastructure protection, data protection, and incident response.

Before you architect any system, you must put security practices in place. You must be able to control who can do what. In addition, you must be able to identify security incidents, protect your systems and services, and maintain the confidentiality and integrity of data through data protection. You should have a well-defined and practiced process for responding to security incidents. These tools and techniques are important because they support objectives such as preventing financial loss or complying with regulatory obligations.

For prescriptive guidance on implementation, see the Security Pillar whitepaper.

This week

- AWS Well-architected Framework
  - Operational Excellence
  - Security
  - **Reliability**
  - Performance
  - Cost
- AWS Trusted Advisor

SWINBURNE UNIVERSITY OF TECHNOLOGY

Reliability pillar

Reliability pillar

Recover from failure and mitigate disruption.

- Focus
  - Prevent and quickly recover from failures to meet business and customer demand.
- Key topics
  - Setting up
  - Cross-project requirements
  - Recovery planning
  - Handling change

The *Reliability pillar* focuses on the ability of a system to recover from infrastructure or service disruptions, dynamically acquire computing resources to meet demand, and mitigate disruptions such as misconfigurations or transient network issues. Key topics include: set up, cross-project requirements, recovery planning, and handling change.

Reliability design principles

Reliability pillar

Recover from failure and mitigate disruption.

- Test recovery procedures
- Automatically recover from failure
- Scale horizontally to increase aggregate system availability
- Stop guessing capacity
- Manage change in automation

There are five design principles that can increase reliability:

- *Test recovery procedures* – Test how your systems fail and validate your recovery procedures. Use automation to simulate different failures or to recreate scenarios that led to failures before. This practice can expose failure pathways that you can test and rectify before a real failure scenario.
- *Automatically recover from failure* – Monitor systems for key performance indicators and configure your systems to trigger an automated recovery when a threshold is breached. This practice enables automatic notification and failure-tracking, and for automated recovery processes that work around or repair the failure.
- *Scale horizontally to increase aggregate system availability* – Replace one large resource with multiple, smaller resources and distribute requests across these smaller resources to reduce the impact of a single point of failure on the overall system.
- *Stop guessing capacity* – Monitor demand and system usage, and automate the addition or removal of resources to maintain the optimal level for satisfying demand.
- *Manage change in automation* – Use automation to make changes to infrastructure and manage changes in automation.

Reliability questions

Foundations

- How do you manage service limits?
- How do you manage your network topology?

Change management

- How does your system adapt to changes in demand?
- How do you monitor your resources?
- How do you implement change?

Failure management

- How do you back up data?
- How does your system withstand component failure?
- How do you test resilience?
- How do you plan for disaster recovery?

The foundational questions for reliability fall under three best practice areas: foundations, change management, and failure management.

To achieve reliability, a system must have both a well-planned foundation and monitoring in place. It must have mechanisms for handling changes in demand or requirements. The system should be designed to detect failure and automatically heal itself.

For prescriptive guidance on implementation, see the Reliability Pillar whitepaper.

This week

- AWS Well-architected Framework
  - Operational Excellence
  - Security
  - Reliability
  - **Performance**
  - Cost Optimisation
- AWS Trusted Advisor

SWINBURNE UNIVERSITY OF TECHNOLOGY

Performance Efficiency pillar

Performance Efficiency pillar

Use resources sparingly.

- Focus
  - Use IT and computing resources efficiently to meet system requirements and to maintain that efficiency as demand changes and technologies evolve.
- Key topics
  - Selecting the right resource types and sizes based on workload requirements
  - Monitoring performance
  - Making informed decisions to maintain efficiency as business needs evolve

The *Performance Efficiency pillar* focuses on the ability to use IT and computing resources efficiently to meet system requirements, and to maintain that efficiency as demand changes or technologies evolve. Key topics include: selecting the right resource types and sizes based on workload requirements, monitoring performance, and making informed decisions to maintain efficiency as business needs evolve.

Performance efficiency design principles

Performance Efficiency pillar

Use resources sparingly.

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

There are five design principles that can improve performance efficiency:

- *Democratize advanced technologies* – Consume technologies as a service. For example, technologies such as NoSQL databases, media transcoding, and machine learning require expertise that is not evenly dispersed across the technical community. In the cloud, these technologies become services that teams can consume. Consuming technologies enables teams to focus on product development instead of resource provisioning and management.
- *Go global in minutes* – Deploy systems in multiple AWS Regions to provide lower latency and a better customer experience at minimal cost.
- *Use serverless architectures* – Serverless architectures remove the operational burden of running and maintaining servers to carry out traditional compute activities. Serverless architectures can also lower transactional costs because managed services operate at cloud scale.
- *Experiment more often* – Perform comparative testing of different types of instances, storage, or configurations.
- *Have mechanical sympathy* – Use the technology approach that aligns best to what you are trying to achieve. For example, consider your data access patterns when you select approaches for databases or storage.

Performance efficiency questions

Selection

- How do you select the best performing architecture?
- How do you select your compute solution?
- How do you select your storage solution?
- How do you select your database solution?
- How do you select your networking solution?

Review

- How do you evolve your workload to take advantage of new releases?

Monitoring

- How do you monitor your resources to ensure they are performing as expected?

Tradeoffs

- How do you use tradeoffs to improve performance?

The foundational questions for performance efficiency fall under four best practice areas: selection, review, monitoring, and tradeoffs.

Use data to design and build a high-performance architecture. Gather data on all aspects of the architecture, from the high-level design to the selection and configuration of resource types. Review your choices periodically to ensure that you are taking advantage of new AWS services. Perform monitoring so that you are aware of any deviance from expected performance and can take prompt action to remediate them. Finally, use tradeoffs in your architecture to improve performance, such as using compression, using caching, or relaxing consistency requirements.

For prescriptive guidance on implementation, see the Performance Efficiency Pillar whitepaper.

This week

- AWS Well-architected Framework
  - Operational Excellence
  - Security
  - Reliability
  - Performance
  - **Cost Optimization**
- AWS Trusted Advisor

SWINBURNE UNIVERSITY OF TECHNOLOGY

Cost Optimization pillar

Cost Optimization pillar

Eliminate unneeded expense.

- Focus
  - Run systems to deliver business value at the lowest price point.
- Key topics
  - Understanding and controlling when money is being spent
  - Selecting the most appropriate and right number of resource types
  - Analyzing spending over time
  - Scaling to meeting business needs without overspending

The *Cost Optimization pillar* focuses on the ability to run systems to deliver business value at the lowest price point. Key topics include: understanding and controlling when money is being spent, selecting the most appropriate and right number of resource types, analyzing spending over time, and scaling to meeting business needs without overspending.

Cost optimization design principles

Cost Optimization pillar

Eliminate unneeded expense.

- Adopt a consumption model
- Measure overall efficiency
- Stop spending money on data center operations
- Analyze and attribute expenditure
- Use managed and application-level services to reduce cost of ownership

There are five design principles that can optimize costs:

- *Adopt a consumption model* – Pay only for the computing resources that you require. Increase or decrease usage depending on business requirements, not by using elaborate forecasting.
- *Measure overall efficiency* – Measure the business output of the workload and the costs that are associated with delivering it. Use this measure to know the gains that you make from increasing output and reducing costs.
- *Stop spending money on data center operations* – AWS does the heavy lifting of racking, stacking, and powering servers, which means that you can focus on your customers and business projects instead of the IT infrastructure.
- *Analyze and attribute expenditure* – The cloud makes it easier to accurately identify system usage and costs, and attribute IT costs to individual workload owners. Having this capability helps you measure return on investment (ROI) and gives workload owners an opportunity to optimize their resources and reduce costs.
- *Use managed and application-level services to reduce cost of ownership* – Managed and application-level services reduce the operational burden of maintaining servers for tasks such as sending email or managing databases. Because managed services operate at cloud scale, cloud service providers can offer a lower cost per transaction or service.

Cost optimization questions

Expenditure awareness

- How do you govern usage?
- How do you monitor usage and cost?
- How do you decommission resources?

Cost-effective resources

- How do you evaluate cost when you select services?
- How do you meet cost targets when you select resource type and size?
- How do you use pricing models to reduce cost?
- How do you plan for data transfer changes?

Matching supply and demand

- How do you match supply of resources with demand?

Optimizing over time

- How do you evaluate new services?

The foundational questions for cost optimization fall under four best practice areas: expenditure awareness, cost-effective resources, matching supply and demand, and optimizing over time.

Similar to the other pillars, there are tradeoffs to consider when evaluating cost. For example, you may choose to prioritize for speed—going to market quickly, shipping new features, or simply meeting a deadline—instead of investing in upfront cost optimization. As another example, designing an application for a higher level of availability typically costs more. You should identify your true application needs and use empirical data to inform your architectural design decisions. Perform benchmarking to establish the most cost-optimal workload over time.

For prescriptive guidance on implementation, see the Cost Optimization Pillar whitepaper.

The AWS Well-Architected Tool

- Helps you review the state of your workloads and compares them to the latest AWS architectural best practices
- Gives you access to knowledge and best practices used by AWS architects, whenever you need it
- Delivers an action plan with step-by-step guidance on how to build better workloads for the cloud
- Provides a consistent process for you to review and measure your cloud architectures
- Well-architected tool video

https://d3nn3d4w2aqyem.cloudfront.net/mp4/en/Getting_started_video.mp4

The activity that you just completed is similar to how you would use the AWS Well-Architected Tool.

The AWS Well-Architected Tool helps you review the state of your workloads and compare them to the latest AWS architectural best practices. It gives you access to knowledge and best practices used by AWS architects, whenever you need it.

This tool is available in the AWS Management Console. You define your workload and answer a series of questions in the areas of operational excellence, security, reliability, performance efficiency, and cost optimization (as defined in the AWS Well-Architected Framework). The AWS Well-Architected Tool then delivers an action plan with step-by-step guidance on how to improve your workload for the cloud.

The AWS Well-Architected Tool provides a consistent process for you to review and measure your cloud architectures. You can use the results that the tool provides to identify next steps for improvement, drive architectural decisions, and bring architecture considerations into your corporate governance process.

Key takeaways

- The AWS Well-Architected Framework provides a consistent approach to evaluate cloud architectures and guidance to help implement designs.
- The AWS Well-Architected Framework documents a set of foundational questions that enable you to understand if a specific architecture aligns well with cloud best practices.
- The AWS Well-Architected Framework is organized into five pillars.
- Each pillar includes a set of design principles and best practices.

Some key takeaways from this section of the module include:

- The AWS Well-Architected Framework documents a set of foundational questions that enable you to understand if a specific architecture aligns well with cloud best practices.
- The AWS Well-Architected Framework is organized into five pillars: operational excellence, security, reliability, performance efficiency, and cost optimization.
- Each pillar includes a set of design principles and best practices.

Module 9: Cloud Architecture

Section 2: Reliability and availability

aws academy

Section 2: Reliability and availability

aws academy

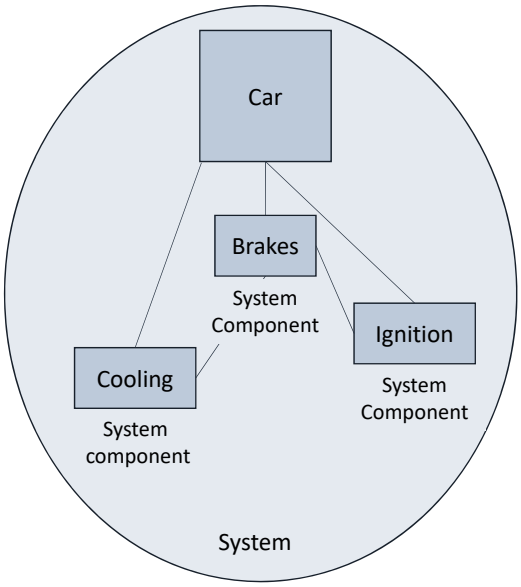
"Everything fails, all the time."

Werner Vogels, CTO, Amazon.com

In the words of Werner Vogels, Amazon's CTO, "Everything fails, all the time." One of the best practices that is identified in the AWS Well-Architected Framework is to plan for failure (or application or workload downtime). One way to do that is to architect your applications and workloads to withstand failure. There are two important factors that cloud architects consider when designing architectures to withstand failure: reliability and availability.

Reliability

- A measure of your system's ability to provide functionality when desired by the user.
- System includes all system components: hardware, firmware, and software.
- Probability that your entire system will function as intended for a specified period.
- Mean time between failures (MTBF) = total time in service/number of failures

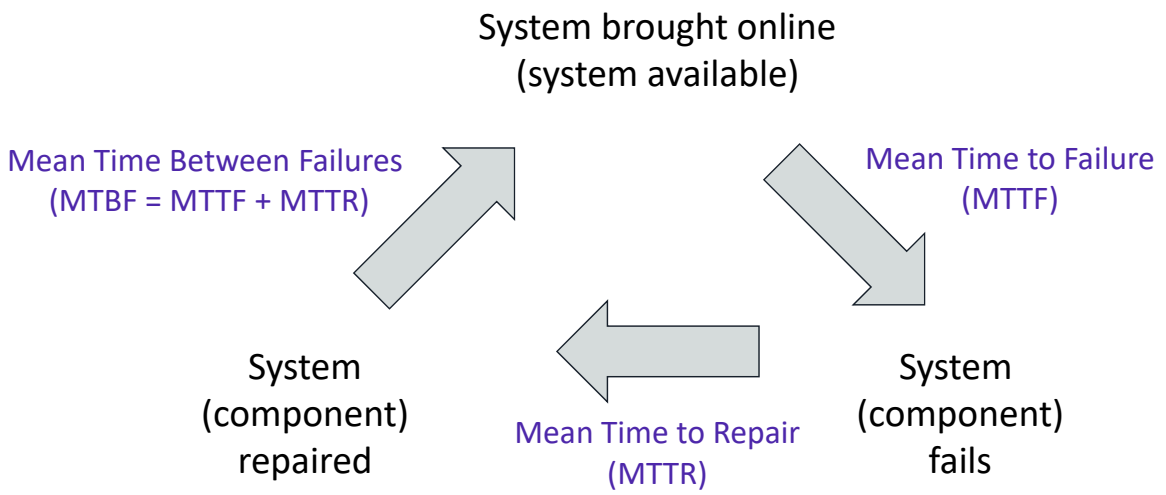


Reliability is a measure of your system's ability to provide functionality when desired by the user. Because "everything fails, all the time," you should think of reliability in statistical terms. Reliability is the probability that an entire system will function as intended for a specified period. Note that a system includes all system components, such as hardware, firmware, and software. Failure of system components impacts the availability of the system.

To understand reliability, it is helpful to consider the familiar example of a car. The car is the system. Each of the car's components (for example, cooling, ignition, and brakes) must work together in order for the car to work properly. If you try to start the car and the ignition fails, you cannot drive anywhere—the car is not available. If the ignition fails repeatedly, your car is not considered reliable.

A common way to measure reliability is to use statistical measurements, such as Mean Time Between Failures (MTBF). MTBF is the total time in service over the number of failures.

Understanding reliability metrics



Say that you have an application that you bring online Monday at noon. The application is said to be *available*. It functions normally until it fails Friday at noon. Therefore, the time to failure (or the length of time the application is available) is 96 hours. You spend from Friday at noon until Monday at noon diagnosing why the application failed and repairing it, at which point you bring the application back online. Therefore, the time to repair is 72 hours.

Then, it happens again: the application fails on Friday at noon, you spend from Friday at noon until Monday at noon repairing it, and you bring it online on Monday at noon.

Say this failure-repair-restore cycle happens *every week*. You can now calculate the average of these numbers. In this example, your mean time to failure (MTTF) is 96 hours, and your mean time to repair (MTTR) is 72 hours. Your mean time between failures (MTBF) is 168 hours (or 1 week), which is the sum of MTTF and MTTR.

Availability

- Normal operation time / total time
- A percentage of uptime (for example, 99.9 percent) over time (for example, 1 year)
- Number of 9s – Five 9s means 99.999 percent availability

As you just learned, failure of system components impacts the availability of the system.

Formally, *availability* is the percentage of time that a system is operating normally or correctly performing the operations expected of it (or normal operation time over total time). Availability is reduced anytime the application isn't operating normally, including both scheduled and unscheduled interruptions.

Availability is also defined as the percentage of uptime (that is, length of time that a system is online between failures) over a period of time (commonly 1 year).

A common shorthand when referring to availability is *number of 9s*. For example, *five 9s* means 99.999 percent availability.

High availability

- System can withstand some measure of degradation while still remaining available.
- Downtime is minimized.
- Minimal human intervention is required.

A *highly available* system is one that can withstand some measure of degradation while still remaining available. In a highly available system, downtime is minimized as much as possible and minimal human intervention is required.

A highly available system can be viewed as a set of system-wide, shared resources that cooperate to guarantee essential services. High availability combines software with open-standard hardware to minimize downtime by quickly restoring essential services when a system, component, or application fails. Services are restored rapidly, often in less than 1 minute.

Availability tiers

Availability	Max Disruption (per year)	Application Category
99%	3 days 15 hours	Batch processing, data extraction, transfer, and load jobs
99.9%	8 hours 45 minutes	Internal tools like knowledge management, project tracking
99.95%	4 hours 22 minutes	Online commerce, point of sale
99.99%	52 minutes	Video delivery, broadcast systems
99.999%	5 minutes	ATM transactions, telecommunications systems

Availability requirements vary. The length of disruption that is acceptable depends on the type of application. Here is a table of common application availability design goals and the maximum length of disruption that can occur within a year while still meeting the goal. The table contains examples of the types of applications that are common at each availability tier.

Factors that influence availability

Fault tolerance

- The built-in redundancy of an application's components and its ability to remain operational.

Scalability

- The ability of an application to accommodate increases in capacity needs without changing design.

Recoverability

- The process, policies, and procedures that are related to restoring service after a catastrophic event.

Though events that might disrupt an application's availability cannot always be predicted, you can build availability into your architecture design. There are three factors that determine the overall availability of your application:

- *Fault tolerance* refers to the *built-in redundancy* of an application's components and the *ability of the application to remain operational* even if some of its components fail. Fault tolerance relies on specialized hardware to detect failure in a system component (such as a processor, memory board, power supply, I/O subsystem, or storage subsystem) and instantaneously switch to a redundant hardware component. The fault-tolerant model does not address software failures, which are the most common reason for downtime.
- *Scalability* is the ability of your application to accommodate increases in capacity needs, remain available, and perform within your required standards. It does not guarantee availability, but it contributes to your application's availability.
- *Recoverability* is the ability to restore service quickly and without lost data if a disaster makes your components unavailable, or it destroys data.

Keep in mind that improving availability usually leads to increased cost. When you consider how to make your environment more available, it's important to balance the cost of the improvement with the benefit to your users.

Do you want to ensure that your application is always alive or reachable, or do you want to ensure that it is servicing requests within an acceptable level of performance?

Section 2 key takeaways

- Reliability is a measure of your system's ability to provide functionality when desired by the user, and it can be measured in terms of MTBF.
- Availability is the percentage of time that a system is operating normally or correctly performing the operations expected of it (or normal operation time over total time).
- Three factors that influence the availability of your applications are fault tolerance, scalability, and recoverability.
- You can design your workloads and applications to be highly available, but there is a cost tradeoff to consider.

Some key takeaways from this section of the module include:

- Reliability is a measure of your system's ability to provide functionality when desired by the user, and it can be measured in terms of MTBF.
- Availability is the percentage of time that a system is operating normally or correctly performing the operations expected of it (or normal operation time over total time).
- Three factors that influence the availability of your applications are fault tolerance, scalability, and recoverability.
- You can design your workloads and applications to be highly available, but there is a cost tradeoff to consider.

This week

- AWS Well-architected Framework
  - Operational Excellence
  - Security
  - Reliability
  - Performance
  - Cost
- **AWS Trusted Advisor**

SWINBURNE UNIVERSITY OF TECHNOLOGY

AWS Trusted Advisor

AWS Trusted Advisor

- Online tool that provides real-time guidance to help you provision your resources following AWS best practices.
- Looks at your entire AWS environment and gives you real-time recommendations in five categories.

AWS Trusted Advisor is an online tool that provides real-time guidance to help you provision your resources following AWS best practices.

AWS Trusted Advisor looks at your entire AWS environment and gives you recommendations in five categories:

- *Cost Optimization* – AWS Trusted Advisor looks at your resource use and makes recommendations to help you optimize cost by eliminating unused and idle resources, or by making commitments to reserved capacity.
- *Performance* – Improve the performance of your service by checking your service limits, ensuring you take advantage of provisioned throughput, and monitoring for overutilized instances.
- *Security* – Improve the security of your application by closing gaps, enabling various AWS security features, and examining your permissions.
- *Fault Tolerance* – Increase the availability and redundancy of your AWS application by taking advantage of automatic scaling, health checks, Multi-AZ deployments, and backup capabilities.
- *Service Limits* – AWS Trusted Advisor checks for service usage that is more than 80 percent of the service limit. Values are based on a snapshot, so your current usage might differ. Limit and usage data can take up to 24 hours to reflect any changes.

For a detailed description of the information that AWS Trusted Advisor provides, see AWS Trusted Advisor Best Practice Checks.

Activity: Interpret AWS Trusted Advisor recommendations

You have a friend who used AWS Trusted Advisor for the first time. She is trying to interpret its recommendations to improve her cloud environment and needs your help. This is her dashboard. While everything looks OK in the cost optimization and service limit categories, you notice that there are a few recommendations that you should review to help her improve her security, performance, and fault tolerance.

Help your friend interpret the following recommendations.

Activity: Recommendation #1

MFA on Root Account

Description: Checks the root account and warns if multi-factor authentication (MFA) is not enabled. For increased security, we recommend that you protect your account by using MFA, which requires a user to enter a unique authentication code from their MFA hardware or virtual device when interacting with the AWS console and associated websites.

Alert Criteria: MFA is not enabled on the root account.

Recommended Action: Log in to your root account and activate an MFA device.

For this recommendation, answer these questions:

- What is the status?
- What is the problem?
- What specific environment details are you given?
- What is the best practice?
- What is the recommended action?

Activity: Recommendation #2

IAM Password Policy

Description: Checks the password policy for your account and warns when a password policy is not enabled, or if password content requirements have not been enabled. Password content requirements increase the overall security of your AWS environment by enforcing the creation of strong user passwords. When you create or change a password policy, the change is enforced immediately for new users but does not require existing users to change their passwords.

Alert Criteria: A password policy is enabled, but at least one content requirement is not enabled.

Recommended Action: If some content requirements are not enabled, consider enabling them. If no password policy is enabled, create and configure one. See Setting an Account Password Policy for IAM Users.

For this recommendation, answer these questions:

- What is the status?
- What is the problem?
- What specific environment details are you given?
- What is the best practice?
- What is the recommended action?

Activity: Recommendation #3

Security Groups – Unrestricted Access

Description: Checks security groups for rules that allow unrestricted access to a resource. Unrestricted access increases opportunities for malicious activity (hacking, denial-of-service attacks, loss of data).

Alert Criteria: A security group rule has a source IP address with a /0 suffix for ports other than 25, 80, or 443.)

Recommended Action: Restrict access to only those IP addresses that require it. To restrict access to a specific IP address, set the suffix to /32 (for example, 192.0.2.10/32). Be sure to delete overly permissive rules after creating rules that are more restrictive.

Region	Security Group Name	Security Group ID	Protocol	Port	Status	IP Range
us-east-1	WebServerSG	sg-xxxxxxx1 (vpc-xxxxxxx1)	tcp	22	Red	0.0.0.0/0
us-west-2	DatabaseServerSG	sg-xxxxxxx2 (vpc-xxxxxxx2)	tcp	8080	Red	0.0.0.0/0

For this recommendation, answer these questions:

- What is the status?
- What is the problem?
- What specific environment details are you given?
- What is the best practice?
- What is the recommended action?

Activity: Recommendation #4

Amazon EBS Snapshots

Description: Checks the age of the snapshots for your Amazon Elastic Block Store (Amazon EBS) volumes (available or in-use). Even though Amazon EBS volumes are replicated, failures can occur. Snapshots are persisted to Amazon Simple Storage Service (Amazon S3) for durable storage and point-in-time recovery.

Alert Criteria:
Yellow: The most recent volume snapshot is between 7 and 30 days old.
Red: The most recent volume snapshot is more than 30 days old.
Red: The volume does not have a snapshot.

Recommended Action: Create weekly or monthly snapshots of your volumes

Region	Volume ID	Volume Name	Snapshot ID	Snapshot Name	Snapshot Age	Volume Attachment	Status	Reason
us-east-1	vol-xxxxxxxx	My-EBS-Volume				/dev/…	Red	No snapshot

For this recommendation, answer these questions:

- What is the status?
- What is the problem?
- What specific environment details are you given?
- What is the best practice?
- What is the recommended action?

Activity: Recommendation #5

Amazon S3 Bucket Logging
Description: Checks the logging configuration of Amazon Simple Storage Service (Amazon S3) buckets. When server access logging is enabled, detailed access logs are delivered hourly to a bucket that you choose. An access log record contains details about each request, such as the request type, the resources specified in the request, and the time and date the request was processed. By default, bucket logging is not enabled; you should enable logging if you want to perform security audits or learn more about users and usage patterns.
Alert Criteria:
Yellow: The bucket does not have server access logging enabled.
Yellow: The target bucket permissions do not include the owner account. Trusted Advisor cannot check it.
Recommended Action:
Enable bucket logging for most buckets.
If the target bucket permissions do not include the owner account and you want Trusted Advisor to check the logging status, add the owner account as a grantee.

Region	Bucket Name	Target Name	Target Exists	Same Owner	Write Enabled	Reason
us-east-2	my-hello-world-bucket		No	No	No	Logging not enabled

For this recommendation, answer these questions:

- What is the status?
- What is the problem?
- What specific environment details are you given?
- What is the best practice?
- What is the recommended action?